Vol.65: e2210409, 2022
https://doi.org/10.1590/1678-4324-202210409
ISSN 1678-4324 Online Edition

Article - Engineering, Technology and Techniques

Local PatchMatch Based on Superpixel Cut for Efficient High-resolution Stereo Matching

Xianjing Cheng[1,2]
https://orcid.org/0000-0002-1644-5384

Zhijun Hu[1]
https://orcid.org/0000-0002-9965-4594

Yong Zhao[1,3]
https://orcid.org/0000-0002-7999-1083

Xiaomin Yu[1,5]
https://orcid.org/0000-0002-7123-5953

Raja Soosaimarian Peter Raj[4]
https://orcid.org/0000-0002-7216-2207

Wenbang Yang[1,6]
https://orcid.org/0000-0003-0436-7064

[1]Guizhou University, College of Computer Science and Technology, Guiyang, Guizhou, China; [2]Zunyi Normal University, Qianbei Information Technology Research Institute, Zunyi, Guizhou, China; [3]Peking University Shenzhen Graduate School, School of Electronic and Computer Engineering, Shenzhen, Guangdong, China; [4]Vellore Institute of Technology, School of Computer Science and Engineering, Vellore-632014, Tamilnadu, India; [5]GuiZhou University of Finance and Economics, School of Information of GUFE, Guiyang, Zuizhou, China; [6]Xingyi Normal University for Nationalities, School of Mathematics, Xingyi, Guizhou, China.

Editor-in-Chief: Alexandre Rasi Aoki
Associate Editor: Alexandre Rasi Aoki

Received: 21-Jun-2021; Accepted: 24-Nov-2021.

*Correspondence: chengxianjing2014@126.com; Tel.: +86-13765958535 (X.C.).

HIGHLIGHTS

- A new method of extracting the feature points in superpixels is proposed to compute the candidate disparities using non-local cost aggregation.
- A novel two-stage optimization framework for superpixel proposals by first getting the labels of feature points quickly and then consists the candidate label sets for updating the labels of pixels within the corresponding superpixel instead of assigning a label randomly to those pixels.
- The weight combination of intensity, gradient and binary image is designed for constructing an optimal minimum spanning tree to compute the aggregated matching cost.

Abstract: Obtaining the accurate disparity of each pixel quickly is the goal of stereo matching, but it is very difficult for the 3D labels-based methods due to huge search space of 3D labels, especially for highresolution images. We present an novel two-stage optimization strategy to get the accurate disparity map for high-resolution stereo image efficiently, which includes feature points optimization and superpixel optimization. In the first stage, we construct the support points including edge points and robust points for triangulation, which is used to extract feature points and then perform spatial propagation and random refinement to get the candidate 3D label sets. In the stage of superpixel optimization, we update per pixel labels of the corresponding superpixels using the candidate label sets, and then perform spatial propagation and random refinement. In order to provide more prior information to identify weak texture and textureless areas, we design the weight combination of “intensity + gradient + binary image” for constructing an optimal minimum spanning tree (MST) to compute the aggregated matching cost, and obtain the labels of minimum aggregated

matching cost. We also design local patch surrounding the corresponding superpixel to accelerate our algorithm in parallel. The experimental result shows that our method achieves a good trade-off between running time and accuracy, including KITTI and Middlebury benchmark, which are the standard benchmarks for testing the stereo matching methods.

Keywords: feature points; superpixel optimization; prior information; weight combination; stereo matching.

INTRODUCTION

The Depth estimation is the goal of stereo matching, which estimates per-pixel disparity while inputting a stereo image pair, which is widely applied in navigation [1], 3D construction [2, 3], autonomous driving [4], robotics localization [5] and so on. Stereo matching approaches generally take four steps [6]: matching cost computation, cost aggregation, disparity computation and disparity refinement.

According to Scharstein and Szeliski [6], stereo matching algorithms can be classified into local approaches and global approaches. Local approaches compute the cost value of corresponding points among images, which lacks the correlation of neighboring pixels and global information, it leads to the inaccurate disparities. However, global approaches based on Markov Random Field (MRF) integrate the correlation of neighboring pixels, but the energy function optimization of global approaches is NP-hard, which can be solved by some approximate approaches, such as fusion methods [7], i.e, belief propagation (BP) [8, 9], graph cut (GC) [10, 11]. BP takes global energy function values as messages and propagates the neighboring pixels by the sequence mode. GC takes pixel of minimize energy values as a anchor point and performs $\propto$-expansion if satisfying the submodular property in the surrounding area [12]. According to the classfication of the representation of pixel labels, stereo matching approaches can be classified into 1D discrete and 3D label continuous approaches. The discrete approaches, e.g, Loopy BP [13], TRW [14], SGM [15], MST [16], exist the stair-artifacts effect. Hence, PatchMatch stereo [17] proposes 3D label continuous method to represent per-pixel disparity, which overcomes the stair-artifacts effect of discrete approaches and gets a smooth disparity map. However,the huge and infinite continuous label space of 3D labels increases computation and gets the accurate labels of pixels quickly is very difficulty, especially for high-resolution images.In this paper, we propose an efficient method using 3D labels based on superpixel cut, which first gets superpixels by SLIC, and then choose edge and robust pixels of each superpixel as support points for triangulation. After triangulating support points, we get some feature points and compute the 3D labels of feature points by MST. While obtaining the candidate label sets that is composed of the feature points, we update the labels of pixels in corresponding superpixel, followed by spatial propagation and random refinement. Our contributions are mainly followed:

(1) we propose a new method extracting feature points of superpixel by triangulation.

(2) we present an novel two-stage optimization framework for superpixel proposals by first getting the labels of feature points quickly and then consists the candidate label sets for updating the labels of pixels within the corresponding superpixel instead of assigning a label randomly to those pixels.

(3) we design the weight combination of intensity, gradient and binary image for constructing an optimal minimum spanning tree (MST) to compute aggregated matching cost, not like MST only using intensity, and design the local patch structure surrounding the corresponding superpixel for accelerating our algorithm in parallel.

(4) the experiment shows the well performance of our method and gets a good trade-off between running time and accuracy.

Related work

Assigning each pixel p a 3D label $\boldsymbol{f_p} = (\boldsymbol{a_p}, \boldsymbol{b}, \boldsymbol{c_p})$, the disparity $\boldsymbol{d_p}$ is uniquely determined by a local disparity plane:

$$\boldsymbol{d_p} = \boldsymbol{a_p u} + \boldsymbol{b_p v} + \boldsymbol{c_p} \tag{1}$$

Where $\boldsymbol{u}$,$\boldsymbol{v}$. is coordinates in the image domain. Obtaining the accurate 3D labels of pixels quickly is very difficult due to the huge and infinity 3D label search space. One of the common solutions is PatchMatch method, PatchMatch Stereo [17] proposes PatchMatch-based 3D label continuous approach, which performs spatial propagation, view propagation and random refinement sequentially to get the disparity map, but it is a local methods and time-consuming. PMF [18] proposes superpixel-based PatchMatch filter method and constructs a superpixel neighborhood structure for spatial propagation, but it is still a local method. PMBP

[19] uses belief propagation message delivery mechanism within patch, which is the sequential mode that is time-consuming. However, some methods integrate segmentation information, e.g, superpixel structure [20,21]. PMSC [22] utilizes SLIC [23] to construct a multilayer superpixel structure, and then generates a series of label proposals, and iteratively updates current best labels by performing 훼-expansion with each proposals. LocalExp [24] performs three different sizes patches iteratively in three layers, which chooses a center-region label and propagates it to the expansion regions for updating the labels of expansion-region pixels, the drawback is that many shared regions for filtering and leads to huge redundant computations. Otherwise, SGM [15] is a semi-global algorithm based on dynamic programming, which takes discrete label and sequence sweep mode individually. HPM-TDP [26] uses the cross-scale framework [27] based on spuerpixel structure and constructs the neighbor structure according to PMF, and uses MST to speed up the algorithm, but the cost volume uses gradient and census information computed on the low-scale layer is not robust, which limits the quality of disparity map. However, there are some methods using triangulation technology, MeshStereo [28] formulates the objective as a twolayer MRF, which are glued together by an alignment energy. ELAS [29] gets the prior by performing triangulation on a set of support points and LS-ELAS [30] adds line segmentation of edges on the basis of ELAS. They both lack global information and the prior information is not robust. Fickel and coauthors [31] construct an vertices set including the uniformly edgepoints by canny detector and robust points by Gaussian kernel filtering, and then perform triangulation, which uses the matching function based on the histogram of pixel gains and cost aggregation using intensity and spatial distance information. Mozerov and van [32] use a two-stage energy minimization to perform the global energy minimization in the full connected model, and then use the unary potential of locally connected MRF model for optimization, which actually is the cost filtering technique. Local plane sweep method (LPS) [33] uses a two-stage optimization strategy that first proposes local slanted plane sweeps to obtain disparity hypotheses for a semi-global matching algorithm, which extracts sparse feature followed by an iterative clustering step to form local plane hypotheses, and then uses semi-global matching to implement global optimization. MST [16] is a no-local cost aggregation approach, which constructs a minimum spanning tree using the neighboring pixels and utilizes a two-pass cost aggregate strategy to get the disparity map quickly, but it uses 1D discrete label. ST [34] integrates segmentation information on the basis of MST. 3DMST [35] uses MC-CNN [36] as the matching cost, whose MC-CNN is better trained than us, and constructs some separate tree structure for spatial propagation and random refinement, which is different from our method, our method uses the sample MC-CNN followed by LocalExp [24] and extracts feature points of each superpixel by triangulation, which can quickly shorten the range of selecting accurate 3D labels for each pixel. 3DMST-CM [37] adds the confidence optimization module based on 3DMST and gets effective improvement in some challenging scenarios.

