IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 28, 2019, accepted April 29, 2019, date of publication May 10, 2019, date of current version May 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2916035

Stereo Matching With Fusing Adaptive Support Weights

WENHUAN WU[1,2], HONG ZHU[1], SHUNYUAN YU[3], AND JING SHI[1]

[1]School of Automation and Information Engineering, Xi'an University of Technology, Xi'an 710048, China
[2]School of Electrical and Information Engineering, Hubei University of Automotive Technology, Shiyan 442002, China
[3]School of Electronic and Information Engineering, Ankang University, Ankang 725000, China

Corresponding author: Hong Zhu (zhuhong@xaut.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61771386, Grant 61673318 and Grant 61801005.

ABSTRACT Stereo matching has been widely used in various computer vision applications and it is still a challenging problem. Adaptive support weights (ASW) methods represent the state of the art in stereo matching and have achieved outstanding performance. However, the local ASW methods fail to resolve the matching ambiguity in low texture areas because their cost aggregation is limited within local fixed or adaptive support windows. On the other hand, the non-local ASW methods perform cost aggregation along a special tree, so that these methods are often sensitive to high texture areas since some useful connectivity constrains between adjacent pixels are broken during constructing the special tree. To solve these problems, in this paper, a novel and generic fusing ASW framework are proposed for stereo matching. In this framework, we establish dual support windows for each pixel, i.e., a local window and the whole image window. As such, the primitive connectivity between each pixel and its neighboring pixels in the local window can be maintained, and then each pixel not only gets appropriate supports from neighboring pixels within its local support window but also receives more adaptive supports from the other pixels outside the local window. Furthermore, a local edge-aware filter and a non-local edge-aware filter, whose kernel windows correspond to the dual support windows, are merged in order to achieve collaborative filtering of the cost volume. The performance evaluation on the Middlebury and KITTI datasets shows that the proposed stereo matching method outperforms the current state-of-the-art methods.

INDEX TERMS Stereo matching, cost aggregation, adaptive support weight, edge-aware filtering.

I. INTRODUCTION

Aiming to endow computers with human-like depth vision capabilities, binocular stereo matching remains one of the most active research topics in computer vision since it plays a crucial role in many applications, including 3D scene reconstruction, 3D tracking and autonomous driving. The goal of stereo matching is to generate a dense disparity map by finding all the correspondences between two rectified images from the same scene which are captured from different viewpoints. Due to the ambiguous nature of the matching problem and existing noise, occlusion, or distortion in the images, stereo matching is very challenging and the recovery of an accurate disparity map still remains an open problem.

A large number of studies have been conducted to solve this problem. An extensive review of the stereo matching algorithms can be found in [1]. According to the taxonomy and evaluation strategy in [1], stereo matching algorithms can be mainly categorized into global methods and local methods. Global methods typically compute all disparities simultaneously by minimizing an energy function defined on the Markov random field model using a global optimization algorithm, e.g., graph cut [2], belief propagation [3] or dynamic programming [4]. Global methods tend to produce more accurate matching results. However, they are generally computationally expensive due to the iterative nature of the underlying optimization process. Local methods consider correlations between adjacent pixels in support windows. Firstly, the matching cost of each pixel at each disparity is calculated with some measurement of pixel similarity. Secondly, in order to reduce the matching ambiguity, the raw matching

The associate editor coordinating the review of this manuscript and approving it for publication was Shaohui Liu.

costs of all pixels within a support window are aggregated to the center pixel at each disparity. Then, an optimal disparity that gives a minimum aggregated cost is chosen through an efficient local optimization process. Compared with global methods, local methods have more efficient computational performance and can better satisfy the requirement of practical applications.

In local methods, there is an implicit smoothness assumption that all pixels in a support window have similar disparities. However, this assumption is broken at depth discontinuities where the support window contains pixels from different depths, and this leads to the well-known edge fattening effect. On the other hand, local methods also cannot work well for large regions with low or repetitive texture since the support windows of these methods are often limited in a pre-defined small window. Thus, to obtain accurate results not only at depth discontinuities but also in homogeneous regions, an appropriate support window should be selected for each pixel adaptively. To this end, variable and multiple support windows methods in early papers were proposed by changing window size [5], shape [6] or center offset [7]. Among these methods, rectangular and constrained-shaped window models may be inappropriate for pixels near arbitrarily shaped depth discontinuities. Additionally, computational cost over multiple windows for each pixel also increases tremendously. To resolve this problem, segmentation-based methods [8] use segmented regions with arbitrary sizes and shapes as the support windows, which were implicitly assumed that the disparity varies smoothly in each region. However, these methods require precise color segmentation that is very difficult when dealing with highly textured images. Furthermore, segmentation-based methods may fail if segments overlap depth boundaries.

The major breakthrough in local methods is the introduction of the adaptive support weights (ASW) methods [9], [11]–[14]. These methods achieve an accuracy comparable to that of global methods. The key idea in ASW methods is to assign an appropriate weight for each pixel within the support window. The support weight represents the probability that the center pixel and a neighbor pixel might belong to the same region. In other words, ASW methods can be treated as segmenting the reference image in a ''soft'' way. Yoon and Kweon [9] firstly introduced the ASW strategy for stereo matching. In their method, a pixel's weight inside a support window is computed based on the color similarity and spatial distance to the center pixel. Note that this is equivalent to the way that weights are computed in bilateral filter (BF) [10] with edge-aware property, so the cost aggregation step of their method can be understood as filtering the cost volume with BF. Hence, ASW methods are also referred as cost volume filtering methods. In [11], the segmented BF weight function that relies on a pre-computed color segmentation was proposed. Hosni *et al.* [12] defined the weights within a window by computing the geodesic distance to the center pixel. The main disadvantage of the above ASW methods is that their computational complexity directly depends on the size of the support windows. As a consequence, these methods performs very slowly since the support windows have to be sufficiently large (e.g. 35×35 in [9]) to better handle low texture regions. In order to reduce the processing time, several acceleration techniques for speeding up the BF weight function had been proposed [13], [14]. However, these fast approximation methods often sacrifice the output quality for high computational speed.

Inspired by BF, various edge-aware filtering techniques have been introduced for better estimating support weights. The guided filter (GF) proposed by He *et al.* [15], which has a runtime independent of the kernel window size, exhibited its superiority over BF on both quality and efficiency. Rhemann *et al.* [16] utilized the GF for filtering the cost volume and outperformed most local methods in terms of both speed and accuracy. To avoid the kernel windows of GF covering the object boundaries or depth discontinuities, several improved GF-based methods which combines with adaptive support windows have been presented, such as adaptive guided filtering [17] and cross-based local multi-point filtering [18]. Hamzah *et al.* [19] presented the iterative guided image filter (IGF) [20] and utilized a cascade model of IGF and BF to filter the cost volume for better preserving the object edges. To improve the performance of the ASW methods, Zhang *et al.* [21] adopted the coarse-to-fine strategy to perform cross-scale cost aggregation and enforced cost volume consistency across multiple scales. To address the dependence problem of the support window size, some recursive edge-aware filters [22], [23] for stereo matching have been presented. However, these recursive filters often carry out cost aggregation in row or column, so a pixel cannot directly get support from those adjacent pixels in other directions. Recently, Yang [24] proposed a non-local cost aggregation approach [25] with extremely low computational complexity, in which the matching costs are aggregated adaptively along a minimum spanning tree (MST). Mei *et al.* [26] employed a segment tree to perform the non-local cost aggregation strategy. In addition, a cross-trees structure consisting of a horizontal tree and a vertical tree is proposed for non-local cost aggregation in [27]. Different from aforementioned various local ASW methods whose support regions are often local fixed-size or adaptive windows, the non-local methods take the whole image as its supporting window. Although the non-local methods can better handle low texture regions, they show poor performance in highly textured regions.

In this paper, in order to improve robustness to lack of texture or highly textured regions, we propose a novel fusing ASW strategy for stereo matching by collaboratively performing cost aggregation on dual support windows with dual edge-aware filters. Specifically, we embed a local edge-aware filter into a non-local edge-aware filter to achieve collaborate filtering of the cost volume. Furthermore, in order to balance accuracy and speed, we use the original GF to filter the cost volume over the local support window, and adopt the MST filter to perform the non-local cost aggregation over the whole image. Then the fusing aggregated cost volume

is the average value of the outputs of the above two filterings. In fact, the proposed fusing ASW strategy is a general framework and its performance may be improved when more outstanding edge-aware filter, no matter whether the local one or the non-local one, is integrated into this general fusing framework. Experiments on the Middlebury benchmark [28] and KITTI benchmark [29] demonstrate the effectiveness of the proposed method and show that our method is one of the state-of-the-art stereo matching algorithms.

The remainder of this paper is organized as follows. In Section II, the related work is discussed. A novel stereo matching algorithm using fusing ASW is systematically described in Section III. Experimental results and analyses are presented in Section IV. Section V concludes this paper.

II. RELATED WORK

A recent comparative study on various weight functions was carried out in [30], so we refer readers to the survey to get an overview of different ASW methods. Since GF shows its quality and speed advantages over the other most local edge-aware filters (e.g. BF), here we will focus on the GF-like local ASW methods and the non-local ASW methods, which are very relevant to our method.

Hosni *et al.* [16] took one image among stereo image pair as the guided image and utilized GF to asymmetrically perform the cost volume filtering. In contrast, in order to aware both edges of left and right images in stereo matching, Zhang *et al.* [31] employed symmetric linear regression model. Obviously, the computational cost of the symmetric guided filter is much more expensive than GF. In order to improve the edges of object matching, Hamzah *et al.* [19] proposed the iterative guided filter (IGF)[20], and then used its weighted form which is obtained by concatenating IGF with BF to conduct cost aggregation. However, this cascade manner greatly increases computational load.

Considering that the simple box support window with fixed size in the above guided image filters easily overlaps object boundaries and depth discontinuities, Yang *et al.* [17] proposed an improved stereo matching method using the adaptive guided filtering (AGF). The AGF adopts adaptive rectangular support window instead of the fixed square window, and applies the integral image technique to achieve a linear computational complexity independent of the window size. In fact, AGF defines a general form of GF's weight function by varying the rectangle window size. To obtain adaptive shape support regions, the cross-based local multipoint filtering (CLMF) [18] uses adaptive cross-based support regions presented by Dai *et al.* [32] as the support windows. CLMF adopts orthogonal integral image technique in [32] for fast filtering over any arbitrarily shaped windows. However, when the linear regression model as in GF is used in CLMF, the computational complexity of CLMF is much higher than GF [16] and AGF [17], and it becomes larger with increasing support region size. Moreover, in contrast to GF, both CLMF and AGF need an additional overhead for each pixel to construct an upright cross support skeleton with four varying arms. Notice that both adaptive rectangular windows and adaptive shape windows are still limited by user-specified maximum length of the skeleton arms. Thus, the above improved GF-based methods like other local window-based methods still can not perform well for low texture regions especially large textureless regions.

