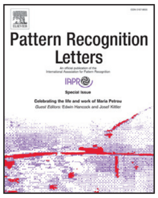
Pattern Recognition Letters 135 (2020) 402–408

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec



Efficient binocular stereo correspondence matching with 1-D Max-Trees

Rafaël Brandt, Nicola Strisciuglio, Nicolai Petkov, Michael H.F. Wilkinson*

Bernoulli Institute, University of Groningen, P.O. Box 407, AK Groningen 9700, the Netherlands

ARTICLE INFO

Article history:
Received 14 June 2019
Revised 4 December 2019
Accepted 19 February 2020
Available online 20 February 2020

MSC:
41A05
41A10
65D05
65D17

Keywords:
Stereo matching
Mathematical morphology
Tree structures

ABSTRACT

Extraction of depth from images is of great importance for various computer vision applications. Methods based on convolutional neural networks are very accurate but have high computation requirements, which can be achieved with GPUs. However, GPUs are difficult to use on devices with low power requirements like robots and embedded systems. In this light, we propose a stereo matching method appropriate for applications in which limited computational and energy resources are available. The algorithm is based on a hierarchical representation of image pairs which is used to restrict disparity search range. We propose a cost function that takes into account region contextual information and a cost aggregation method that preserves disparity borders. We tested the proposed method on the Middlebury and KITTI benchmark data sets and on the TrimBot2020 synthetic data. We achieved accuracy and time efficiency results that show that the method is suitable to be deployed on embedded and robotics systems.

1. Introduction

Extraction of depth from images is of great importance for computer vision applications, such as autonomous car driving [19], obstacle avoidance for robots [16], 3D reconstruction [24], Simultaneous Localization and Mapping [5], among others. Given a pair of rectified images recorded by calibrated cameras, a typical pipeline for binocular stereo matching exploits epipolar geometry to find corresponding pixels between the left and right image and create a map of their horizontal displacement, i.e. a disparity map. For a pixel (x, y) in the left image, its corresponding pixel $(x-d, y)$ is searched for in the right image and a matching cost is associated with it. If a corresponding pixel is found, the perceived depth is computed as Bf/d where B is the baseline, f the camera focal length and d is the measured disparity. The match with the lowest cost is used to select the best disparity value and construct the disparity map.

In the literature, various approaches to compute the matching cost have been proposed. The similarity between two pixels has often been expressed as their absolute image gradient or gray-level difference [23]. In regions with repeating patterns or without texture, the matching cost of a pixel can be very low at multiple disparities. To reduce such ambiguity, the similarity of the surrounding region of the concerned pixels can be measured instead. The matching cost of a pixel pair is computed as the (weighted) average of the matching cost of corresponding pixels in the surrounding regions. Therefore, the disparity predictions near disparity borders are unreliable when surrounding pixels with different disparity than the considered pixel pair have a non-zero weight [17]. Disparity borders have been estimated, for instance, using color similarity and proximity to weigh the contribution of a pixel to an average of another pixel by Yoon and Kweon [33]. A scheme which takes into account the strength of image boundaries in between pixels has been proposed by Chen et al. [1]. Zhang et al. [35] constructed horizontal and vertical line segments based on color similarity and spatial distance of pixels, and costs were aggregated over horizontal and then over vertical line segments.

The creation of large stereo data-sets with ground-truths [22] has facilitated the development of methods that learn a similarity measure between (two) image patches using convolutional neural networks (CNNs). One of the first CNN stereo matching methods, based on a siamese network architecture, has been proposed by Zbontar et al. [34]. An efficient variation has been proposed by Luo et al. [11] that formulated the disparity computation as a multi-class problem, in which each class is a possible disparity value. These two approaches are restricted to small patch inputs. Using larger patches may produce blurred boundaries [17]. Approaches to increase the receptive field while keeping details

[*] Corresponding author.
E-mail address: m.h.f.wilkinson@rug.nl (M.H.F. Wilkinson).