In the deep learning techniques, cascade stereo [37], AANet [38], PSMNet [39] and GANet [40] use multi-sacle structure and endocer-decoder convolution layers to get the disparity maps. CSPN [41] uses the spatial information to get the accurate disparities by the 3D-CSPN module. Deeppruner [42] adpots the differentiable patchmatch to compute the confidence of disparity in patches. AdaStereo [43] realizes a more standard, complete and effective domain adaptation pipeline. But they need a lot of training images and effective training strategy.

PROPOSED METHOD

Given a rectified stereo image pair, our aim is to estimate accurate disparity map. Our proposed method first utilizes triangulation to obtain robust feature points described in section A, the matching cost aggregation process is defined in section B, our optimization procedure is presented in section C and we discuss fast implementation in section D.

A Triangular Tessellation

We found that the similar geometric properties and spatial proximity of pixels share the same disparity or the range of they disparities is small. Hence, the image is segmented into superpixels using SLIC [23] according to the color and spatial distance as Figure 4, and the range of disparities of pixels in the same superpixel is small. To process superpixels efficiently, we use triangulation to generate some irregular local planes in the superpixel. Assume a feature point represents a local plane, we try to extract some robust feature points and calculate the candidate disparities instead of computing a lot of disparities for each pixel with 3D label. We can shorten the candidate disparities range quickly.

First, we construct the support points for triangulation. The support points include two parts, i.e, one is the edge points, which can make full use of edge information of superpixels and another is the robust points. According to the average number of pixels of a superpixel, for example, a image is split as 600 superpixels, the average number of each superpixel is 36x36-size pixels, we take the sqrt of the 36x36 size

pixels, i.e, 6 pixels, we get the set of edge points by sampling every 6 pixels uniformly. Then, we use Gausian pyramid downsampling by the rate 1/2 to get half-resolution image, and use bicubic interpolation to restore the original scale image. Next, we extract pixels with same intensity value as the original image, called the robust points. The support points are consisted of the edge points and the robust points.

After finishing the extraction of feature points, we triangulate the support points by Delaunay triangulation and get many irregular triangular plane, then we extract the centroid point of each triangular plane, i.e, the centroid point is feature point. An illustration of the process of extracting feature points is shown in Figure 1. Through the above process, we get the number of feature points that is far less than the number of pixels of the corresponding superpixel. At last, we construct minimum spanning tree using the feature points, as described in Figure 3.

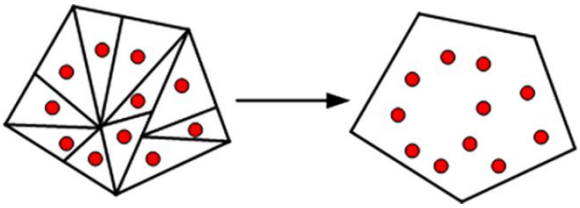

Figure 1. Extraction of feature points.

B Cost aggregation within superpixel

Yang [35] proposes an non-local cost aggregation method, while constructing minimum spanning tree (MST), which uses a two-pass matching cost aggregation optimization strategy including leaf nodes to root node and root node to leaf nodes. we follow their optimization strategy in this paper, but we adopt the 3D labels representing the disparities of pixels.

The reference image is regarded as a undirected grid graph, each pixel is a node, the neighboring pixels are connected with an edge, the weight of the connected pixel p and q is defined as:

$$\omega(\mathrm{p},\mathrm{q}) = \mathrm{w}(\mathrm{q},\mathrm{p}) = ||\mathrm{I}(\mathrm{p}) - \mathrm{I}(\mathrm{q})||_1 \tag{2}$$

Where $||\mathrm{I}(\mathrm{p}) - \mathrm{I}(\mathrm{q})||_1$ is the L1 color distance between the pixel p and q.

However, the gradient information can enhance the interaction between nodes, and binary image is used to identify the object boundaries as depicted in Figure 2. Hence, our weight is defined as followed.

$$\omega(\mathrm{p},\mathrm{q}) = \mathrm{w}(\mathrm{q},\mathrm{p}) = |\mathrm{I}(\mathrm{p}) - \mathrm{I}(\mathrm{q})| + |\Delta\mathrm{I}(\mathrm{p}) - \Delta\mathrm{I}(\mathrm{q})| + |\mathrm{B}(\mathrm{p}) - \mathrm{B}(\mathrm{q})| \tag{3}$$

Where $\Delta\mathrm{I}(\mathrm{p})$ includes the x-coordinates and y-coordinates gradient of p, the define as follows:

$$|\Delta\mathrm{I}(\mathrm{p}) - \Delta\mathrm{I}(\mathrm{q})| = 0.5 * |\Delta_{\mathrm{x}}\mathrm{I}(\mathrm{p}) - \Delta_{\mathrm{x}}\mathrm{I}(\mathrm{q})| + 0.5 * |\Delta_{\mathrm{y}}\mathrm{I}(\mathrm{p}) - \Delta_{\mathrm{y}}\mathrm{I}(\mathrm{q})| \tag{4}$$

For the binary image B.p/, we utilizes the adaptive mean threshold method to get the initial binary image, and then invert the value of the binary image, i.e, the value greater than the adaptive threshold is 0, and the value less than the threshold is 1. The purpose by this is to identify the regions where the intensity and gradient are difficult to calculate, and provides some effective prior to construct an optimal tree by using the combination of "intensity + gradient + binary image" weight.

(a) Gradient image (b) Binary image

Figure 2. Visualization of gradient and binary images of “Teddy” in training set of Middlebury V3 dataset, (a) gradient image, (b) binary image.

The similarity between node p and q in tree is defined as [40]:

$$\mathrm{S}(\mathrm{p},\mathrm{q}) = \mathrm{S}(\mathrm{q},\mathrm{p}) = \exp(\frac{\mathrm{D}(\mathrm{p},\mathrm{q})}{\gamma}) \tag{5}$$

Here D(p,q) is the the shortest path of node p and q in tree, which is the sum of weights of the connected edges, γ is a user-defined parameter.

We perform the MST-based matching cost aggregation strategy using 3D labels. Tence, the matching cost aggregation with 3D label f is computed as:

$$C^A(p,f) = \sum_q S(p,q)C(q,f) \tag{6}$$

The cost aggregation of pixel p calculates all connect node in the tree. However, the robustness of matching cost C(q,f) is very important for stereo matching algorithm. Hirschmuller et al [25] describe the robustness of different matching cost function under the illuminations situations and found census transform and the combination of color and gradient have well robustness, but recent work found the matching cost computed by convolution neural network (CNN) [45, 47] has high robustness than the above matching cost functions. Hence, we use pretrained CNN as our matching cost, which is defined as follows.

$$\mathrm{C}(\mathrm{q},\mathrm{f}) = \mathrm{C}_{\mathrm{CNN}}(\mathrm{I}^{\mathrm{L}}(\mathrm{q}_{\mathrm{x}},\mathrm{q}_{\mathrm{y}}),\mathrm{I}^{\mathrm{R}}(\mathrm{q}_{\mathrm{x}}-\mathrm{d}_{\mathrm{f}},\mathrm{q}_{\mathrm{y}})) \tag{7}$$

Where d_{f} is the disparity of pixel q with label f, CCNN is the matching cost under the 11x11 image patch centered at pixel q in the left image IL and the corresponding image patch in the right image IR.

A brute force implementation of the aggregated cost of all nodes has an extremely high complexity. For fast implementation, we use a two-pass optimization strategy followed by Yang [35]. In the first pass, the tree is traced from leaf nodes to root node. The aggregated cost of a node p is accumulated by the aggregated cost of all its child nodes

$$\mathrm{C}^{\mathrm{A}\uparrow}(\mathrm{p},\mathrm{f}) = \mathrm{C}(\mathrm{p},\mathrm{f}) + \sum_{\mathrm{q}\in\mathrm{Ch}(\mathrm{p})} \mathrm{S}(\mathrm{p},\mathrm{q})\mathrm{C}^{\mathrm{A}\uparrow}(\mathrm{q},\mathrm{f}) \tag{8}$$

Where Ch(p) is the set of all children of pixel p, $\mathrm{C}^{\mathrm{A}\uparrow}(\mathrm{p},\mathrm{f})$is the intermediate aggregated costs. For the leaf nodes,$\mathrm{C}^{\mathrm{A}\uparrow}(\mathrm{p},\mathrm{f})$is set to $\mathrm{C}^{\mathrm{A}}(\mathrm{p},\mathrm{f})$ during initialization, the root node calculates the weighted cost from their child nodes, while the rest of nodes calculate the weighted cost from their subtrees as depicted in Figure 3 (c). In the second pass, we trace the tree from root node to leaf nodes as Figure 3 (d). The final aggregated cost of node p is computed by its parent P a.p/ as followed:

$$\begin{aligned}\mathrm{C}^{\mathrm{A}\uparrow}(\mathrm{p},\mathrm{f}) &= \mathrm{S}(\mathrm{Pa}(\mathrm{p}),\mathrm{p})\mathrm{C}^{\mathrm{A}}(\mathrm{Pa}(\mathrm{p}),\mathrm{f}) + [1-\mathrm{S}^2(\mathrm{Pa}(\mathrm{p}),\mathrm{p})]\mathrm{C}^{\mathrm{A}\uparrow}(\mathrm{p},\mathrm{f}) \\ &= \mathrm{C}^{\mathrm{A}\uparrow}(\mathrm{p},\mathrm{f}) + \mathrm{S}(\mathrm{Pa}(\mathrm{p}),\mathrm{p})\mathrm{C}^{\mathrm{A}}(\mathrm{Pa}(\mathrm{p}),\mathrm{f}) - \mathrm{S}(\mathrm{Pa}(\mathrm{p}),\mathrm{p})\mathrm{C}^{\mathrm{A}\uparrow}(\mathrm{p},\mathrm{f})\end{aligned} \tag{9}$$

We get the final aggregated cost of all pixels in very low computational complexity, which is linear to the number of image pixels.