In order to solve this problem, by introducing the coarse-to fine (CTF) strategy into cost aggregation, Lu *et al.* [21] reformulated the cost volume filtering from a weighted least-squares (WLS) optimization perspective and introduced an inter-scale regularization term into the WLS optimization objective to enforce the consistency of the cost volume across multiple scales. This cross-scale strategy can effectively improve the disparity accuracy at expense of some extra computational load. On the other hand, Yang [24], [25] proposed a non-local cost aggregation method by building a MST over the image graph. In this method, each pixel is able to get supports from all the other pixels of the image through unique paths on a MST, so the support window of the non-local cost aggregation method is the whole image. The non-local cost aggregation method performs better around low texture regions than local ASW methods, since the former can guarantee to cover the whole low texture regions. Besides, the non-local cost aggregation method [25] has great advantage in extremely low computational complexity. By enforcing tight connections for the pixels inside the same segment, Mei *et al.* [26] proposed a segment-tree (ST) instead of MST to perform the non-local cost aggregation strategy. However, ST is easily influenced by the accuracy of image segmentation and it also requires additional overhead for image segmentation. In [27], two crossed trees, namely horizontal tree and vertical tree, were employed, and the non-local cost aggregation strategy is done twice by traversing the two crossed trees successively. Although these non-local ASW methods work well for low texture regions, they can not handle highly textured or noised regions well. It is because that some useful connectivity constrains between local neighboring pixels are removed when these non-local ASW methods construct their special tree structures.

Inspired by tree filtering (TF) [33] and fully connected guided filtering (FCGF) [34], we propose a novel fusing ASW strategy to improve the accuracy of stereo matching. Both TF and FCGF relax their kernel windows to the entire image, and define the spatial affinity between pixels by building a MST as in [25]. To smooth out high-contrast details while preserving major edges, TF adopts the non-local aggregation algorithm [25] to smooth the outputs of BF in a cascade manner as in [19]. To boost the performance of GF, FCGF introduces the spatial affinity defined on the MST into the fully connected linear regression model, and substitutes the box filterer with the non-local aggregation algorithm [25] to compute the new model. Obviously, both TF and FCGF have high computational burden. Besides, they consider spatial affinity rather than intensity similarity between pixels, so TF and FCGF are not suitable for stereo matching. In contrast, our fusing ASW strategy for stereo

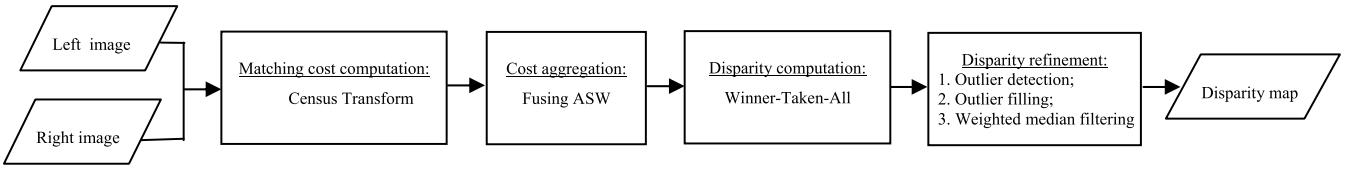


FIGURE 1. The flowchart of the proposed algorithm.

matching averages the weights of GF and the MST filter that is derived from the non-local aggregation algorithm [25]. As such, when performing cost aggregation with the proposed fusing ASW, the MST filter can effectively compensate for the deficiencies of GF, and vice versa. The computational complexity of our fusing method is approximate to that of GF since the MST filter has extremely low complexity and it is several times faster than GF.

III. THE PROPOSED ALGORITHM

Stereo matching methods typically consist of the four steps: 1) matching cost computation; 2) cost aggregation using ASW, i.e., cost volume filtering; 3) disparity computation; and 4) disparity refinement. An overview of the proposed algorithm is shown in Fig.1. Following this pipeline, the proposed algorithm will be described in detail below.

A. MATCHING COST COMPUTATION

In this step, the initial cost volume is constructed by computing the matching cost at each pixel at each disparity. The census transform [35] encodes each pixel value into a bit string representing the relative ordering of the neighboring pixels, and therefore is robust against radiometric changes. The census transform also shows the best overall performance in the evaluation on various matching costs [36]. Thus, we use the census transform as the measurement of pixel similarity to compute matching cost. Let $\mathcal{D}$ denote the set of allowed disparity levels. Here the left image is taken as the reference image. Given a pixel $p = (x, y)$ in the left image and an allowed disparity value $d \in D$, the corresponding pixel on the right image is denoted as $p_d = (x - d, y)$.

For a pixel p, its census transform is computed by comparing its intensity with the intensities of the neighboring pixels in a fixed-size window $\mathcal{N}(p)$ around p. The results of these comparisons are then concatenated into a single bit string. Thus, the census transform of pixel p is formulated as

$$cen(p) = \otimes_{q \in N(p)} \xi(p, q) \tag{1}$$

where $\otimes$ represents the concatenation operation for generating the bit string $cen(p)$, and $\xi(p, q)$ is a binary function defined as follows

$$\xi(p, q) = \begin{cases} 1, & if\ I(q) < I(p) \\ 0, & else \end{cases} \tag{2}$$

where $I(p)$ and $I(q)$ are the gray value of pixel p and pixel q.

Accordingly, after applying the census transform to both the left and right images, the matching cost of pixel p with disparity d is the Hamming distance of $cen(p)$ and $cen(p_d)$ expressed as follows

$$C(p, d) = Hamming\,(cen(p), cen\,(p_d)) \tag{3}$$

The computational complexity of computing matching cost based on the census transform mainly depends on the window size of the census transform. The window size of the census transform is set experimentally to 7×7, in order to reduce computational amount without affecting the quality of similarity measurement.

Once the matching costs $C(p, d)$ for all pixels and all possible disparity levels are computed, we can obtain the initial cost volume C which is a 3D array.

B. COST AGGREGATION WITH FUSING ASW

Cost aggregation which aggregates each pixel's matching cost over a support window is the most important step to reduce the matching ambiguities and noise in the initial cost volume. For ASW stereo methods, the cost aggregation is formulated by filtering the cost volume. To be more precise, the filtered cost value of pixel p at disparity d is a weighted average of all pixels in a support window in the d^{th} xy-slice, which can be formulated as:

$$C^A(p, d) = \sum\nolimits_{q \in \omega_p} W(p, q) C(q, d) \tag{4}$$

where ω_p denotes a support window centered at pixel p, and $W(p, q)$ is usually the weight function of a specified edge-aware filter such as BF and GF. Note that the filter weight $W(p, q)$ depends on the guidance image, which is the reference image I in the case of stereo matching.

As described in previous section, the local ASW methods using local support windows as shown in Fig.2 (a), no matter whether the fixed-size windows or the adaptive windows, cannot work well for low texture regions whose sizes are often arbitrary and unknowable. To avoid designing the optimal adaptive windows, the non-local ASW methods directly take the whole image as the global support window of each pixel and then exploit a specified tree to perform the cost aggregation as shown in Fig.2 (b). The non-local ASW methods show good performance in low texture regions, but these methods are sensitive to high texture regions since the original local Markov random fields around the pixel of interest is broken during constructing the special tree. To better deal with both low texture regions and high texture regions, we

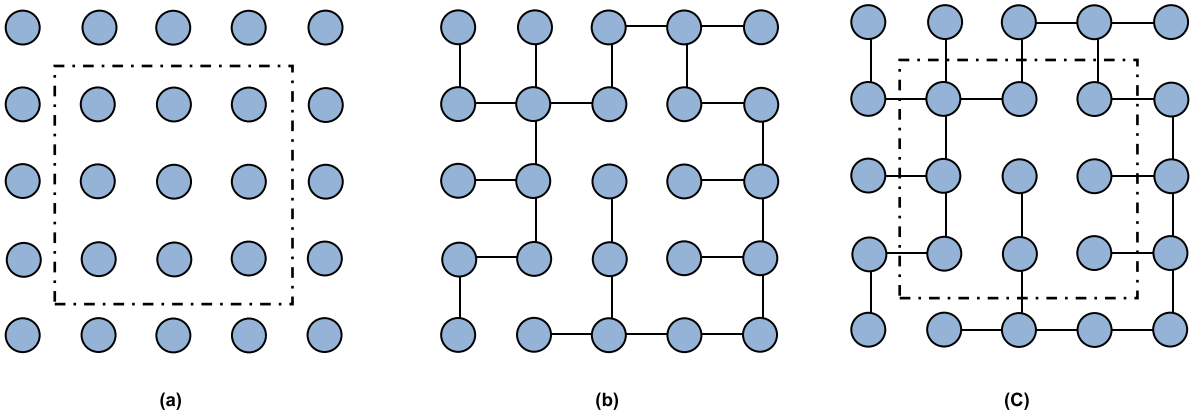


FIGURE 2. The support regions of different ASW methods. (a) The local ASW methods use local support region denoted with the dashed box to perform cost aggregation. The center pixel only gets supports from its neighboring pixels within the local window. (b) The non-local ASW method take the whole image as its support region. The center pixel can receive supports from all pixels of the image along a tree (e.g., MST). However, the connected edges between the center pixel and adjacent pixels may be lost. (c) The proposed fusing ASW method has two support regions, i.e., the local support window and the whole image, which correspond to the kernel windows of the local edge-aware filter and the non-local edge-aware filter respectively. Thus, the connected edges in the local window can be maintained. The center pixel not only gets proper supports from all neighboring pixels within the local window, but also gets adaptive supports from the other pixels outside the local window.

propose a novel fusing ASW strategy, that is, a local edge-aware filter and a non-local edge-aware filter are merged to collaboratively complete cost volume filtering. Different from the cascade way as in [19], [33], the weight function of our fusing ASW is the average value of the weight functions of the local filter and the non-local filter. Accordingly, as illustrated in Fig.2(c), we establish dual support regions for each pixel, i.e., a local window and the whole image, which correspond to the kernel windows of the local filter and the non-local filter, respectively. As such, the important connected edges in the local window can be preserved, and each pixel not only gets proper supports from its neighboring pixels within the local window, but also gets more adaptive supports from the other pixels outside the local window. Considering that GF not only can produce higher quality results than most local edge-aware filters, but it also has very low computational complexity which is linear to the number of image pixels, so we adopt GF with a small fixed-size window to perform local cost aggregation. On the other hand, the non-local cost aggregation algorithm [25] extracts a MST to conduct non-local cost aggregation. Note that the MST is also edge-aware because edges with large intensity difference are removed during spanning tree construction. Hence, when MST and GF are fused for cost aggregation, they can promote the performance of each other. In order to amalgamate GF and MST on the same scale, we derive a new *MST filter* (MF) from the MST non-local aggregation algorithm [25]. Then the MST filter is used to perform non-local cost aggregation. Note that the MST non-local cost aggregation has extremely low computational complexity and it is several times faster than GF as reported in [25]. Thus, for cost aggregation, the computational complexity of our ASW method fusing GF and MF is close to that the GF-based ASW method since the filtering process of MF only requires a small amount of extra computational load. Below we will describe the formulation of the proposed fusing ASW framework and the process of filtering the cost volume using GF and MF respectively.