https://doi.org/10.1016/j.patrec.2020.02.019

have been proposed. Chen et al. [2] used pairs of siamese networks, each receiving as input a pair of patches at different scales. An inner product between the responses of the siamese networks computes the matching cost. A multi-size and multi-layer pooling module is used to learn cross-scale feature representations by Ye et al. [32]. Disparity search-range can be reduced by computing a coarse disparity map: [7] defined a triangulation on a set of support points which can be robustly matched. All resulting points need to be matched to obtain the coarse map. An alternative approach was to use image pyramids to reduce disparity search range [12,26]. Starting at the top of the pyramid, a coarse disparity map is constructed considering the full disparity range. The disparity search range used in the construction of higher-resolution disparity maps is dictated by the disparity map computed in the previous iteration. Matching (hierarchically structured) image regions rather than pixels to increase efficiency and reduce matching ambiguity has been proposed by Cohen et al. [3], Medioni and Nevatia [14], Todorovic and Ahuja [27]. Such methods may include computationally expensive segmentation steps. CNN-based methods are able to reconstruct very accurate disparity maps, although they require a large amount of labeled data to be trained effectively. Mayer et al. [13] showed that properly designed synthetic data can be used to train networks for disparity estimation. The main drawback of CNN-based approaches concerns their high computation requirements to process the large number of convolutions they are composed of. Although this can be efficiently achieved with GPUs, problems arise for embedded or power-constrained systems such as battery-powered robots or drones, where GPUs cannot be easily used and algorithms for depth perception are required to find a reasonable trade-off between accuracy and computational efficiency.

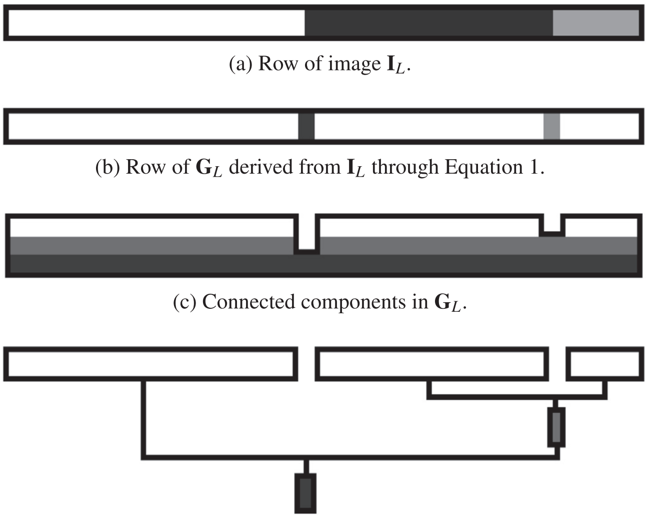
In this light, we propose a stereo matching method that balances efficiency with effectiveness, appropriate for applications in which limited computational and energy resources are available. It is based on a representation of image scan-lines using Max-Trees [20] and disparity computation via tree matching. Our main contribution is an efficient binocular narrow-baseline stereo matching algorithm which contains: a) a tree-based hierarchical representation of image pairs which is used to restrict disparity search range; b) a cost function that includes contextual information computed on the tree-based image representation; c) an efficient tree-based edge preserving cost aggregation scheme. We achieve competitive performance in terms of speed and accuracy on the Middlebury 2014 data set [22], KITTI 2015 data set [15] and the Trimbot2020 3DRMS Workshop 2018 data set [28]. We released the source code at the url https://github.com/rbrandt1/MaxTreeS.

2. Proposed method

We propose to construct a hierarchical representation of a pair of rectified stereo images by computing 1D Max-Trees on the scanlines. Leaf nodes in a Max-Tree correspond to fine image structures, while ancestors of leaf nodes correspond to coarser image structures. Nodes are matched in an iterative process according to a matching cost function that we define on the tree in a coarse-to-fine fashion, until leaf nodes have been matched. A depth map refinement step is performed at the end to remove erroneously matched regions.

2.1. Background: Max-Tree

Applying a threshold t to a 1D gray-scale image (Fig. 1b) results in a binary image, wherein a set of 1 valued pixels for which no 0 valued pixel exists in between any of the pixels is called a connected component [21]. Applying a threshold $t + 1$ will not result in connected components that consist of additional pixels. Connected components resulting from different thresholds can, instead, be represented hierarchically in the Max-Tree data structure proposed by Salembier et al. [20].


(a) Row of image $\mathbf{I}_L$.

(b) Row of $\mathbf{G}_L$ derived from $\mathbf{I}_L$ through Equation 1.

(c) Connected components in $\mathbf{G}_L$.