To construct an optimal tree for each superpixel, we use the strategy of ST [29]. First, we construct random forest that each pixel represents a tree, all edges of pixels within superpixel are visited in weight-increasing order, two subtrees are linked by the connected edge w.p; q/, which is defined as:

$$\omega(\mathrm{p},\mathrm{q}) \le \min(\mathrm{Int}(\mathrm{T}_{\mathrm{p}}) + \frac{\mathrm{k}}{|\mathrm{T}_{\mathrm{p}}|}, \mathrm{Int}(\mathrm{T}_{\mathrm{q}}) + \frac{\mathrm{k}}{|\mathrm{T}_{\mathrm{q}}|}) \tag{10}$$

Here T_p is the size of tree p, $\mathrm{Int}(\mathrm{T}_{\mathrm{p}})$ denotes the maximum weight of edges in treeT_{p}, k is a constant parameter. We construct an optimal tree by the above merged edges progressively and select the first node representing the root node of each tree.

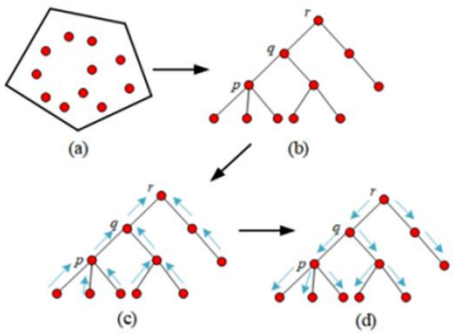

Figure 3. Processing of feature points optimization and twopass cost aggregation on a tree. (a) feature points in a superpixel. (b) The corresponding MST. (c) The cost aggregation form leaf nodes to root node. (d) The cost aggregation from root node to leaf nodes.

C optimization procedure

In this paper, we propose an efficient two-stage optimization strategy, which first extracts robust feature points and then constructs an optimal tree, followed by computing their labels in the optimal tree. Since the number of feature points is far less than the number of pixels of the corresponding superpixel, we can get the accurate 3D labels of feature points quickly, the process is described in Figure 3 and Algorithm1.

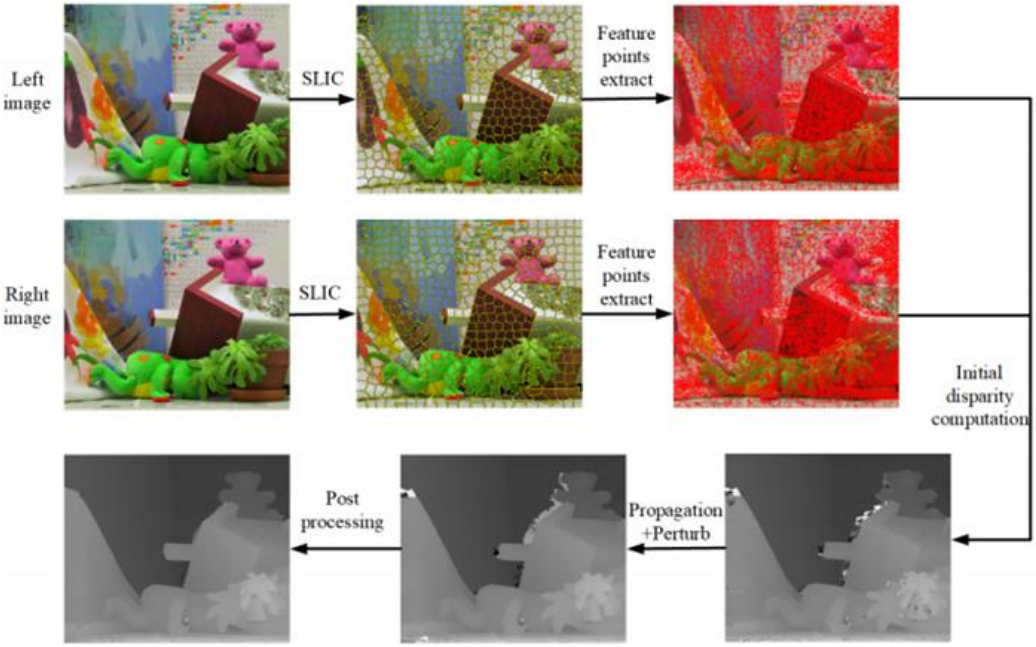

Figure 4. The workflow of our method. We first generate the superpixel image by SLIC, and extract the feature points by triangulation. After we get the labels of feature points, we update the corresponding superpixel, and then perform spatial propagation and random refinement. Propagation is spatial propagation, Perturb is random refinement.

After obtaining the labels of feature points, we form the candidate 3D labels set for the corresponding superpixel and compute the labels of minimum aggregated cost followed by the strategy of MST. Next, we select a label from each neighboring superpixels to form the labels set for performing spatial propagation, then, we update the labels of pixels. Finally, we perform random refinement as described in PatchMatch Stereo [17], and get perturbed labels set followed by computed the labels of minimum aggregated cost as spatial propagation. The procedure is described as Algorithm 2.

Algorithm 1 Feature points optimization
Input: superpixel image
Output: the 3D labels of feature points
1: construct superpixels S by SLIC;
2: for each superpixel s ， S do
3: extract edge points and robust points in s;
4: consist feature points and construct MST by Eq.(3);
5: assign a random 3D labels to feature points
6: compute aggregated cost by Eq.(8) and Eq.(9)
7: for each superpixel s ， S do
8: while max iteration time Kfea is not reached:
9: //Spatial propagation
10: construct labels set P from the feature points of neighbor superpixels.
11: for each f∈P:
12: compute aggregated cost by Eq.(8) and Eq.(9)
13: update feature points labels with lower aggregated cost;
//Refinement
14: select a label fp randomly from the feature points in s ;
15: for Δ is sufficiently small:

16: $f_p' \leftarrow f_p + \Delta$
17: compute aggregated cost by Eq.(8) and Eq.(9)
18: update feature points labels with lower aggregated cost;
19: $\Delta \leftarrow \Delta/2$
Algorithm 2 Overview of superpixel optimization
Input: stereo image pair
Output: the optimal labels f of each pixel
1: construct superpixel image S by SLIC;
2: feature optimization by Algorithm 1
3: for each superpixel s∈S:
4: construct labels set L from the feature points of s
5: for each f∈L :
6: compute aggregated cost by Eq.(8) and Eq.(9)
7: update the label of each pixel of s with lower aggregated cost;
8: for each superpixel s∈S:
9: while max iteration time KSP is not reached:
10: //spatial propagation
11: construct labels set P from neighbor superpixels
12: for each f∈P:
13: compute aggregated cost by Eq.(8) and Eq.(9)
14: update the label of each pixel of s with lower aggregated cost;
//Refinement
15: select a label fp randomly from pixels labels of s ;
16: for Δ is sufficiently small:
17: $f_p' \leftarrow f_p + \Delta$
18: compute aggregated cost by Eq.(8) and Eq.(9)
19: update the label of each pixel of s with lower aggregated cost;
20: $\Delta \leftarrow \Delta/2$
21: Do post processing;

In Algorithm 1, line 3-4 constructs feature points, line 5-6 initializes the feature points. For the plane $\mathbf{f_p}$, we select a random disparity $\boldsymbol{Z_0}$ in the allowed continuous disparity range [0, dispmax], and the normal vector of plane as a random unit $\mathbf{n} = (\mathbf{n_x}, \mathbf{n_y}, \mathbf{n_z})$, then, the plane is represented by $\mathbf{a_p} = -(\mathbf{n_x}/\mathbf{n_z}), \mathbf{b_p} = -(\mathbf{n_y}/\mathbf{n_z})$, $\mathbf{c_p} = (\mathbf{n_x x_0} + \mathbf{n_y y_0} + \mathbf{n_z z_0})/\mathbf{n_z}$. line 10-13 performs spatial propagation, which uses the labels of neighboring superpixel feature points to update center-superpixel feature points labels, line 14-19 uses random refinement, the perturbation term Δ is implemented as described in PatchMatch Stereo. We convert a label $\mathbf{f_p}$ to the form of a disparity d and normal vector representation $\vec{\mathbf{n}}$. Then, we iteratively add one random value $\Delta \in [-\Delta_{\mathbf{d}}^{\mathbf{max}}, \Delta_{\mathbf{d}}^{\mathbf{max}}]$ to d and one vector $[-\Delta_{\mathbf{d}}^{\mathbf{max}}, \Delta_{\mathbf{d}}^{\mathbf{max}}]$ to $\vec{\Delta}_{\mathbf{n}}$, i.e, $\mathbf{d}' = \mathbf{d} + \Delta_{\mathbf{d}}$ and $\vec{\mathbf{n}}' = \mathbf{u}(\vec{\mathbf{n}} + \vec{\Delta}_{\mathbf{n}})$, the new disparity and normal vector are converted to obtain a perturbed label $\mathbf{f}'$, we start set $\Delta_{\mathbf{d}}^{\mathbf{max}} = \mathbf{dispmax/2}$ and $\Delta_{\mathbf{n}}^{\mathbf{max}} = \mathbf{1}$, and then reduce exponentially the search scope by half until $\Delta_{\mathbf{d}}^{\mathbf{max}} < \mathbf{0.1}$ to obtain the candidate labels set and update the labels of feature points by Eq.(8) and Eq.(9).