Suppose that $W_{GF}(p, q)$ is the weight function of the original GF [15] as a local filter which is defined in a local squared window ω_k, and $W_{MF}(p, q)$ is the weight function of MF as a non-local filter which is defined in the whole reference image I. In order to achieve the proposed fusing ASW framework, we reformulate an extended weight function $W'_{GF}(p, q)$ of GF by relaxing the local definition domain to the entire image I as follows

$$W'_{GF}(p, q) = \begin{cases} W_{GF}(p, q), & q \in \omega_k \\ 0, & q \notin \omega_k \end{cases} \tag{5}$$

where ω_k is a square window centered at pixel k.

The weight function $W_{FA}(p, q)$ of our fusing ASW is the average value of $W'_{GF}(p, q)$ and $W_{MF}(p, q)$ as

$$W_{FA}(p, q) = (1/2)(W'_{GF}(p, q) + W_{MF}(p, q)) \tag{6}$$

To get the aggregated cost with the proposed fusing ASW strategy, by substituting (6) into (4), we have

$$\begin{aligned} C^A(p, d) &= \sum\nolimits_{q \in I} W_{FA}(p, q) C(q, d) \\ &= \sum\nolimits_{q \in I} (1/2)(W'_{GF}(p, q) + W_{MF}(p, q))) C(q, d) \\ &= (1/2)\left(C^A_{GF}(p, d) + C^A_{MF}(p, d)\right) \end{aligned} \tag{7}$$

where C^A_{GF} is the filtered cost volume with GF and C^A_{MF} is the filtered cost volume with MF. It can be seen that the final aggregated cost volume C^A with the proposed fusing ASW strategy is the average value of the two filtered cost volume. Suppose that the guidance image I is a color one. Next, we will use GF and MF to smooth the initial cost volume, respectively.

1) COST VOLUME FILTERING WITH GF

According to Formula (4), each slice of the initial cost volume C should be filtered respectively. Thus, for each disparity level d, we apply GF to filter the d^{th} xy-slice $C(\cdot, d)$ of C and then the output of filtering is the d^{th} xy-slice $C^A_{GF}(\cdot, d)$ of C^A_{GF}. Note that the weight function $W_{GF}(p, q)$ of GF does not need to be calculated explicitly, and the filtering output can be obtained directly according to the definition of GF for simplifying calculation. The key assumption of GF is a local linear regression model between the guidance image I and the filtering output. Specifically, for the cost volume filtering, it is assumed that the filtering output $C^A_{GF}(\cdot, d)$ is a linear transform of the guidance image I in the window ω_k defined as

$$C^A_{GF}(p, d) = \boldsymbol{a}_k^{\mathrm{T}} \boldsymbol{I}(p) + b_k, \quad \forall p \in \omega_k \tag{8}$$

where $\boldsymbol{a}_k$ is a 3×1 coefficient vector and b_k is a scalar, and $\boldsymbol{I}(p)$ is the 3×1 color vector of pixel p. The values of $\boldsymbol{a}_k$ and b_k are then obtained by minimizing a energy function $E(\boldsymbol{a}_k, b_k)$ in the window ω_k which is defined as

$$E(\boldsymbol{a}_k, b_k) = \sum\nolimits_{p \in \omega_k} \left(\left(\boldsymbol{a}_k^{\mathrm{T}} \boldsymbol{I}(p) + b_k - C(p, d) \right)^2 + \epsilon \boldsymbol{a}_k^{\mathrm{T}} \boldsymbol{a}_k \right) \tag{9}$$

Here, ϵ is a regularization parameter for penalizing large $\boldsymbol{a}_k$. The solution of $\boldsymbol{a}_k$ and b_k is computed by the linear regression as

$$\boldsymbol{a}_k = (\boldsymbol{\Sigma}_k + \epsilon \boldsymbol{U})^{-1} \left(\frac{1}{|w_k|} \sum\nolimits_{p \in \omega} \boldsymbol{I}(p) C(p, d) - \boldsymbol{\mu}_k \overline{C(k, d)} \right) \tag{10}$$

$$b_k = \overline{C(k, d)} - \boldsymbol{a}_k^{\mathrm{T}} \boldsymbol{\mu}_k \tag{11}$$

where $\boldsymbol{\mu}_k$ is the 3×1 mean vector and $\boldsymbol{\Sigma}_k$ is the 3×3 covariance matrix of $\boldsymbol{I}$ in ω_k respectively. $\boldsymbol{U}$ is a 3×3 identity matrix. $|\omega_k|$ is the number of pixels in ω_k. $\overline{C(k, d)}$ is the mean of the input cost slice $C(\cdot, d)$ in ω_k.

Note that the pixel p is covered by several different windows ω_k and each window ω_k yields a different output value of $C^A_{GF}(p, d)$ in (8) with different $\boldsymbol{a}_k$ and b_k. In order to enhance robustness, we average all the output values generated by all windows ω_k covering the pixel p. Hence the final filtering output is given as follows

$$C^A_{GF}(p, d) = \overline{\boldsymbol{a}}_{\boldsymbol{p}}^{T} \boldsymbol{I}(p) + \overline{b}_p \tag{12}$$

where $\overline{\boldsymbol{a}}_p = \frac{1}{|\omega_k|} \sum_{k \epsilon \omega_p} \boldsymbol{a}_k$ and $\overline{b}_p = \frac{1}{|\omega_k|} \sum_{k \epsilon \omega_p} b_k$ are the average coefficients of all windows overlapping the pixel p.

These calculations can be efficiently implemented with a sequence of box filtering using the integral image technique as described in [15], which makes the runtime independent of the window size. During the cost volume filtering, GF should be performed once for each disparity level, so the computational complexity of cost volume filtering with GF is $O(N|D|)$ where N is the number of pixels in the guidance image and $|\mathcal{D}|$ is the number of disparity levels in $\mathcal{D}$.

2) COST VOLUME FILTERING WITH MF

For the non-local cost volume filtering, the guidance image I is represented as a connected, undirected graph $G = (V, E)$, where each node in V corresponds to a pixel in I, and each edge in E connects a pair of the nearest neighboring pixels. For an edge e connecting pixels s and r, its weight is decided as follows:

$$\rho_e = \rho(s, r) = \|\boldsymbol{I}(s) - \boldsymbol{I}(r)\|_\infty \tag{13}$$

where $\boldsymbol{I}(s)$ and $\boldsymbol{I}(r)$ represent the color vector of pixels s and r respectively. Here the L_∞ norm of the vector is used to ensure that only when two neighboring pixels have similar intensity in all RGB channels, their edge is assigned a high weight. Experiments also verify that using L_∞ norm in (13) performs better than using other norms such as L_1. Then a MST with the minimum sum of the edge weights can be constructed by selecting $N-1$ edges with small weights. The intuition is that an edge is less likely to cross the depth borders if its two nodes have higher intensity similarity. For any two pixels p and q, there is one unique path connecting them on the MST, and their distance $D(p, q)$ is determined by the sum of the edge weights along their path as follows:

$$D(p, q) = D(q, p) = \sum\nolimits_{i=1}^{n-1} \rho(q_i, q_{i+1}) \tag{14}$$

The support weight $K(p, q)$ between p and q is defined as:

$$K(p, q) = \exp(-\frac{D(p, q)}{\sigma}) \tag{15}$$

where σ is a user-defined parameter used to adjust the weight between two nodes. The pixel with shorter distance from p is assigned a larger support weight. As far as the cost volume filtering is concerned, the support weight $K(p, q)$ does not need to be normalized. However, as described in [15], the weights of GF are normalized automatically during the filtering with the local linear regression model. As such, the support weight $K(p, q)$ should also be normalized for fusing with GF on the same scale. The normalized version of $K(p, q)$ corresponds to a new filter, namely *MST filter*. Thus, the weight function $W_{MF}(p, q)$ of the MST filter (MF) is expressed as:

$$W_{MF}(p, q) = \frac{K(p, q)}{\sum_{q \in I} K(p, q)} \tag{16}$$

Similar to the cost volume filtering with GF, we apply MF to filter each slice of the initial cost volume C respectively. For the disparity level d, the output of filtering d^{th} xy-slice $C(\cdot, d)$ with MF is $C^A_{MF}(\cdot, d)$ which is computed as a weighted sum of $C(\cdot, d)$ as follows

$$C^A_{MF}(p, d) = \sum\nolimits_{q \in I} W_{MF}(p, q) C(q, d) \tag{17}$$

Substituting (16) into (17), we have

$$C^A_{MF}(p, d) = \frac{\sum_{q \in I} K(p, q) C(q, d)}{\sum_{q \in I} K(p, q)} \tag{18}$$

Directly computing $C^A_{MF}(p, d)$ for each pixel p by using (18) iteratively is really time-consuming. Fortunately,

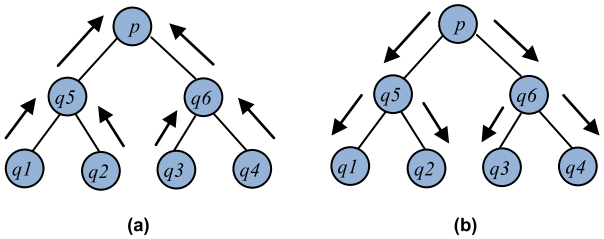


FIGURE 3. Two–pass cost aggregation on a tree. Here the pixel p is treated as the root node of the tree. (a) 1st pass: From leaf to root; (b) 2nd pass: From root to leaf.