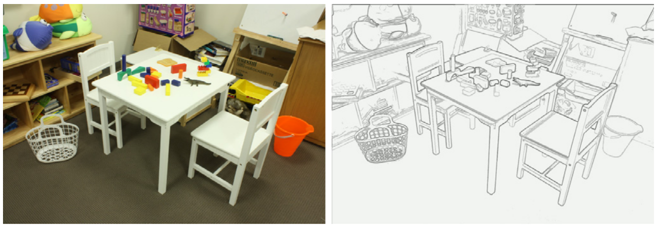
(d) Max-Tree of $\mathbf{G}_L$.

Fig. 1. Example of the construction of a Max-Tree for the image row in (a).

Each node in a Max-Tree corresponds to a set of pixels that have an equal gray level. Furthermore, all pixels in such a set are part of the same connected component arising when a threshold equal to the gray level of the pixels in the set is applied. The pixels in the connected component that have a lower gray level are included in a sub-tree of the concerned Max-Tree node. Recursively, all pixels in the sub-tree correspond to the same connected component arising when a threshold equal to the gray level of the pixels in the set is applied. Nodes may have attributes stored in them such as width, area, eccentricity, and so on. We denote the value of an attribute *attr* of node n as $attr(n)$. The connected components resulting from applying thresholds to Fig. 1b are illustrated in Fig. 1c. The corresponding Max-Tree is depicted in Fig. 1d. We construct Max-Trees using a 1-D version of the algorithm by Wilkinson [31].

Algorithm 1 Proposed stereo matching method.

```
Require: Input images $\mathbf{F}_L$ and $\mathbf{F}_R$, the maximum number of colors $q \in \mathbb{N}$, the coarse to fine levels $S \in \{\mathbb{N} \cup 0\}^n$, the maximum neighbourhood size $\theta_\gamma \in \mathbb{N}$, the weight of different cost types $0 \leq \alpha \in \mathbb{R}^+ \leq 1$, the minimum size of matched nodes $\theta_\alpha \in \mathbb{R}^+$, and the maximum size of matched nodes $\theta_\beta \in \mathbb{R}^+$, similarity threshold $\theta_\omega \in \mathbb{N}^+$.
Apply median blur to $\mathbf{F}_L$, and $\mathbf{F}_R$, resulting in $\mathbf{I}_L$, and $\mathbf{I}_R$.
Derive $\mathbf{G}_L$ and $\mathbf{G}_R$ from $\mathbf{I}_L$ and $\mathbf{I}_R$ through Eq. (1).
Compute a Max-Tree for each row in $\mathbf{G}_L$ and $\mathbf{G}_R$.
for coarse-to-fine levels, i.e. $i \in S$ do
    for each row $r$ do
        Determine nodes $\phi^i_{M^r_L}$ and $\phi^i_{M^r_R}$ (Section 2.2.1).
        if $i \neq S(0)$ then
            Determine disparity search range of nodes in $\phi^i_{M^r_L}$ and $\phi^i_{M^r_R}$ (Section 2.4).
        end if
        WTA matching based on aggregated cost.
        Left-right consistency check (Eq. (6)).
    end for
end for
Disparity refinement and map computation (Section 2.6).
return Disparity map.
```



(a) Original image (taken from Scharstein et al. (2014)). (b) Pre-processed image.

Fig. 2. Example of a pre-processed image.

Matching nodes in 1D, rather than 2D Max-Trees, has computational benefits: 1D Max-Trees can be constructed more efficiently than 2D Max-Trees. However, it also has benefits in terms of reconstruction accuracy. Our context cost (Section 2.3) allows to distinguish shapes because area is considered on a per line basis. When 2D area is used in the calculation of context cost, this is not possible.

2.2. Hierarchical image representation

Our method only uses gray-scale information of a stereo image pair. Let $\mathbf{F}_L$ and $\mathbf{F}_R$ denote the left and right images of a rectified gray-scale binocular image pair, with b-bit color-depth. To reduce noise, we apply a 5×5 median blur to both images, resulting in $\mathbf{I}_L$ and $\mathbf{I}_R$, respectively. Let $\mathbf{G}_L$ and $\mathbf{G}_R$ be inverted gradient images derived from $\mathbf{I}_L$ and $\mathbf{I}_R$, in which lighter regions correspond to more uniformly colored regions, while darker regions correspond to less uniformly colored regions (e.g. edges). An example of a pre-processed image is given in Fig. 2. We compute $\mathbf{G}_k, k \in \{L, R\}$ as:

$$\mathbf{G}_k = \left(\Phi\left((2^b - 1)\mathbf{J} - \frac{|\mathbf{I}_k * \mathbf{S}_x| + |\mathbf{I}_k * \mathbf{S}_y|}{2}\right) div\ \frac{2^b}{q}\right) \times \frac{2^b}{q}, \qquad (1)$$

where $q \in \mathbb{N} \leq 2^b$ controls the number of intensity levels in $\mathbf{G}_L$ and $\mathbf{G}_R$, $\mathbf{J}$ is an all-ones matrix, $\mathbf{S}_x$ and $\mathbf{S}_y$ are Sobel operators of size 5×5 measuring image gradient in the x and y direction, $*$ is the convolution operator, div denotes integer division, and $\Phi(\mathbf{X})$ is a function which linearly maps the values in $\mathbf{X}$ from $[2^{b-1} - 1, 2^b - 1]$ to $[0, 2^b - 1]$. We construct a one-dimensional Max-Tree for each row in $\mathbf{G}_L$ and $\mathbf{G}_R$. We denote the set of constructed Max-Trees based on a row in the left (right) image as M_L (M_R).

2.2.1. Hierarchical disparity prediction

Stereo matching methods typically assume that regions of uniform disparity are likely surrounded by an edge on both sides which is stronger than the gradient within the region [33,35]. We exploit this assumption by matching such regions as a whole. Efficiency can be gained in this way because the pixels in a region of uniform disparity do not need to be matched individually. Another advantage of region based matching is that matching ambiguity of pixels in uniformly colored regions is reduced.

Edges of varying strength exist in images. When all regions with a constant gradient of zero surrounded by an edge are matched, the advantage of this approach is limited because such regions are relatively small in area and large in number. When only regions surrounded by strong edges are matched, the number of regions will be smaller but these regions will contain edges which may correspond to disparity borders. To solve this problem, we match regions surrounded by strong edges first, and then iteratively match regions surrounded by edges of decreasing strength. After two regions are matched with reasonable confidence, only regions within those regions are matched in subsequent iterations, i.e. nodes (n_L, n_R) can be matched when (n_L, n_R) passes Eq. (5). The Max-Tree representation of scan-lines that we used favours efficient hierarchical matching of image regions. Similarly to the multi-scale image segmentation scheme proposed by Todorovic and Ahuja [27], we store the inclusion relation of non-uniformly colored image structures being composed of structures which contain less contrast. We call *top nodes* those nodes in a Max-Tree that correspond to regions surrounded by an edge on both sides which is stronger than the gradient within the region. We categorize a *top node* as a *fine top node* when the gradient within the node is uniform, and as a *coarse top node* when the gradient is not uniform. Let (M_L^r, M_R^r) denote the pair of Max-Trees at row r in the images. We define the set $\phi^0_{M^r}$ of *fine top nodes* in Max-Tree M^r as:

$$\phi^0_{M^r} = \{n \in M^r \mid \theta_\alpha < area(n) < \theta_\beta \wedge \exists!\ n_2 \in M^r : p(n_2) = n\},$$

where $p(n)$ indicates the parent node of n. Consequently, a *fine top node* n corresponds to a tree leave with $\theta_\alpha < area(n) < \theta_\beta$. To increase efficiency, nodes with width smaller than a threshold θ_α or larger than a threshold θ_β are not matched. *Coarse top* nodes can be determined by traversing the ancestors of *fine top nodes*. *Top nodes* with a higher level denote regions surrounded by stronger edges. The level 0 *coarse top nodes* in a Max-Tree M^r denotes its *fine top nodes*. Coarse top nodes at ith level are inductively defined as the nodes which are the parent of at least one $(i-1)$th level *coarse top node*, which do not have a descendant which is also a ith level *coarse top node*. We define the set of *coarse top nodes* at the ith level of the tree M^r as:

$$\phi^i_{M^r} = \{n \in M^r \mid \exists\ n_2 \in \phi^{i-1}_{M^r} : p(n) = n_2 \\ \wedge \exists!\ n_3 \in desc(n) : n_3 \in \phi^i_{M^r}\},$$

where $desc(n)$ denotes the set of descendants of node n.