In Algorithm 2, line 3-7 performs initialization using the labels of feature points, instead of random initialization.Finally, post-processing uses left-right consistency check and weighted median filtering for further improving the results, which is widely employed in recent methods.

D Fast implemention

Since the shape of the superpixel is irregular, it is timeconsuming if tracing all superpixels sequentially. Hence, we design the local patch surrounding the corresponding superpixel as depicted in Figure 5 to perform mask operation in parallel, which accelerates our algorithm. Then, we discuss the complexity of two stage optimization separately. In the stage of the feature points optimization, the complexity of initialization of each superpixel is O(N), N is the number of feature points in a superpixel. The feature points in the same superpixel are assigned a random label during initialization, i.e, the feature points in the same superpixel share the same label. The complexity of spatial propagation is O(N*P), P is the number of neighbor-superpixels labels. The complexity of refinement is O(N*R), R is the number of perturbed labels about 10.

In the stage of superpixel optimization, the complexity of initialization is O(M*F), M is the number of pixels in a superpixel, F is the number of the corresponding labels of feature points, and then the complexity of spatial propagation and random refinement is similar to the feature points optimization strategy.

EXPERIMENTS

In the experiments, we first evaluate our ranks under different criteria on Middlebury benchmark V3 benchmark [44] and KITTI 2015 [45] with realted methods. Then, we give comparison results with other similar methods: MeshStereo [28], Local Plane Sweep (LPS) [33], ELAS [29] and 3D MST [35] in qualitative and quantitative aspects. We also compare with other methods on Middlebury benchmark V2: ST [34] and PatchMatch Stereo [17]. Next, we analyze the sensitivity to key parameters and the effectiveness of the proposed core models of our algorithm.

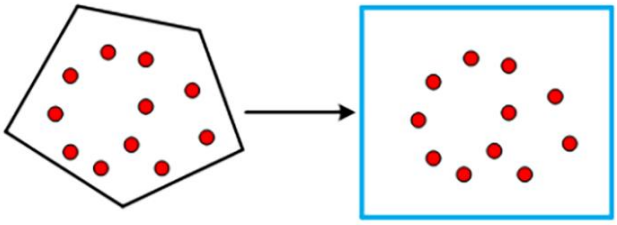

Figure 5. Fast implementation of our method. We transform the irregular superpixel structure into regular patch structure to accelerate our algorithm in parallel.

Parameter Setting

The experiments is conducted on the running environment with a desktop computer with an i5-6400U CPU 2.90-GHZ with 8GB memory. All methods are implemented using C++ and OpenCV. Our matching cost is the pretrained MC-CNN followed by LocalExp [24], the number of superpixels S is 600, two iterative parameters: K_{fea} to control the feature points optimization iteration is set to 9 for Middlebury V3 and KITTI 2015, 7 for Middlebury V2, K_{SP} to control the superpixel optimization iteration is set to 7 for Middlebury V3, 4 for KITTI 2015 and 3 for Middlebury V2. The parameter γ is set to 50 and k is set to 8000.

Evaluation on Middlebury Benchmark V3

Table 1. Snapshot of ranks on the test set of Middlebury benchmark V3 under the criterion “bad 2.0 nonocc metric” by the time of submission (April 2 2021). Five realted algorithms with published papers are listed here. The best results are in bold.

Method	Ours	Mesh[28]	AdaStereo [43]	AANet++[38]	SGM[15]	LPS[33]	Deeppruner [42]	ELAS[29]
Res	H	H	H	H	H	H	H	F
Avg	**10.7**	13.2	13.7	15.4	18.4	19.2	30.1	32.3
Austr	**5.15**	5.90	19.6	17.5	40.3	6.14	34.2	50.9
AustrP	**4.23**	4.88	7.41	8.37	4.54	5.34	19.9	9.17
Bicyc2	**5.48**	10.8	10.6	10.2	8.03	9.24	24.3	11.0
Class	**6.38**	12.9	14.5	9.86	22.9	7.53	23.8	33.0
ClassE	16.5	**10.6**	15.7	23.9	40.5	96.0	47.2	88.2
Compu	**7.84**	11.0	7.85	9.82	11.4	12.3	26.1	18.3
Crusa	**9.56**	12.2	22.6	17.7	24.7	9.61	26.1	47.3
CrusaP	10.3	**9.01**	9.32	15.9	10.1	9.40	22.8	26.8
DjembL	4.02	5.39	7.00	**3.25**	5.40	5.18	18.4	11.7
Hoops	**19.0**	23.5	22.4	27.1	28.5	27.4	36.5	37.4
Liv	**17.7**	17.7	14.5	16.2	23.9	24.3	23.2	23.7
Nkuba	18.5	21.0	17.8	**18.4**	20.0	23.0	31.7	28.8
Plants	**9.73**	15.4	14.8	20.0	14.2	10.0	48.3	63.0
Stairs	**18.0**	20.9	24.2	37.7	30.9	25.6	44.8	42.8

We test our algorithm for comparison with other similar algorithms on Middlebury benchmark V3, which has more challenging difficulties as different exposure and illuminations between an image pair, slight rectification errors, etc. It has 30 high-resolution image pairs, which are divided into 15 image pairs for

training and 15 image pairs for test. For fair comparison with other algorithms, we use half-resolution, and interpolate the disparity map to restore the full-resolution, which is the same as other methods. The metric “bad 2.0” under full-resolution image is the percentage of “bad” pixels whose error is greater than 2.0, which equals to the percentage of “bad” pixels whose error is greater than 1.0 under half-resolution image. Hence, we use the “bad 2.0” metric as our main criteria, which are the widely used criteria for comparison of stereo matching accuracy, and there exists two type of criteria for comparison, i.e, “nonocc” and “all”, nonocc is the pixels in the non occlued regions and all is the pixels in all regions including non occluded and occluded regions.

Table 1 show the rankings of our method for the bad 2.0 "nonocc" metric under the evaluation about the test set, and compare with the other classic methods, including MeshStereo [28], LPS [33], ELAS [29] and SGM[15] and the recent state-of-the-art deep learning methods, e.g, AdaStereo [43], AANet++ (improved AANet) [38] and Deeppruner [42]. Our method outperforms them and gets the minimum error rate within 2.0 pixels at full resolution, and has very high robustness for non-occluded and all areas under the evaluation of bad 1.0, 4.0, avgerr metric.

Visualization of the comparison with the classic methods is shown in Figure 6. The disparity maps of our method are visually much more accurate than the classic methods and natural at challenging image regions, e.g, weak texture or textureless regions as wire in “Motorcycle”, pipeline in “Pipes”, chair and wall in “Playroom”, lid in “Recycle”. Moreover, our method perform better than the recent deep learning methods on the test set of Middlebury benchmark V3 as shown in Figure 7.