both the numerator and denominator can be computed efficiently using the MST non-local aggregation algorithm [25]. Here we briefly describe how to rapidly calculate the numerator for all pixels at the same time. The denominator as the normalization term in (18) can be solved in the same way as the numerator since the denominator is the special case of the numerator with $C(q, d) \equiv 1$. Let $S^A(p, d)$ denote the numerator of $C^A_{MF}(p, d)$ in (18), i.e.,

$$S^A(p, d) = \sum_{q \in I} K(p, q) C(q, d) \tag{19}$$

Yang [25] proved that $S^A(p, d)$ for all pixels can be in one breath computed efficiently by traversing the MST in two sequential passes as illustrated in Fig.(3). In the first pass, the MST is traced from the leaf nodes to the root node. The intermediate aggregated cost $S^{A\uparrow}(p, d)$ for pixel p can be computed by using the following equation:

$$S^{A\uparrow}(p, d) = C(p, d) + \sum_{q \in Ch(p)} K(p, q) \cdot S^{A\uparrow}(q, d) \tag{20}$$

where the set $Ch(p)$ contains all the children of pixel p. After the first pass, the root node receives the weighted costs from all the other nodes, while the rest only receive the costs from their sub-trees. In the second pass, the tree is traversed from the root node to the leaf nodes. For pixel p, its final aggregated cost $S^A(p, d)$ is determined with its parent $Pr(p)$ as follows:

$$\begin{aligned} S^A(p, d) = {} & K(Pr(p), p) \cdot S^A(Pr(p), d) \\ & + (1 - K^2(Pr(p), p)) \cdot S^{A\uparrow}(p, d) \end{aligned} \tag{21}$$

For each disparity level d, the computational complexity of computing $S^A(p, d)$ for all pixels by using the above two sequential passes is $O(N)$. Then the computational complexity of computing $C^A_{MF}(p, d)$ for all pixels at each disparity is also $O(N)$ since the denominator in (18) can be solved in the same way as the numerator. Hence, the computational complexity of filtering cost volume with MF is $O(N|\mathcal{D}|)$ which is the same as that of filtering cost volume with GF.

Once the initial cost volume C is filtered with GF and MF respectively, the final aggregated cost volume C^A is the average value of the two filtered cost volume as expressed in (7). Hence, the total computational complexity of computing the final aggregated cost volume C^A is $O(N|\mathcal{D}|)$.

C. DISPARITY COMPUTATION

In this step, we adopt a commonly used winner-take-all (WTA) strategy to compute the disparity map from the aggregated cost volume C^A. The WTA strategy generates each pixel's optimal disparity d_p by choosing the disparity with the lowest aggregated cost value in all allowed disparity levels according to

$$d_p = arg \ \min_{d \in \mathcal{D}} C^A(p, d) \tag{22}$$

Then the raw disparity map can be obtained by mapping each pixel to its own optimal disparity level according to the above WTA strategy.

D. DISPARITY REFINEMENT

The raw disparity map usually contains a lot of outliers (i.e., invalid pixels) whose disparity values are invalid, especially near depth discontinuities and in occluded regions. In order to handle these outliers and make the disparity map more accurate, we adopt the disparity refinement method proposed in [16] to post-process the raw disparity map. First, to detect the outliers, the left-right consistency check is applied on the left and right raw disparity maps which are generated when the left image and right image are used as the reference image respectively. A pixel is marked as the outlier whose disparity value is not identical to that of its matching point. Next, the scan line filling technique is performed to fill the detected outliers. For each outlier, we extract the disparity values of the closest valid pixel to the left and to the right of the current outlier. Then the outlier is assigned to the minimum value between the two valid disparity values due to the nature of the occlusion in which the occluded pixels belong primarily to the background objects. Finally, the disparity map is smoothed using a weighted median filter [16] to remove streak-like artifacts that usually are produced by the filling process.

IV. EXPERIMENTAL RESULTS

The proposed stereo matching with fusing ASW (FASW) is implemented in C++. We evaluate our method on both the Middlebury benchmark [28] and the KITTI benchmark [29]. All the following experiments are conducted on a PC with a 3.2 GHZ Intel Core i5-6500 CPU and 8-GB memory.

A. EXPERIMENTAL SETTINGS

We mainly focus on two benchmarks: the Middlebury benchmark [28] from the indoor scene, and the KITTI benchmark [29] from the outdoor scene. First, we online evaluate the performance of the proposed stereo matching algorithm on the version 3 of Middlebury stereo evaluation. This new stereo evaluation adopts the newest dataset v3 [37] consisting of a training set and a test set. The training set comprises 15 stereo pairs with publicly available ground truth disparity maps and they are used to determine the parameters for stereo matching algorithms, while the ground truth disparity maps of all 15 stereo pairs in the test set are not released. The disparity map results of all 15 stereo pairs in the test set have to be uploaded into the Middlebury online system for

TABLE 1. The error rates of stereo matching methods on 15 test stereo pairs from Middlebury dataset v3.

Data	DDL[39]	IGF[20]	DSGCA[40]	ISM[19]	PSMNet_ROB[41]	ADSM[42]	MPSV[43]	BSM[44]	DoGGuided[45]	DF[46]	FASW
Australia	23.8	22.4	23.4	20.5	**17.0**	18.8	39.8	42.3	26.0	27.5	20.2
AustraliaP	8.11	7.03	8.34	9.20	13.1	**5.99**	14.6	12.0	9.88	14.0	6.50
Bicycle2	13.2	11.1	11.2	13.3	17.0	12.0	16.2	15.9	16.1	23.2	**9.03**
Classroom2	12.6	19.0	16.3	19.0	16.4	21.3	21.7	26.6	21.6	32.7	**11.9**
Classroom2E	**23.6**	29.7	28.4	28.0	31.2	44.1	37.5	48.7	39.6	50.7	**23.6**
Computer	11.5	18.3	16.1	20.1	13.0	11.7	16.3	22.6	16.8	40.5	**8.45**
Crusade	12.6	20.2	18.9	33.7	20.0	36.8	34.9	25.3	42.1	57.7	**9.89**
CrusadeP	7.29	15.7	14.3	31.2	21.0	32.1	30.4	17.0	37.5	55.7	**6.28**
Djembe	5.66	5.86	5.92	8.95	11.3	6.47	13.6	11.3	8.46	22.2	**3.94**
DjembeL	27.9	33.3	35.3	33.0	63.2	44.7	43.4	54.2	49.5	80.5	**23.4**
Hoops	**16.0**	25.8	25.9	31.1	33.4	41.5	31.6	35.4	39.6	59.0	16.3
Livingroom	16.5	20.8	21.6	23.0	19.9	23.6	29.5	30.8	25.6	40.0	**14.7**
Newkuba	13.7	17.5	19.9	18.0	20.9	18.6	26.4	31.5	22.6	36.4	**13.6**
Plants	14.2	19.3	20.0	18.9	42.0	22.2	26.0	24.9	28.6	67.9	**11.2**
Staircase	**17.7**	28.9	32.4	58.6	52.9	58.8	35.3	55.5	61.5	85.8	17.9
Avg (%)	13.6	18.0	18.0	22.5	23.5	23.6	25.9	26.9	27.0	43.4	**11.7**

TABLE 2. The disparity errors of stereo matching methods on 15 test stereo pairs from Middlebury dataset v3.

Data	DDL[39]	PSMNet_ROB[41]	IGF[20]	ISM[19]	DSGCA[40]	MPSV[43]	ADSM[42]	DoGGuided[45]	BSM[44]	DF[46]	FASW
Australia	9.87	7.98	7.48	6.49	11.0	13.7	**5.81**	12.3	23.4	17.1	7.42
AustraliaP	6.73	6.95	4.50	**4.46**	6.75	6.72	3.86	6.62	9.58	11.9	5.19
Bicycle2	7.17	5.07	4.97	4.37	7.01	6.38	8.17	11.2	9.29	10.0	**4.22**
Classroom2	8.25	3.70	10.5	11.5	13.7	8.61	15.8	16.3	26.7	27.2	**3.33**
Classroom2E	11.0	**7.38**	17.0	17.8	21.5	26.7	41.3	62.6	52.0	42.4	11.5
Computer	4.20	**3.31**	5.87	4.53	5.90	5.21	4.25	6.83	9.81	14.6	3.44
Crusade	4.34	4.13	5.36	7.64	6.72	9.07	28.2	34.0	21.6	29.1	**2.98**
CrusadeP	3.68	4.24	4.67	6.95	5.85	8.59	26.3	30.6	14.9	28.9	**2.86**
Djembe	1.96	2.08	2.08	2.47	2.78	3.68	2.15	3.65	6.31	5.90	**1.80**
DjembeL	9.50	14.7	10.3	13.2	22.2	34.6	24.1	37.0	40.5	33.5	**8.49**
Hoops	6.92	11.3	10.7	9.83	17.2	15.5	34.1	35.0	23.9	30.9	**6.25**
Livingroom	5.65	**4.35**	5.85	6.10	11.9	8.78	12.0	13.4	17.8	13.8	4.55
Newkuba	4.93	4.93	6.06	5.66	11.1	9.94	8.50	14.2	22.6	32.5	**4.81**
Plants	11.2	19.1	9.30	**6.78**	14.1	11.6	14.6	19.1	16.8	65.2	7.32
Staircase	9.27	9.87	12.4	23.6	23.8	24.6	25.7	34.4	48.0	52.9	**6.97**
Avg (px)	6.51	6.68	7.05	7.67	10.7	10.9	15.1	19.7	19.9	26.2	**4.86**

online evaluating their accuracy. Secondly, we compare the proposed stereo matching method with other state-of–the-art ASW methods on the Middlebury dataset v2 [38]. In this test, we use 27 stereo pairs from more various scenes. In addition, we also alone evaluate the cost aggregation performance of these different ASW methods since the ASW strategy is mainly applied to the cost aggregation step. Lastly, we also carry out the experiments on the KITTI benchmark to test the adaptability of our algorithm.

The experimental parameter settings are defined as follows. The parameters of GF for cost volume filtering are $\{\omega_k, \epsilon\} = \{7 \times 7, 0.0001\}$; the single parameter of MF for cost volume filtering is $\sigma = 0.05$. In order to make the results more convincing, all the parameters are kept constant for all data sets.

B. EVALUATION ON MIDDLEBURY DATASET V3

In order to online evaluate the performance of our whole stereo matching method, we carry out experiments on the newest Middlebury dataset v3 and upload their final disparity maps after post-processing to the Middlebury stereo evaluation website [28]. Here the quarter resolution image pairs in Middlebury dataset v3 are used. The error metric is the same as in [19] and [39], i.e., measuring the average disparity error and the error rate in non-occluded regions, where the error rate is the percentage of bad pixels as well as the bad pixels are the ones having disparity error more than 1 pixel. They respectively correspond to the metrics ''bad 4.0'' and ''avgerr'' with mask ''nonocc'' in the online Middlebury stereo evaluation website [28]. These quantitative evaluation results can be directly obtained from the Middlebury stereo evaluation website. The error rate and the average disparity error of all the 15 test stereo pairs are shown in Table 1 and 2 respectively. In order to avoid bias in image resolution, here we only compare with the current state-of-the-arts methods using quarter image resolution on the online platform, and these quantitative results are presented in Table 1 and 2 in descending order of overall performance. The visual comparisons with some top-performing methods here are shown in Fig.4.

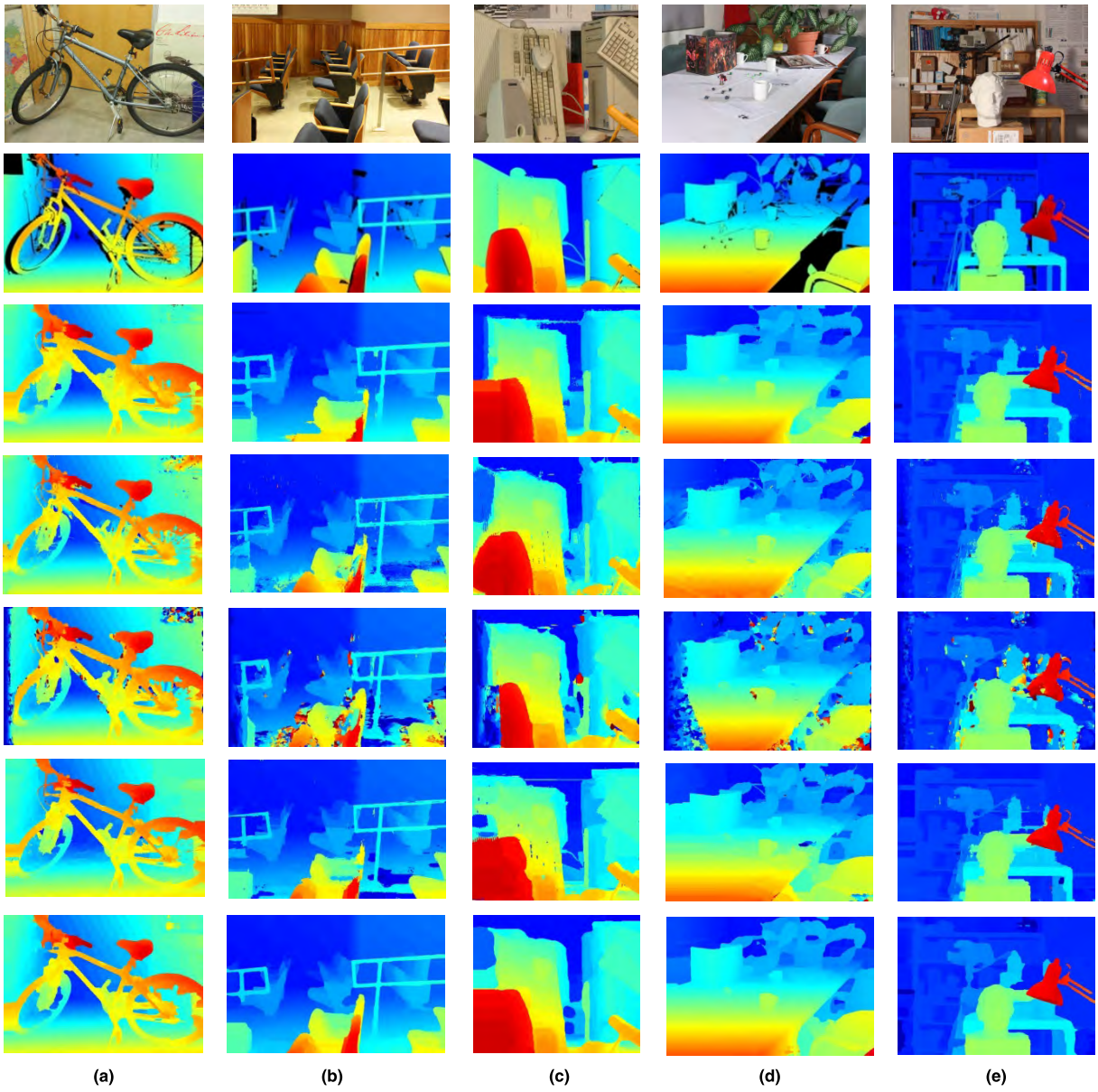


FIGURE 4. The disparity map results of Bicycle2, Classroom2, Computer, Crusade and Newkuba in the test set (from left to right). The 1st row: The reference images; the 2nd row: Ground truth; the 3rd row: Disparity maps computed with DDL; the 4th row: Disparity maps computed with IGF; the 5th row: Disparity maps computed with DSGCA; the 6th row: Disparity maps computed with ISM; the 7th row: Disparity maps computed the proposed method (i.e., FASW). Note that all disparity maps coded in false color can be directly obtained from the Middlebury stereo evaluation website [28]. (a) Bicycle2. (b) Classroom2. (c) Computer. (d) Crusade. (e) Newkuba.

As can be seen from Tables 1 and 2, no matter whether in terms of the error rate or the average disparity error, the proposed method ranks at the first place among all 11 methods. In comparison with second-ranked DDL [39], the average error rate and average disparity error of the proposed method respectively decrease by 1.9% and 1.65 px. Note that DDL performs stereo matching by learning a discriminative dictionary, and DSGCA [40], PSMNet_ROB [41] and DF [46] attempt to improve the performance of stereo matching by training deep convolutional neural networks. We can see that the disparity accuracy of DDL is higher than the other three learning methods. In addition, it can be seen that our algorithm also achieves better performance than ISM [19], and our average error rate as well as the average disparity error are respectively 10.8% and 2.81 px lower than ISM. Obviously, our method also outperforms the remaining four non-learning stereo methods such as DoGGuided [45] by a large margin. As illustrated in Fig.4, Bicycle2, Classroom2, Computer, Crusade and Newkuba in the test set of the Middlebury dataset v3 are used for visual comparison. We can find that the disparity maps produced with the proposed method are smoother than the other four state-of-the-art methods. Our

TABLE 3. The computational time of different methods.

Algorithm	Environments	Runtime (s)
PSMNet_ROB[41]	Nvidia GeForce GTX 1080 Ti /PCIe /SSE2 (CUDA, Python/PyTorch)	0.55
DSGCA[40]	i7-4770 @ 3.40 GHz; GTX 1080 GPU (Matlab)	11.0
ADSM[42]	8 i7 cores; Nvidia GTX460 SE (CUDA, C/C++)	35.8
DDL[39]	4 i7 cores @ 4.0 GHz (Matlab/C)	112
IGF[20]	1 i5 Core@3.2 GHZ (C++/OpenCV)	132
BSM[44]	Intel(R) Core(TM)2 Duo CPU P7370 @ 2.00GHz (C++/OpenCV)	244
ISM[19]	1 i5 core @3.2GHz(C/C++)	330
MPSV[43]	1 i5 core @2.7Ghz (Python)	594
DoGGuided[45]	2 i5 cores@3.0GHz (Matlab)	630
DF[46]	Matlab 2017	9999
FASW	Intel Core i5-6500@3.2GHZ (C++/OpenCV)	40.5

disparity maps not only contain less noise, but also better preserve the edges of objects. More quantitative comparison and visual comparison with more other state-of-the-art methods can be found from the Middlebury stereo evaluation website [28].

We briefly analyze the computational complexity of the proposed algorithm. Suppose that the window size of the census transform is $M \times M$. As mentioned above, N are the number of pixels in the guidance image and $|\mathcal{D}|$ is the number of disparity levels in $\mathcal{D}$. As such, the complexity of each step is characterized as: $O(NM^2|\mathcal{D}|)$ for the cost computation; $O(N|\mathcal{D}|)$ for the cost volume aggregation; $O(N|\mathcal{D}|)$ for disparity computation with WTA; $O(N|\mathcal{D}|)$ for disparity refinement. Thus, it can be found that the overall complexity of the proposed stereo matching algorithm is $O(NM^2|\mathcal{D}|)$ and the majority of the computational complexity comes from the cost computation with the census transform. Furthermore, the computational time of the above compared methods on different environments are listed in Table 3. It can be observed that our running environment is similar to ISM [19] and IGF [20], and neither parallelism or acceleration technique is utilized. However, our method is much faster than IGF and ISM. In addition, the learning methods such as PSMNet_ROB and DSGCA need to train models, and the pre-training is often much more time-consuming.

C. EVALUATION ON MIDDLEBURY DATASET V2

We compare the proposed method with four current state-of-the-art ASW methods on the Middlebury dataset v2, including AGF [17], MST [24], CSCF [21] and ISM [19], since they are very related to our algorithm and have good performance. For the four compared methods, their parameters follow the settings of the corresponding papers. In this experiment, we use 27 stereo pairs from various scenes, including four standard stereo pairs (Tsukuba, Venus, Teddy, Cones) of the Middlebury dataset v2 [38], to give a more reliable evaluation. The ground truth disparity maps of these

TABLE 4. The error rates of final disparity maps.