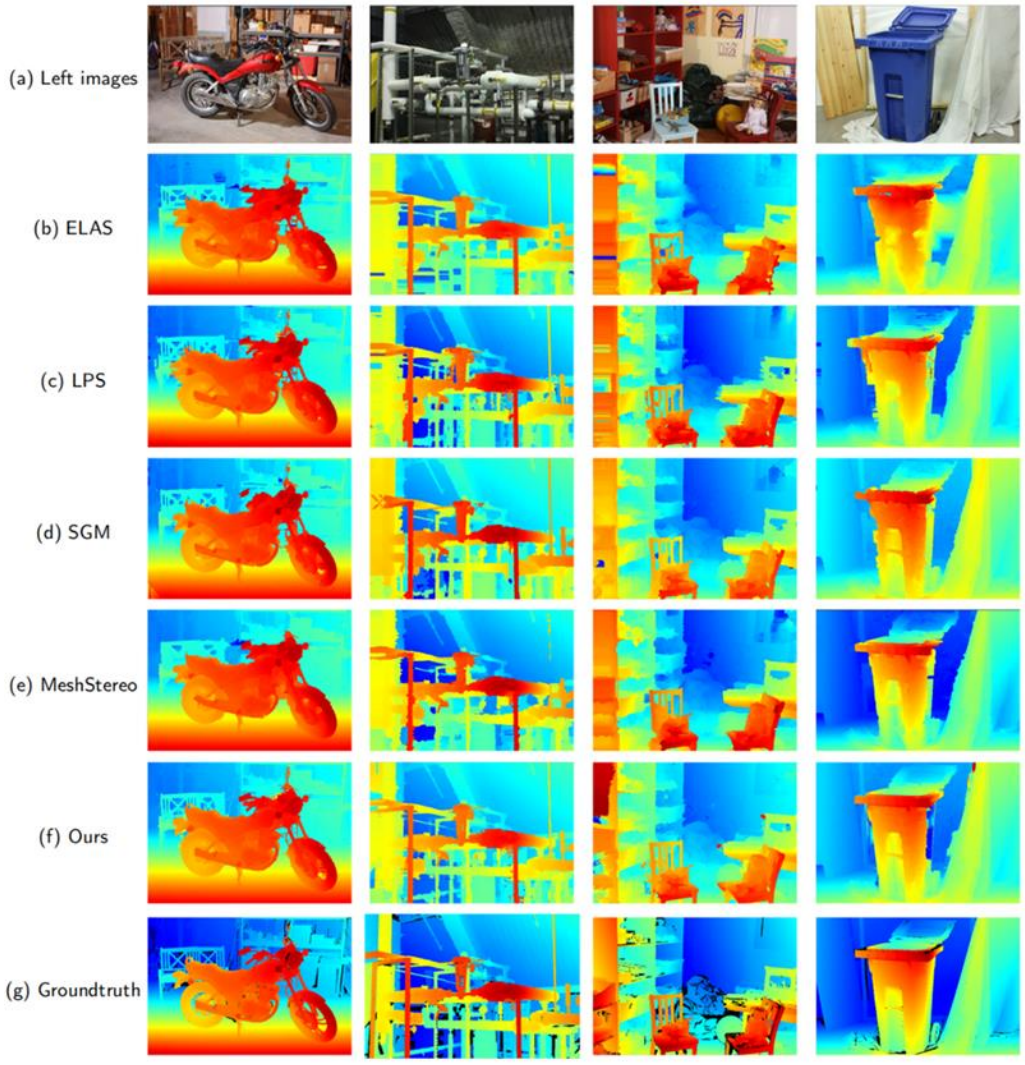

Figure 6. Disparity maps of our proposed method and classic methods on training set of Middlebury V3 benchmark, including (a) input left images, (b) ELAS [29], (c) LPS [33], (d) SGM [15], (e) MeshStereo [28] and (f) Our method.

Evaluation on KITTI 2015

Table 2. Quantitative results about error rates with threshold 3px on the test set of KITTI 2015.

Method	D1-bg	D1-fg	D1-all
Pretrained MC-CNN	13.20	28.26	15.71
MST [16]	45.83	38.22	44.57
ELAS [29]	7.86	19.04	9.72
Ours	5.18	17.81	7.28
3DMST [35]	3.36	13.03	4.97

KITTI 2015 is a real-world dataset with street views, which contains 200 training image pairs with sparse groundtruth disparities obtained using LiDAR and 200 testing image pairs. Image resolution is 376 x 1240 pixels. KITTI 2015 contains many weak texture and textureless regions about street. It is a challenge for our algorithm to get accurate disparity maps on KITTI 2015. “D1-all” is the error rate of all pixels that have ground truth with an error threshold of 3 pixels, “D1-bg” and “D1-fg” are error rates of foreground and background pixels. Our algorithm adpots a pretrained MC-CNN [45] (the convolution output without subsequent optimization) as cost volume to compute matching cost in Eq.(7). Moreover, we compare the releated algorithm with ours, such as ELAS [29], MST [16], 3DMST [35].

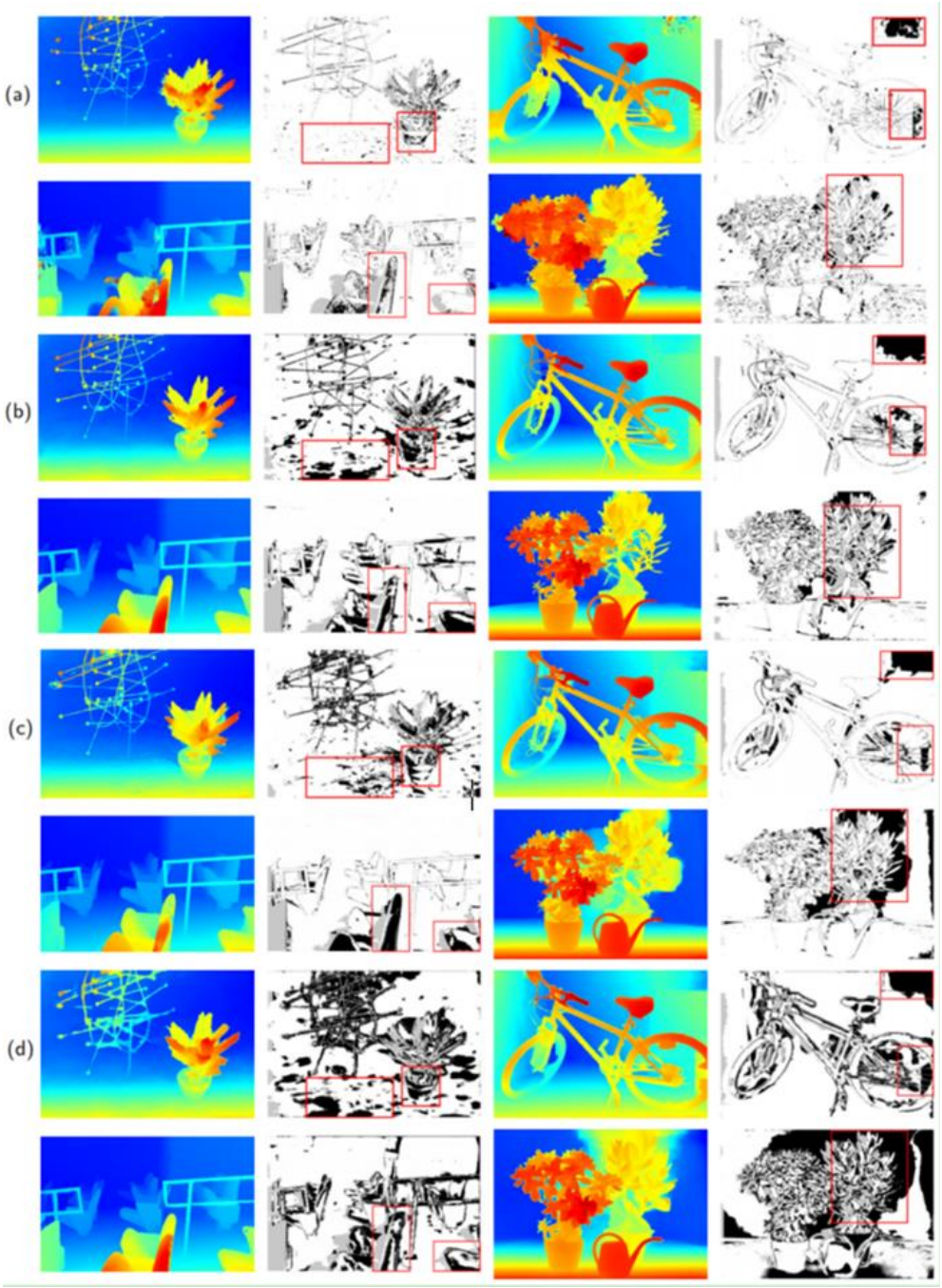

Figure 7. Comparison of disparity maps and error maps generated from the recent state-of-the-art deep learning methods on test set of Middlebury benchmark V3. (a) Ours, (b) AdaStereo [43], (c) AANet [38] and (d) DeepPruner [42], where black regions represent the error regions.

The quantitative comparison is described as Table 2. Our method outperforms them by a margin except 3DMST. Although the error rate of 3D MST is lower than ours, our algorithm runs faster than 3DMST[35]. Our average running time is about 58s, the running time of 3DMST is 92s. We compute disparity map by extracting candidate disparities for the corresponding superpixel, not like 3DMST performs random disparities for all pixels. The qualitative comparison is shown as Figure 8, since our method computes the disparity map independently without global regularization, such as GC [10] and BP [8], our results exist noisier but have better detail information.

Compare with other classic methods

Since the Middlebury benchmark V2 is no longer active, it consists of four image pairs named “Tsukuba”, “Venus”, “Teddy” and “Cones”, which has upgraded to V3 version with more challenging conditions. However, most of the existing classic methods are designed for about 0.1 Mpixel lowresolution stereo images (450x375). Therefore, we compare our methods with other classic methods: PatchMatch stereo [17], ST [34] on Middlebury benchmark V2, the code is provided public by authors.