Data	AGF[17]	MST[24]	CSCA[21]	ISM[19]	FASW
Aloe	3.75	4.94	5.18	7.30	**3.27**
Art	8.63	10.41	8.83	14.74	**7.73**
Baby1	3.47	8.54	2.98	3.44	**1.67**
Baby2	3.23	15.49	**2.23**	4.56	2.24
Baby3	3.27	4.10	3.30	6.08	**2.56**
Books	8.05	9.60	7.55	10.60	**7.03**
Bowling1	11.34	20.80	9.30	9.84	**3.20**
Bowling2	4.88	11.06	4.74	6.73	**3.73**
Cloth1	0.28	0.48	0.84	0.37	**0.21**
Cloth2	2.26	3.97	2.96	4.67	**1.22**
Cloth3	1.37	1.91	1.90	2.95	**0.95**
Cloth4	1.04	1.23	1.50	2.38	**0.62**
Dolls	4.27	6.42	4.52	8.61	**3.66**
Flowerpots	9.71	15.26	8.28	10.31	**6.52**
Lampshade1	7.76	11.36	6.72	10.75	**2.85**
Lampshade2	16.37	10.71	16.36	12.20	**2.91**
Laundry	14.89	10.96	**10.31**	18.62	12.15
Moebius	8.68	7.97	8.29	10.52	**6.38**
Reindeer	5.26	8.57	4.28	7.29	**3.21**
Rocks1	2.32	2.70	2.71	3.03	**1.39**
Rocks2	**1.09**	2.07	1.35	2.54	1.12
Wood1	2.90	10.17	3.28	3.09	**1.78**
Wood2	0.39	1.47	**0.32**	1.60	0.46
Tsukuba	1.88	**1.52**	1.91	4.38	3.98
Venus	**0.16**	0.42	0.18	1.50	0.45
Teddy	6.59	6.34	6.04	10.21	**6.02**
Cones	3.41	3.22	**2.79**	6.79	2.91
Avg (%)	5.08	7.10	4.76	6.85	**3.34**

27 stereo pairs are provided by the Middlebury benchmark. In order to evaluate the overall matching performance of each method, we evaluate their final disparity map results by computing the error rate with error threshold 1 pixel in non-occluded regions as in [30]. The quantitative comparison results are shown in Table 4, while their visual comparison is presented in Fig.5. To better observe the disparity maps, the bad pixels are marked in red in Fig.5. As can be seen from Table 4, the matching error of the proposed method is 3.34%, which is lowest among all the five methods. Besides, our method is more accurate than the other four methods in most stereo pairs. The matching error of the proposed method is respectively decreased by 1.74%, 3.76%, 1.42% and 3.51% compared with AGF [17], MST [24], CSCA [21] and ISM [19]. The visual comparison in Fig.5 also shows that our method has better performance both in low texture regions and high texture regions. We also observe that MST can effectively cope with the low texture regions well due to its non-local aggregation ability, but it is powerless for high texture regions as shown in Fig.5. In addition, we can see that the performance of AGF is similar to that of CSCA, and both AGF and CSCA are more accurate than ISM. However, these three local ASW methods tend to yield more bad pixels in low texture areas compared to MST and our method.

Note that the ASW strategy is mainly applied to cost aggregation step. Therefore, it is necessary to evaluate the cost aggregation accuracy of each ASW method. Firstly, the initial

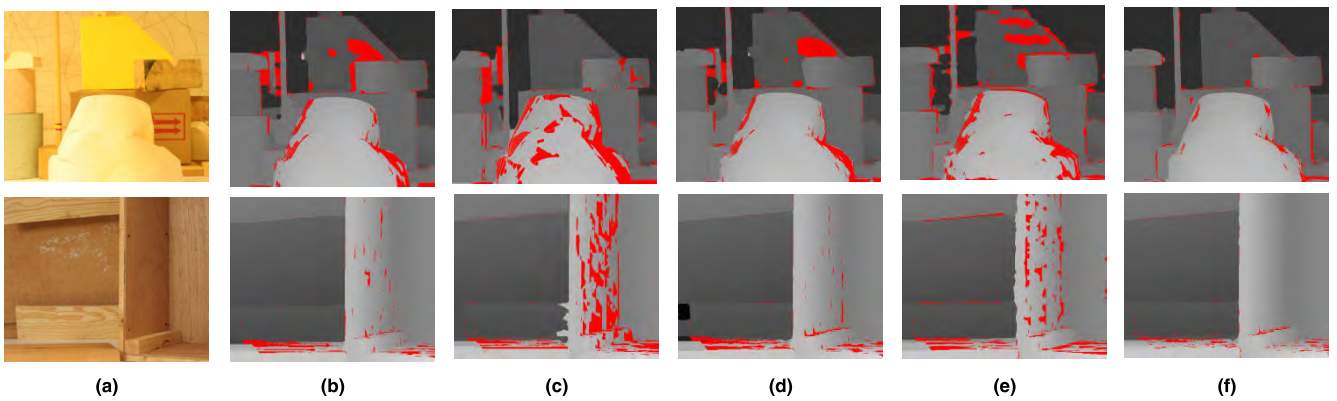


FIGURE 5. The final disparity maps of Lampshade1 and Wood1 in Middlebury dataset v2. The bad pixels in the disparity maps are marked red. (a) The reference images. (b) Disparity maps computed with AGF. (c) Disparity maps computed with MST. (d) Disparity maps computed with CSCA. (e) Disparity maps computed with ISM. (f) Disparity maps computed with the proposed method (i.e., FASW).

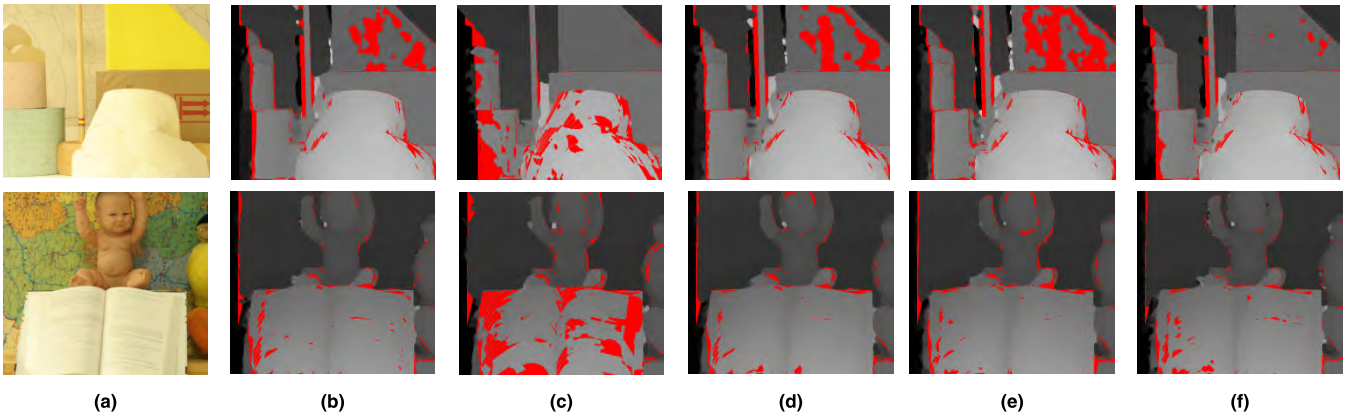


FIGURE 6. The raw disparity maps of Lampshade2 and Baby2 in Middlebury dataset v2. The bad pixels in the disparity maps are marked red. (a) The reference images. (b) Raw disparity maps computed with AGF. (c) Raw disparity maps computed with MST. (d) Raw disparity maps computed with CSCA. (e) Raw disparity maps computed with ISM. (f) Raw disparity maps computed with the proposed method (i.e., FASW).

cost volume is calculated by applying the census transform explained in Section III (A) and it is used as common input to all the ASW methods for fair comparison. Next, each ASW strategy is performed on the same initial cost volume and then its raw disparity map is directly established with the WTA optimization. Note that here no post-processing for refining the raw disparity maps is employed for more reliable evaluation. Accordingly, in order to evaluate the aggregation accuracy of different ASW methods, we compute the error rates with error threshold 1 pixel in non-occlusion regions of the raw disparity maps. Their quantitative evaluation results are presented in Table 5. For visual comparison, the raw disparity maps yielded by the above five ASW algorithms are shown in Fig.6.

Firstly, it can be seen in Table 5 that the aggregation error of the proposed method is the lowest among all the five methods. We can also find that the vast majority of the best results are obtained by our method. The aggregation error of the proposed method is respectively decreased by 0.99%, 3.46%, 1.21% and 2.19% compared with AGF, MST, CSCA and ISM. Note that ISM uses a cascade model of IGF and BF to smooth the cost volume. It can be found in the visual comparison of the raw disparity maps in Fig. 6 that ISM performs worse in low texture regions than AGF and CSCA since its cascade filter simply utilizes fixed-size window without spatial adaptivity. In comparison, in order to improve the performance of cost volume filtering, AGF remodels the weight kernel of GF by adaptively adjusting local kernel window, while CSCA enforces the inter-scale consistency on the multi-scale cost volume when performing cost volume filtering with GF. However, the support regions of these local ASW methods are still limited in local windows of user-specified size. Due to this reason, local aggregation methods are usually vulnerable to the lack of texture. On the other hand, we can see from Fig. 6 that the aggregation performance of MST shows better than AGF, CSCA and ISM for low texture regions. This is because MST performs non-local cost aggregation in the whole image as each pixel adaptively receives supports from all other pixels along a MST. However, it also shows that MST is less accurate than other four methods in the quantitative evaluation. This is because that MST removes some important connected edges, which leads to

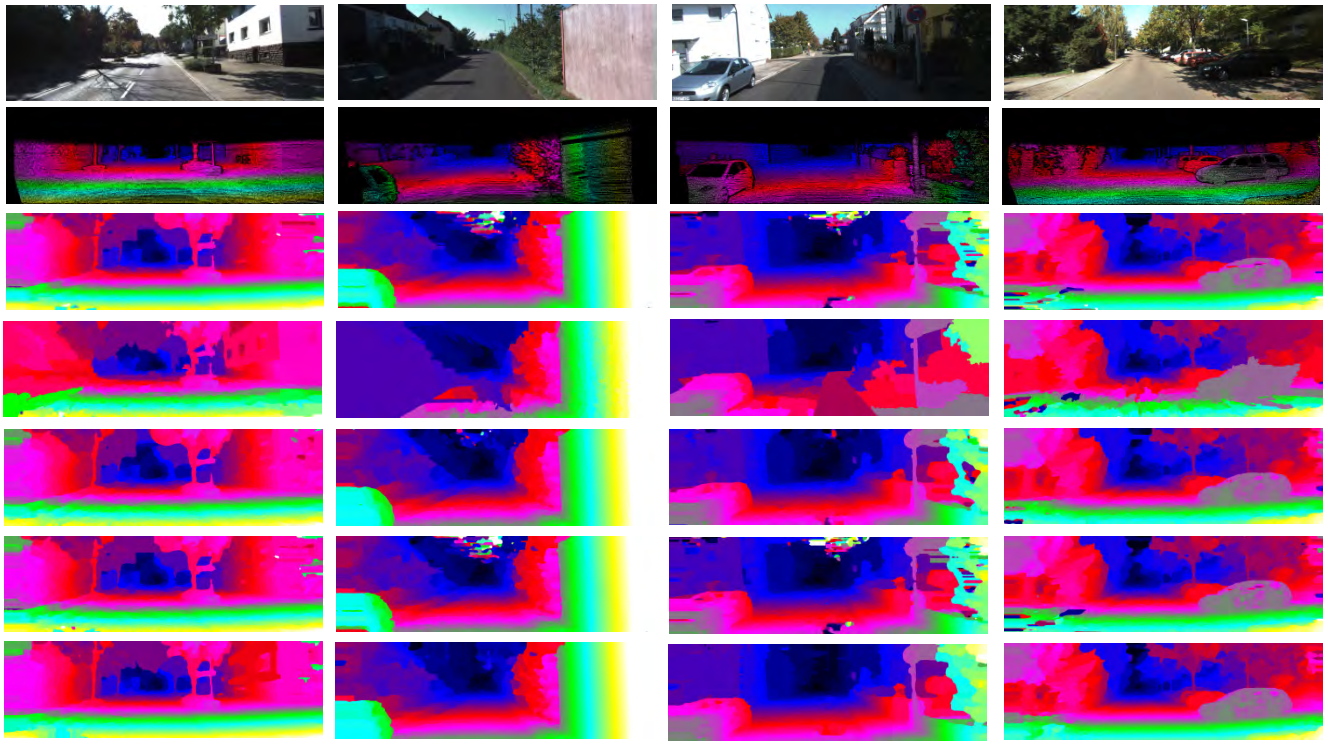


FIGURE 7. The final disparity map results of 000000_10, 000001_10, 000002_10, 000003_10 in the KITTI training set (from left to right). The 1st row: The reference images; the 2nd row: The ground truths; the 3rd row: Disparity maps computed with AGF; the 4th row: Disparity maps computed with MST; the 5th row: Disparity maps computed with CSCA; the 6th row: Disparity maps computed with ISM; the 7th row: Disparity maps computed with the proposed method (i.e., FASW). Note that the disparity maps are coded in false color by using the KITTI development kit [29].