For fair comparison, we replace the matching cost with the "gradient + color" matching cost as the same as PatchMatch stereo and ST, which is defined as:

$$C(p,f) = (1-\alpha)\min(||I_L(p) - I_R(p')||_1, \tau_{col}) + \alpha(|\Delta_x I_L(p) - \Delta_y I_R(p')||_1, \tau_{grad}) \quad (11)$$

Here, $\{\alpha,\tau_{col},\tau_{grad}\}$={0.9,10,2}, S = 200, $\{K_{fea},K_{SP}\}$={7,3},the other parameters as default. The number of feature points for four pairs of steroe images obtained by our method are about 1/12, 1/10, 1/8, 1/8 of original image respectively. The average processing time of ST taks 0.5s, ours takes 4.5s,PatchMatch stereo takes 700s.

The performance comparison is shown in Figure 9, our method gets the accurate disparities near edge regions and background than PM and ST, but we still can not deal with the some challenging situations, such as the lamp in tsukuba. Although the running time of ST [34] is faster than ours, the disparity of our algorithm is more accurate than ST [34] by a margin.

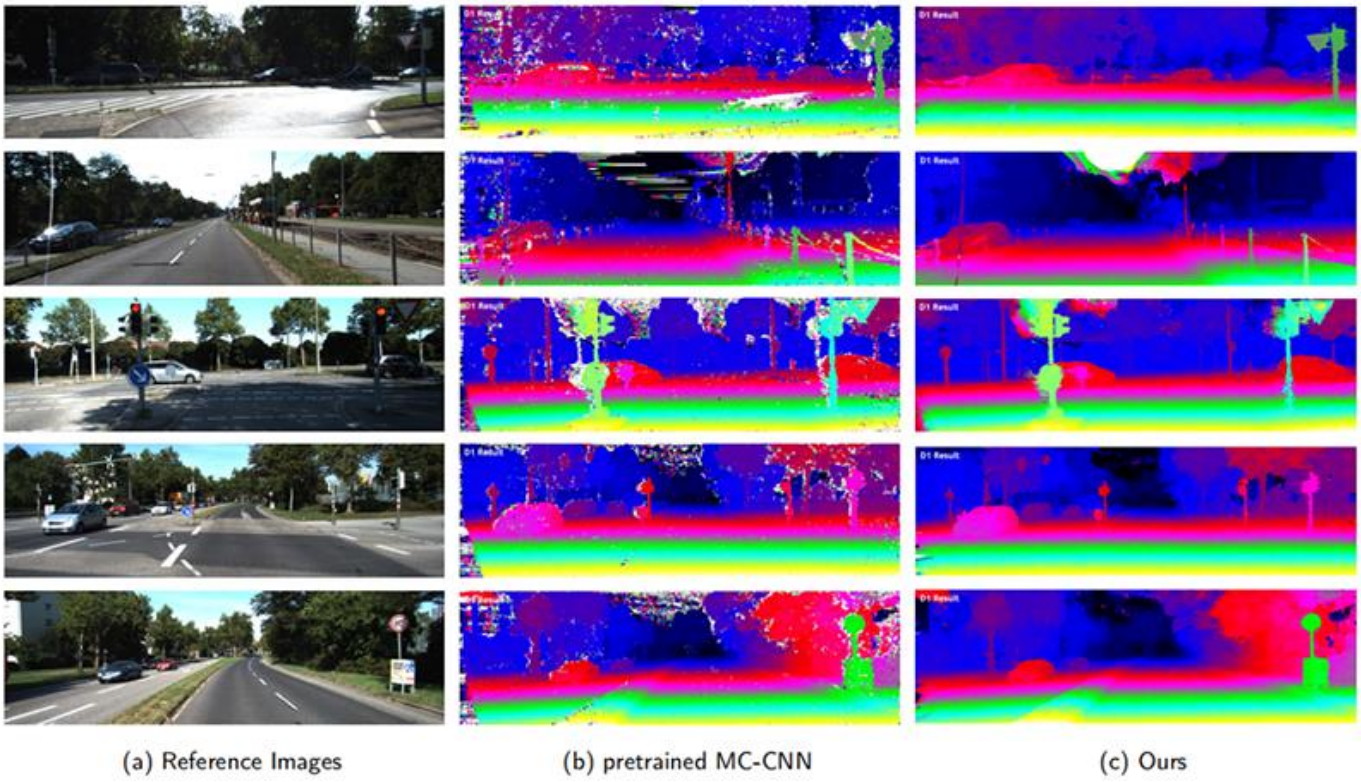

Figure 8. Qualitative results of the proposed approach on the training set of KITTI 2015 benchmark, (a) Reference image, (b) pretrained MC-CNN [45], (c) Our method. The disparity maps of our method achieve greatly improvement based on pretrained MC-CNN [45].

Parameters Analysis

In Algorithm 1 and 2, there exists three main parameters for control the optimization of our method, i.e, the number of superpixels S, the iterations of feature points optimizationK_{fea}, the iterations of superpixels K_{SP}. We select six stereo images of the training set (Adirondack, Atrl, Motorcycle, Pipes, Recycle, Teddy) to analyze the parameters. In Algorithm 1, the number of the feature points obtained by our feature points extraction method in left image is about 1/5, 1/7, 1/8, 1/8, 1/4, 1/10 of the original scale image, hence, we can compute the disparities of feature points for the candidate disparities of corresponding superpixels quickly and take the error rate of the metric "nonocc" within 1 pixel as our evaluation. First, we random choose a superpixel image, e.g, 500, and perform Algorithm 1. As shown in Figure 10(a), the error rate in the optimization of feature points is no longer reduced and approximate stable while the number of iterations exceeds 9.

Hence, we set the number of iterations K_{fea} to 9. Next, we need to set the iteration number K_{SP} to be large enough, such as 20, to ensure the convergence of our algorithm for determining the number of superpixels S. Figure 10(b) shows the optimal number is 600 for the trade-off between accuracy and time. After we determine K_{fea} and K_{SP}, we then determine the parameter K_{SP}. As described in Figure 10(c), some

of stereo images get the optimal disparities in the first 2 or 3 iterations and the error rate of most images is no longer declined where the number of iteration exceeds 7. Hence, we set K_{SP} to 7 for comprehensive consideration.

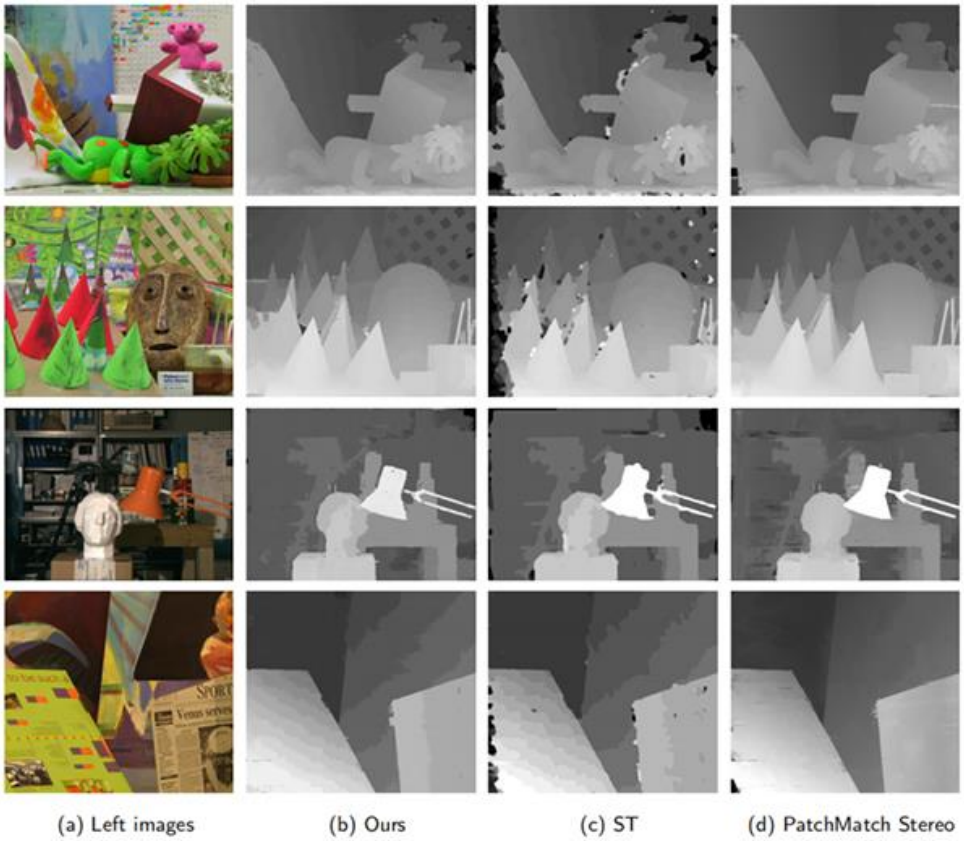

Figure 9. Qualitative disparity results of our proposed approach and classical approaches on Middlebury 2.0 benchmark, where (a) Left images, (b) Our method, (c) ST [34], (d) PatchMatch Stereo [17].

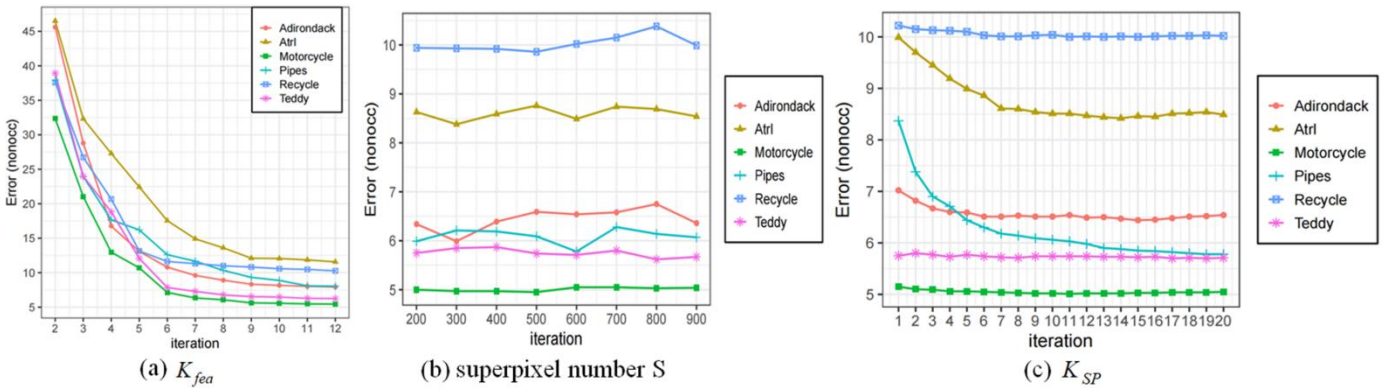

Figure 10. Sensitivity analysis of parameters on the Middlebury V3 benchmark. Variation of error rates with different parameters values separately, in which the x-axis represents the error values under “nonocc” regions ,i.e., non-occluded regions, the y-axis represents the iteration number, (a) the number of iterations K_{fea} in the feature extraction stage , (b) the superpixel number S and (c) the number of iterations K_{SP} in the superpixel optimization stage.

Effect of each model

Our method has two core models: (1) the feature points optimization, which get the candidate label sets for superpixels; (2) the weight of cost aggregation uses the combination of intensity, gradient and binary image, instead of only intensity; We employ ablation experiments to demonstrate the contribution of each part of our methods. As described in Figure 10(c), some images can get the optimal disparities at the begining iterations after the update of candidate labels extracted by feature points, which is better than assigning a random label to the pixels of superpixels and then optimization by many iterations. In the case of no feature points optimization, the error rate of superpixels gets the minimum after many iterations. Even some images can not get the minimum error rate after many iterations as "AtrL" in Figure 11. Since the number of feature

points in a superpixel is far less than the number of the corresponding-superpixel pixels, the feature points optimization is obviously faster than to optimize all pixels of corresponding superpixel, which can obtain the accurate candidate label sets quickly.

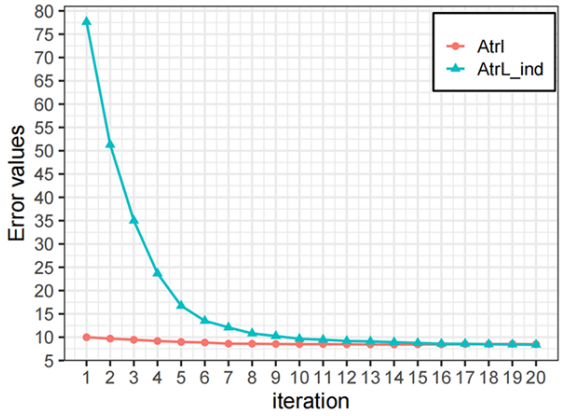

Figure 11. The error rate map of superpixels without the feature points optimization, AtrL_ind in map represents AtrL without the feature points optimization, where the x-axis represents the error values under "nonocc" regions, i.e., non-occluded regions, the y-axis represents the iteration number.

We select some key evaluation metric to compare different weights, e.g, "aggerr" means average absolute error of pixels and table 3 shows the quantitative comparison of the weight using "intensity", "intensity + gradient" and our method. The "intensity + gradient" weight reduces some error rate relative to the weight using intensity. Since neither intensity nor gradient can deal with weak texture or textureless regions well, the binary image improve the effect of weak texture or textureless regions as the inset boxes in Figure 12. Hence, our method adds binary image of superpixels based on the weight of intensity + gradient as shown in table 5. Our method gets the minimize error rate under the evaluation.

Table 3. Comparison of the effectiveness of different module under different criteria on training set of Middlebury 3.0 dataset. The best results are in bold.

Method	Bad 2.0		Bad 1.0		Bad 4.0		Aggerr	
	nonocc	all	nonocc	all	nonocc	all	nonocc	all
Intensity	12.4	18.7	24.7	30.8	7.68	13.4	3.37	7.36
Intensity+gradient	12.2	18.6	24.7	30.8	7.57	13.2	3.31	7.40
Ours	**11.8**	**18.2**	**24.3**	**30.4**	**7.26**	**12.9**	**3.06**	**7.06**

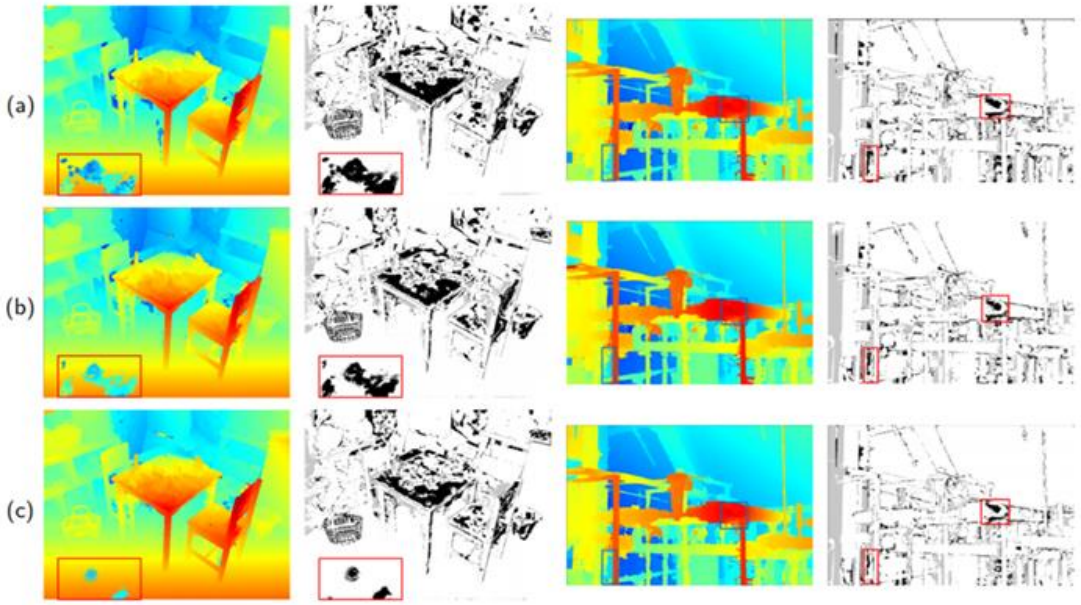

Figure 12. Comparative analysis view of different combination weights. (a) intensity,(b)intensity + gradient, (c) ours, where black regions represent the error regions.

CONCLUSION

In this paper, we propose a novel two-stage optimization strategy to get accurate 3D labels for pixels efficiently, i.e, the feature points and superpixel optimization. Moreover, we design the weight combination of "intensity + gradient + binary image" to construct an optimal minimum spanning tree and obtain the accurate label of minimum aggregated matching cost. Moreover, we also design the local patch surrounding the corresponding superpixel to accelerate our algorithm. Experiment shows that the proposed method a good trade-off between running time and accuracy on the multiple datasets among similar classic methods and deep learning methods. However, the proposed method is still difficulty for some challenging scenarios, such as occluded and textureless regions. In the future, we will add the efficient fusion method to improve our method.

Acknowledgments: This work is supported by the Foundation of Guizhou Provincial Department of Education Youth Science and Technology Talents Growth Project(Project Nos.qianjiaoheKYzi[2017]251,[2018]411), The Basic Research Program of 2019 Science and Technology Fund Project of Guizhou (Qiankehe foundation [2019]1250), and Science and Technology Planning of Shenzhen(JCYJ20180503182133411), Guizhou Science and Technology Planning Project(Qiankehejichu[2020]1Y420).
Conflicts of Interest: The authors declare no conflict of interest.