the low discrimination for matching ambiguity especially in highly-textured regions. By adopting dual support windows, our aggregation method with FASW can effectively solve the defects of local methods and non-local methods. The visual comparison in Fig.6 shows that our aggregation method performs better in low texture regions because it performs the cost volume filtering with the non-local edge-aware filter over the whole image to establish the non-local optimized result for each pixel. On the other hand, the visual comparison in Fig.6 also illustrates that our aggregation method produces less bad pixels (i.e., the pixels with disparity error more than 1 pixel) in highly-textured regions since the primitive connectivity in the local window can be maintained when performing the cost aggregation with the local edge-aware filter. The above experiment results demonstrate that the proposed method outperforms the other state-of-the-art ASW methods in terms of the overall matching performance and the cost aggregation performance.

D. EVALUATION ON KITTI DATASET

In this section, we carry out the experiments on the KITTI benchmark [29] to further verify the adaptability of our method. The KITTI dataset [47] contains 195 test image pairs and 194 training image pairs for evaluating stereo matching algorithms. These image pairs from various real complex road scenes are taken by a pair of high-resolution cameras equipping on an autonomous driving platform. All the KITTI images are captured under the real-world illumination condition. Hence, most image pairs contain large low texture regions, e.g., sky, walls and cars, and inconsistent illumination conditions, e.g., shades and light reflection. Hence, the KITTI benchmark is challenging. We use the whole 194 training image pairs with ground truth disparity maps available to evaluate our method. In addition, the proposed method is still compared with the above four state-of-the-art methods, i.e., AGF [17], MST [24], CSCA [21], and ISM [19]. The accuracy of the final disparity maps is measured in terms of the average disparity error, as well as the error rate with default error threshold 3 pixels. For each stereo pair in the KITTI dataset, there are two different regions needed to be evaluated, that is, the whole reference image region denoted as ''All'' and the non-occluded region denoted as ''Noc''. Accordingly, each region has a ground truth of disparity map available. Table 6 reports the quantitative evaluation of the five ASW methods by computing the average error on the whole training data sets. Fig. 7 illustrates the visual comparison, where the false color disparity maps are displayed by using the KITTI development kit [29].

From Table 6, we can see that our proposed algorithm performs better than the other methods in terms of both average disparity error and average error rate. As can be seen in Fig.7, MST has worse performance in roads that contains a lot of high texture and noise compared to the other local methods. This makes sense, because the primitive local Markov random fields of the images are destroyed during generating the MST. However, the local ASW methods such

TABLE 5. The error rates of raw disparity maps.

Data	AGF[17]	MST[24]	CSCA[21]	ISM[19]	FASW
Aloe	5.17	6.19	6.51	6.94	**4.50**
Art	11.17	12.92	11.91	13.42	**10.05**
Baby1	3.01	7.37	3.23	3.19	**2.26**
Baby2	3.60	13.96	3.77	4.21	**3.51**
Baby3	4.31	7.85	4.63	4.77	**3.76**
Books	9.24	11.11	9.48	10.49	**8.13**
Bowling1	7.58	17.17	**5.71**	6.38	5.76
Bowling2	7.49	12.58	7.81	7.40	**5.29**
Cloth1	0.77	0.96	1.65	1.09	**0.66**
Cloth2	2.80	4.60	3.82	3.28	**2.15**
Cloth3	2.06	2.55	2.48	2.73	**1.68**
Cloth4	1.74	1.86	2.00	2.06	**1.31**
Dolls	5.51	7.10	6.68	7.71	**4.88**
Flowerpots	9.32	15.51	**8.87**	9.71	8.97
Lampshade1	9.85	10.96	9.91	14.80	**6.37**
Lampshade2	9.52	12.69	10.65	16.93	**6.42**
Laundry	18.75	17.84	**17.01**	20.83	17.13
Moebius	9.14	11.03	10.51	10.98	**8.63**
Reindeer	7.51	11.14	7.34	7.73	**5.62**
Rocks1	2.52	3.63	3.61	4.06	**2.22**
Rocks2	2.00	2.91	2.50	2.76	**1.75**
Wood1	5.16	10.53	4.55	4.95	**2.95**
Wood2	3.43	5.95	2.75	2.53	**2.42**
Tsukuba	4.09	4.35	**3.66**	5.79	4.03
Venus	1.71	1.95	1.77	2.14	**1.42**
Teddy	7.77	7.60	8.20	9.89	**7.41**
Cones	4.25	4.07	4.36	5.22	**3.59**
Avg (%)	5.91	8.38	6.13	7.11	**4.92**

TABLE 6. Quantitative evaluation on the KITTI training set.

Algorithm	Out-Noc[a]	Out-All[b]	Avg-Noc[c]	Avg-All[d]
AGF[17]	8.59%	9.73%	1.77 px	1.99 px
MST[24]	23.27%	24.41%	3.47 px	4.15 px
CSCA[21]	7.84%	8.96%	1.52 px	1.68 px
IGM[19]	8.88%	10.0%	1.87 px	2.07 px
FASW	**6.89%**	**8.12%**	**1.31 px**	**1.45 px**

[a] Out-Noc: the error rate in non-occluded areas;
[b] Out-All: the error rate in total;
[c] Avg-Noc: the average disparity error in non-occluded areas;
[d] Avg-All: the average disparity error in total.

AGF and ISM yield more erroneous disparity values in some large textureless regions, e.g., sky and cars because their local support windows cannot fit the whole low texture areas adaptively. In contrast, our method is adaptive to different texture regions. It can also be found in Fig.7 that the disparity maps generated by our method are more smoothing and contain less outliers.

E. SENSITIVITY OF PARAMETERS

The two key parameters used in our FASW strategy are the dimension r of GF kernel window and the parameter σ of the MST weighting function in (15). We use the training dataset in quarter resolution from Middlebury dataset v3 to study the performance of the proposed stereo matching method with respect to these two parameters. The average error rate of the training dataset is evaluated in this test. We first change the dimension r from 3 to 21 (the interval is 2) while keep all the other parameters settings constant. Fig.8 (a) shows the test results with different dimension r of GF kernel window. Similarly, we vary the parameter σ from 0.01 to 0.1 while fix the values of the other parameters. Fig.8 (b) depicts the effect of changing the parameter σ.

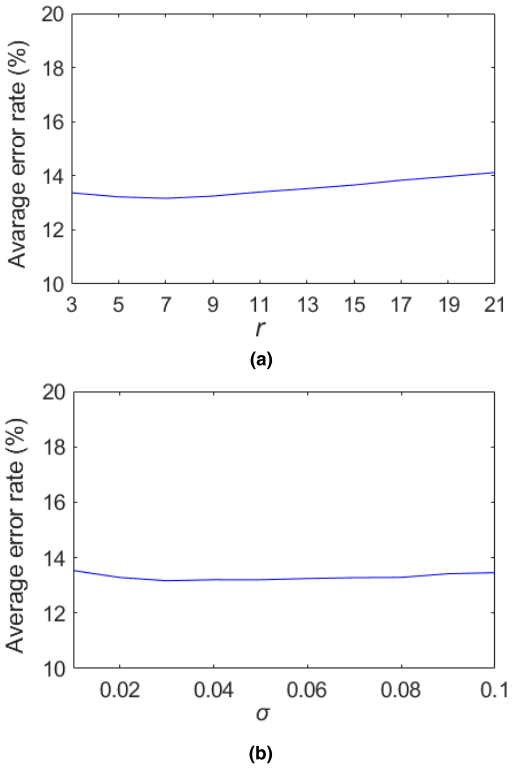
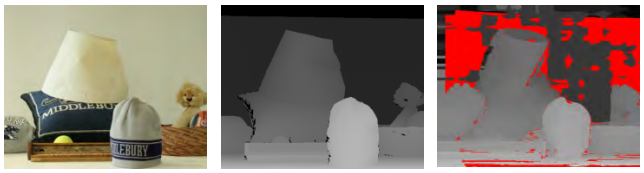


FIGURE 8. Performance of the proposed stereo algorithm with respect to the dimension r of GF kernel window and the parameter σ of the MST weight function. (a) Effect of the dimension r of the GF kernel window. (b) Effect of the parameter σ of the MST weight function.

Fig. 8(a) demonstrates that our method is insensitive to the change of the dimension r of GF kernel window, but the proposed stereo algorithm shows the best performance when $r = 7$. Similarly, it can be seen from Fig. 8(b) that the variance of the matching error is very low when σ ranges from 0.01 to 0.1, and the error is minimum when $\sigma = 0.05$.

F. DISCUSSION

A challenge problem of the ASW-based stereo methods is that the background is the main part of the scene image and the background itself is mainly occupied by large homogeneous region or repeated texture. In such case, it is very difficult to find accurate corresponding points for such background region as illustrated in Fig.9. On the one hand, in such background region, it is inherently ambiguous to consider the color difference or local shallow features such as the census transform as the measurement of pixel similarity. On the other hand, as described in [21], [27], the matching ambiguities in such ill-posed region also cannot be effectively reduced after performing cost aggregation because the aggregated cost curve of a pixel in such ill-posed region still contains



(a) (b) (c)

FIGURE 9. The final disparity map result of Midd1 in Middlebury dataset v2. (a) The reference image. Its background has large homogeneous region. (b) The ground truth of the disparity map. (c) The final disparity map computed with the proposed method (i.e., FASW).The bad pixels in the disparity map is marked red.

multiple local minima or flat valley around the true minimum. However, the matching performance for such ill-posed region may be improved by learning more robust and deep discriminative features with deep convolutional neural networks as the matching cost metric.