REFERENCES

1. Biswas J, Veloso M. Depth camera based localization and navigation for indoor mobile robots.RGB-D Workshop at RSS. 2011,2011(21). https://doi.org/10.1109/ICRA.2012.6224766
2. Pingbo Hu, Bisheng Yang, Zhen Dong, Pengfei Yuan, Ronggang Huang, Hongchao Fan, et al.Towards reconstructing 3D buildings from ALS data based on gestalt laws. Remote Sensing, 2018,10(7):1127. https://doi.org/10.3390/rs10071127
3. Xiao J, Furukawa Y. Reconstructing the world's museums. Int. J. Comput. Vis., 2014,110(3):243-258. https://doi.org/10.1007/s11263-014-0711-y
4. Chenyi Chen, Ari Seff, Alain Kornhauser, Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. Proceedings of the IEEE international conference on computer vision. 2015:2722-30. https://doi.org/10.1109/ICCV.2015.312
5. Haque AU, Nejadpak A. Obstacle avoidance using stereo camera. arXiv preprint arXiv:1705.04114, 2017.https://arxiv.org/abs/1705.04114
6. Scharstein D, Szeliski R. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Int. J. Comput. Vis., 2002, 47(1): 7-42. https://doi.org/10.1109/SMBV.2001.988771
7. Victor Lempitsky, Carsten Rother, Stefan Roth, Andrew Blake. Fusion moves for markov random field optimization. IEEE transactions on pattern analysis and machine intelligence, 2009,32(8):1392-405. https://doi.org/10.1109/TPAMI.2009.143
8. Sun J, Zheng N N, Shum H Y. Stereo matching using belief propagation. IEEE Transactions on pattern analysis and machine intelligence, 2003,25(7):787-800. https://doi.org/10.1109/ICIP.2008.4712121
9. Klaus A, Sormann M, Karner K. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. 18th International Conference on Pattern Recognition (ICPR'06). IEEE, 2006,3:15-8. https://doi.org/10.1109/ICPR.2006.1033
10. Boykov Y, Veksler O, Zabih R. Fast approximate energy minimization via graph cuts. IEEE Transactions on pattern analysis and machine intelligence, 2001,23(11):1222-39. https://doi.org/10.1109/ICCV.1999.791245
11. Kolmogorov V, Zabih R. Computing visual correspondence with occlusions using graph cuts. Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001. IEEE, 2001,2:508-15. https://doi.org/10.1109/ICCV.2001.937668
12. Tappen M F, Freeman W T. Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. Computer Vision, IEEE International Conference on. IEEE Computer Society, 2003, 3: 900-7. https://doi.org/10.1109/ICCV.2003.1238444
13. Ihler A T, Fisher III J W, Willsky A S, et al. Loopy belief propagation: convergence and effects of message errors. J. Mach. Learn. Res. 2005,6(5):905-36. https://doi.org/10.1.1.70.5347
14. Kolmogorov V. Convergent tree-reweighted message passing for energy minimization. International Workshop on Artificial Intelligence and Statistics. PMLR, 2005:182-9. https://doi.org/10.1109/TPAMI.2006.200
15. Hirschmuller H. Stereo processing by semiglobal matching and mutual information. IEEE Transactions on pattern analysis and machine intelligence, 2007,30(2):328-41. https://doi.org/10.1109/TPAMI.2007.1166
16. Yang Q. A non-local cost aggregation method for stereo matching. 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2012:1402-9. https://doi.org/10.1109/CVPR.2012.6247827

17. Bleyer M, Rhemann C, Rother C. PatchMatch Stereo-Stereo Matching with Slanted Support Windows. Bmvc. 2011,11:1-11. http://dx.doi.org/10.5244/C.25.14
18. Lu J, Yang H, Min D, et al. Patch match filter: Efficient edge-aware filtering meets randomized search for fast correspondence field estimation. Proceedings of the IEEE conference on computer vision and pattern recognition. 2013:1854-61. https://doi.org/10.1109/CVPR.2013.242
19. Frederic Besse, Carsten Rother, Andrew Fitzgibbon, Jan Kautz. Pmbp: Patchmatch belief propagation for correspondence field estimation. Int. J.Comput. Vis, 2014,110(1):2-13. https://doi.org/10.1007/s11263-013-0653-9
20. Zhuoqun Fang, Xiaosheng Yu, Chengdong Wu, Dongyue Chen, Tong Jia. Superpixel Segmentation Using Weighted Coplanar Feature Clustering on RGBD Images. Applied Sciences, 2018,8(6):902. https://doi.org/10.3390/app8060902
21. Qian X, Li X, Zhang C. Weighted superpixel segmentation. Visual Comput., 2019,35(6):985-96. https://doi.org/10.1007/s00371-019-01682-x
22. Lincheng Li, Shunli Zhang, Xin Yu, Li Zhang. PMSC: PatchMatch-based superpixel cut for accurate stereo matching. IEEE Transactions on Circuits and Systems for Video Technology, 2016,28(3):679-92. https://doi.org/10.1109/TCSVT.2016.2628782
23. Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, .Sabine Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. IEEE transactions on pattern analysis and machine intelligence, 2012,34(11):2274-82. https://doi.org/10.1109/TPAMI.2012.120
24. Taniai T, Matsushita Y, Sato Y, Naemura, T. Continuous 3D label stereo matching using local expansion moves. IEEE transactions on pattern analysis and machine intelligence, 2017,40(11):2725-39. https://doi.org/10.1109/TPAMI.2017.2766072
25. Hirschmuller H, Scharstein D. Evaluation of cost functions for stereo matching. 2007 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2007:1-8. https://doi.org/10.1109/CVPR.2007.383248
26. Mao Tian, Bisheng Yang, Chi Chen, Ronggang Huang, Liang Huo. HPM-TDP: An efficient hierarchical PatchMatch depth estimation approach using tree dynamic programming. ISPRS J. Photogramm. Remote Sens., 2019,155:37-57. https://doi.org/10.1016/j.isprsjprs.2019.06.015
27. Kang Zhang, Yuqiang Fang, Dongbo Min, Lifeng Sun, Shiqiang Yang, Shuicheng Yan, et al. Cross-scale cost aggregation for stereo matching. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2014:1590-7. https://doi.org/10.1109/TCSVT.2015.2513663
28. Chi Zhang, Zhiwei Li, Yanhua Cheng, Rui Cai, Hongyang Chao, Yong Rui. Meshstereo: A global stereo model with mesh alignment regularization for view interpolation. Proceedings of the IEEE International Conference on Computer Vision. 2015:2057-65. https://doi.org/10.1109/ICCV.2015.238
29. Geiger A, Roser M, Urtasun R. Efficient large-scale stereo matching. Asian conference on computer vision. Springer, Berlin, Heidelberg, 2010:25-38. https://doi.org/10.1007/978-3-642-19315-6_3
30. Radouane Ait Jellal, Manuel Lange, Benjamin Wassermann, Andreas Schilling, Andreas Zell. LS-ELAS: Line segment based efficient large scale stereo matching. 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2017:146-52. https://doi.org/10.1109/ICRA.2017.7989019
31. Guilherme Pinto Fickel, Claudio R. Jung, Tom Malzbender, Ramin Samadani, Bruce Culbertson. Stereo matching and view interpolation based on image domain triangulation. IEEE transactions on image processing, 2013,22(9):3353-65. https://doi.org/10.1109/TIP.2013.2264819
32. Mozerov M G, Van De Weijer J. Accurate stereo matching by two-step energy minimization. IEEE Transactions on Image Processing, 2015,24(3):1153-63. https://doi.org/10.1109/TIP.2015.2395820
33. Sinha S N, Scharstein D, Szeliski R. Efficient high-resolution stereo matching using local plane sweeps. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2014:1582-9. https://doi.org/10.1.1.648.5914
34. Xing Mei, Xun Sun, Weiming Dong, Haitao Wang, Xiaopeng Zhang. Segment-tree based cost aggregation for stereo matching. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2013: 313-320. https://doi.org/10.1109/CVPR.2013.47
35. Lincheng Li, Xin Yu, Shunli Zhang, Xiaolin Zhao, Li Zhang. 3D cost aggregation with multiple minimum spanning trees for stereo matching. Appl. Opt., 2017, 56(12): 3411-20. https://doi.org/10.1007/978-3-030-41404-7_25
36. Zbontar J, LeCun Y. Stereo matching by training a convolutional neural network to compare image patches. J. Mach. Learn. Res., 2016, 17(1): 2287-2318. https://arxiv.org/abs/1510.05970
37. Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 2495-504. https://doi.org/10.1109/CVPR42600.2020.00257

38. Xu H, Zhang J. Aanet: Adaptive aggregation network for efficient stereo matching. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020:1959-68. https://doi.org/10.1109/CVPR42600.2020.00203
39. Chang J R, Chen Y S. Pyramid stereo matching network. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018:5410-8. https://doi.org/10.1109/CVPR.2018.00567
40. Feihu Zhang, Victor Prisacariu, Ruigang Yang, Philip H.S. Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019: 185-94. https://doi.org/10.1109/CVPR.2019.00027
41. Cheng X, Wang P, Yang R. Learning depth with convolutional spatial propagation network. IEEE transactions on pattern analysis and machine intelligence, 2019,42(10):2361-79. https://doi.org/10.1109/TPAMI.2019.2947374
42. Shivam Duggal, Shenlong Wang, Wei-Chiu Ma, Rui Hu, Raquel Urtasun. Deeppruner: Learning efficient stereo matching via differentiable patchmatch. Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019:4384-93. https://doi.org/10.1109/ICCV.2019.00448
43. Xiao Song, Guorun Yang, Xinge Zhu, Hui Zhou, Zhe Wang, Jianping Shi. AdaStereo: a simple and efficient approach for adaptive stereo matching. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021:10328-37. https://arxiv.org/abs/2004.04627
44. Scharstein D, Szeliski R, Hirschmüller H. Middlebury evalution benchmark website. https://vision.middlebury.edu/stereo/
45. Menze M, Geiger A. Object scene flow for autonomous vehicles. Proceedings of the IEEE conference on computer vision and pattern recognition. 2015:3061-70. https://doi.org/10.1109/CVPR.2015.7298925