V. CONCLUSIONS

This paper describes a novel, yet powerful fusing ASW framework for stereo matching. By combining a local edge-aware filter and a non-local edge-aware filter to collaboratively smooth the cost volume, this proposed stereo matching algorithm can effectively overcome the disadvantages of the local ASW methods and the non-local ASW methods. The quantitative evaluation on, no matter whether the Middlebury benchmark of the indoor scene or the KITTI benchmark of the outdoor scene, demonstrates that the proposed method outperforms the state-of-the-art ASW methods. In the future work, we will consider more outstanding local or non-local edge-aware filter and integrate it in our fusing ASW framework. In addition, we would like to apply the proposed fusing ASW framework to other computer vision tasks.

REFERENCES

[1] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Int. J. Comput. Vis.*, vol. 47, nos. 1–3, pp. 7–42, Apr. 2002.

[2] T. Taniai, Y. Matsushita, Y. Sato, and T. Naemura, "Continuous 3D label stereo matching using local expansion moves," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 11, pp. 2725–2739, Nov. 2018.

[3] M. G. Mozerov and J. V. D. Weijer, "Accurate stereo matching by two-step energy minimization," *IEEE Trans. Image Process.*, vol. 24, no. 3, pp. 1153–1163, Mar. 2015.

[4] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 328–341, Feb. 2008.

[5] S. A. Adhyapak, N. Kehtarnavaz, and M. Nadin, "Stereo matching via selective multiple windows," *J. Electron. Imag.*, vol. 16, no. 1, Jan. 2007, Art. no. 013012.

[6] H. Hirschmüller, P. R. Innocent, and J. Garibaldi, "Real-time correlation-based stereo vision with reduced border errors," *Int. J. Comput. Vis.*, vol. 47, no. 1, pp. 229–246, 2002.

[7] O. Veksler, "Fast variable window for stereo correspondence using integral images," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2003, pp. 556–561.

[8] M. Gerrits and P. Bekaert, "Local stereo matching with segmentation-based outlier rejection," in *Proc. 3rd Can. Conf. Comput. Robot Vis.*, Jun. 2006, p. 66.

[9] K.-J. Yoon and I. S. Kweon, "Adaptive support-weight approach for correspondence search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 650–656, Apr. 2006.

[10] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. 6th IEEE Int. Conf. Comput. Vis.*, Jan. 1998, pp. 839–846.

[11] F. Tombari, S. Mattoccia, and L. D. Stefano, "Segmentation-based adaptive support for accurate stereo correspondence," in *Proc. Pacific–Rim Symp. Image Video Technol.*, Dec. 2007, pp. 427–438.

[12] A. Hosni, M. Bleyer, M. Gelautz, and C. Rhemann, "Local stereo matching using geodesic support weights," in *Proc. 16th IEEE Int. Conf. Image Process. (ICIP)*, Nov. 2009, pp. 2093–2096.

[13] C. Richardt, D. Orr, I. Davies, A. Criminisi, and N. A. Dodgson, "Real-time spatiotemporal stereo matching using the dual-cross-bilateral grid," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2010, pp. 510–523.

[14] Q. Yang, "Hardware-efficient bilateral filtering for stereo matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 5, pp. 1026–1032, May 2014.

[15] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 6, pp. 1397–1409, Jun. 2013.

[16] A. Hosni, C. Rhemann, M. Bleyer, C. Rother, and M. Gelautz, "Fast cost-volume filtering for visual correspondence and beyond," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 2, pp. 504–511, Feb. 2013.

[17] Q. Yang, P. Ji, D. Li, S. Yao, and M. Zhang, "Fast stereo matching using adaptive guided filtering," *Image Vis. Comput.*, vol. 32, no. 3, pp. 202–211, 2014.

[18] J. Lu, K. Shi, D. Min, L. Lin, and M. N. Do, "Cross-based local multipoint filtering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 430–437.

[19] R. A. Hamzah, A. F. Kadmin, M. S. Hamid, S. F. A. Ghani, and H. Ibrahim, "Improvement of stereo matching algorithm for 3D surface reconstruction," *Signal Process., Image Commun.*, vol. 65, pp. 165–172, Jul. 2018.

[20] R. A. Hamzah, H. Ibrahim, and A. H. A. Hassan, "Stereo matching algorithm based on per pixel difference adjustment, iterative guided filter and graph segmentation," *J. Vis. Commun. Image Represent.*, vol. 42, pp. 145–160, Jan. 2017.

[21] K. Zhang, Y. Fang, D. Min, L. Sun, S. Yang, and S. Yan, "Cross-scale cost aggregation for stereo matching," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 5, pp. 965–976, May 2017.

[22] C. Cigla, "Recursive edge-aware filters for stereo matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2015, pp. 27–34.

[23] C. C. Pham and J. W. Jeon, "Domain transformation-based efficient cost aggregation for local stereo matching," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 7, pp. 1119–1130, Jul. 2013.

[24] Q. Yang, "Stereo matching using tree filtering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 4, pp. 834–846, Apr. 2015.

[25] Q. Yang, "A non-local cost aggregation method for stereo matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1402–1409.

[26] X. Mei, X. Sun, W. Dong, H. Wang, and X. Zhang, "Segment-tree based cost aggregation for stereo matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 313–320.

[27] F. Cheng, H. Zhang, M. Sun, and D. Yuan, "Cross-trees, edge and superpixel priors-based cost aggregation for stereo matching," *Pattern Recognit.*, vol. 48, no. 7, pp. 2269–2278, Jul. 2015.

[28] D. Scharstein and R. Szeliski. *Middlebury Stereo Evaluation Version 3*. Accessed: Jan. 2019. [Online]. Available: http://vision.middlebury.edu/stereo/eval3/

[29] A. Geiger, P. Lenz, and R. Urtasun. *The KITTI Vision Benchmark Suite*. Accessed: Jan. 2019. [Online]. Available: http://www.cvlibs.net/datasets/kitti/eval_stereo_flow.php?benchmark=stereo,

[30] A. Hosni, M. Bleyer, and M. Gelautz, "Secrets of adaptive support weight techniques for local stereo matching," *Comput. Vis. Image Understand.*, vol. 117, no. 6, pp. 620–632, Jun. 2013.

[31] L. De-Maeztu, S. Mattoccia, A. Villanueva, and R. Cabeza, "Linear stereo matching," in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 1708–1715.

[32] K. Zhang, J. Lu, and G. Lafruit, "Cross-based local stereo matching using orthogonal integral images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 7, pp. 1073–1079, Jul. 2009.

[33] L. Bao, Y. Song, Q. Yang, H. Yuan, and G. Wang, "Tree filtering: Efficient structure-preserving smoothing with a minimum spanning tree," *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 555–569, Feb. 2014.

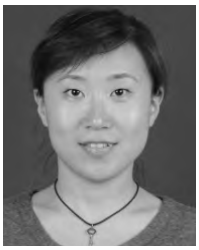
[34] L. Dai, M. Yuan, F. Zhang, and X. Zhang, "Fully connected guided image filtering," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 352–360.

[35] R. Zabih and J. Woodfill, "Non-parametric local transforms for computing visual correspondence," in *Proc. Eur. Conf. Comput. Vis.*, Jun. 1994, pp. 151–158.
[36] H. Hirschmuller and D. Scharstein, "Evaluation of stereo matching costs on images with radiometric differences," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 9, pp. 1582–1599, Jul. 2009.
[37] D. H. Scharstein *et al.*, "High-resolution stereo datasets with subpixel-accurate ground truth," in *Proc. German Conf. Pattern Recognit.*, Oct. 2014, pp. 31–42.
[38] H. Hirschmuller and D. Scharstein, "Evaluation of cost functions for stereo matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.
[39] J. Yin, H. Zhu, D. Yuan, and T. Xue, "Sparse representation over discriminative dictionary for stereo matching," *Pattern Recognit.*, vol. 71, pp. 278–289, Nov. 2017.
[40] I. K. Park, "Deep self-guided cost aggregation for stereo matching," *Pattern Recognit. Lett.*, vol. 112, pp. 168–175, Sep. 2018.
[41] J.-R. Chang and Y.-S. Chen, "Pyramid stereo matching network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5410–5418.
[42] N. Ma, Y. Men, C. Men, and X. Li, "Accurate dense stereo matching based on image segmentation using an adaptive multi-cost approach," *Symmetry*, vol. 8, no. 12, p. 159, 2016.
[43] J.-C. Bricola, M. Bilodeau, and S. Beucher. (2016). *Morphological Processing of Stereoscopic Image Superimpositions for Disparity Map Estimation*. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01330139/
[44] K. Zhang, J. Li, Y. Li, W. Hu, L. Sun, and S. Yang, "Binary stereo matching," in *Proc. 21st Int. Conf. Pattern Recognit.*, Nov. 2012, pp. 356–359.
[45] M. Kitagawa, I. Shimizu, and R. Sara, "High accuracy local stereo matching using DoG scale map," in *Proc. 15th Int. Conf. Mach. Vis. Appl. IAPR*, May 2017, pp. 258–261.
[46] W. Mao and M. Gong, "Disparity filtering with 3D convolutional neural networks," in *Proc. 15th Conf. Comput. Robot Vis. (CRV)*, May 2018, pp. 246–253.
[47] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361.

WENHUAN WU received the M.S. degree from Nanchang Hangkong University, Nanchang, China, in 2009. He is currently pursuing the Ph.D. degree with the School of Automation and Information Engineering, Xi'an University of Technology, Xi'an, China. He is currently a Lecturer with the Hubei University of Automotive Technology, Shiyan, China. His research interests include computer vision, pattern recognition, and image processing.

HONG ZHU received the Ph.D. degree from Fukui University, Fukui, Japan, in 1999. She is currently a Professor with the School of Automation and Information Engineering, Xi'an University of Technology, Xi'an, China. Her research interests include image analysis, intelligent video surveillance, and pattern recognition.

SHUNYUAN YU received the Ph.D. degree from the School of Automation and Information Engineering, Xi'an University of Technology, Xi'an, China, in 2017. She is currently with the School of Electronic and Information Engineering, Ankang University, Ankang, China. Her research interests include pattern recognition and digital images processing.



JING SHI received the M.S. degree from the School of Automation and Information Engineering, Xi'an University of Technology, Xi'an, China, in 2009, where she is currently pursuing the Ph.D. degree with the School of Automation and Information Engineering. Her research interests include pattern recognition, scene classification, and digital images processing.