Edges in images may not be sharp. Hence *coarse top nodes* at level i and $i+1$ of the tree can differ very little. To increase the difference between *coarse top nodes* of subsequent levels, we use the value of the parameter q in Eq. (1). Our method includes parameter $S \in \{\mathbb{N} \cup 0\}^n$, where $n \in \mathbb{N}$. S is a set of *coarse top node* levels. The *coarse top nodes* corresponding to the levels in S are matched from the coarsest to the finest level.

2.3. Matching cost and cost aggregation

We define the cost of matching a pair of nodes ($n_L \in M_L$, $n_R \in M_R$) as a combination of the gradient cost C_{grad} and the node context cost $C_{context}$, which we define in the following.

Gradient. Let $y = row(n_L) = row(n_R)$, $left(n)$ the x-coordinate of the left endpoint of node n and $right(n)$ the x-coordinate of the right endpoint of node n. We define the gradient cost C_{grad} as the sum of the ℓ_1 distance between the gradient vectors at the left and right end points of the nodes:

$$\begin{aligned} C_{grad}(n_L, n_R) = &\ |(\mathbf{I}_L * \mathbf{S}_x)(left(n_L), y) - (\mathbf{I}_R * \mathbf{S}_x)(left(n_R), y)| \\ &+ |(\mathbf{I}_L * \mathbf{S}_x)(right(n_L), y) - (\mathbf{I}_R * \mathbf{S}_x)(right(n_R), y)| \\ &+ |(\mathbf{I}_L * \mathbf{S}_y)(left(n_L), y) - (\mathbf{I}_R * \mathbf{S}_y)(left(n_R), y)| \\ &+ |(\mathbf{I}_L * \mathbf{S}_y)(right(n_L), y) - (\mathbf{I}_R * \mathbf{S}_y)(right(n_R), y)|. \end{aligned} \qquad (2)$$

Node context. Let a_L and a_R be the ancestors of nodes n_L and n_R, respectively. We compute the node context cost $C_{context}$ as the average difference of the area of the nodes in the sub-trees comprised between the nodes n_L and n_R and the root node of their respective Max-Trees:

$$\begin{aligned} C_{context}(n_L, n_R) = &\ \frac{2^b}{min(\#a_L, \#a_R)} \\ &\cdot \sum_{i=0}^{\min(\#a_L, \#a_R)} \left| \frac{area(a_L(i))}{area(a_L(i)) + area(a_R(i))} - 0.5 \right|, \end{aligned} \qquad (3)$$

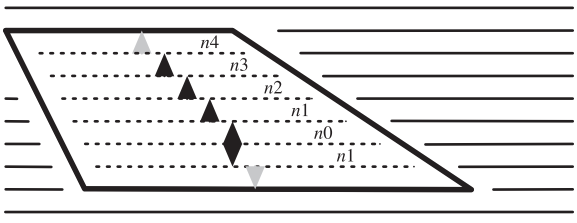


Fig. 3. The edge between uniformly colored foreground and background objects is denoted by a thick line. Thin lines (solid or striped) are coarse top nodes. Dotted lines are coarse top nodes which are a neighbor of n_0. Arrows denote where the presence of a top node is checked. Gray (black) arrows indicate the absence (presence) of a coarse top node.

where b denotes the color depth (in bits) of the stereo image pair, $\#a_L$ and $\#a_R$ indicate the number of ancestor nodes of n_L and n_R, respectively.

We compute the matching cost of a region in the image by aggregating the costs of the nodes in such region and their neighborhood. The neighborhood of node n is a collection (which includes n) of vertically connected nodes that likely have similar disparity. All nodes in this collection are *coarse top nodes* of the same level. We define that n_1 is part of the neighborhood of node n_0 if n_1 crosses the x-coordinate of the center of node n_0, and n_1 has y-coordinate in the image one lower or higher than that of n_0 (i.e. $left(n_1) \le center(n_0) \le right(n_1)$). In an incremental way, node n_{j+1} is part of the neighborhood of n_0 if n_{j+1} crosses the x-coordinate of the center of node n_j, and n_{j+1} has a y-coordinate which is one lower or higher than that of n_j. Note that image gradient constraints which nodes are considered a neighbor of a node. In Fig. 3, we show an example of node neighborhood and illustrate this gradient constraint. At the coordinates of pixels corresponding to an edge (depicted as a thick black line), there is absence of a coarse top node. Therefore, the gray arrows indicate absence of a coarse top node, and the fact that there are no neighbors of n_0 above/below the edge. We use a parameter θ_γ to regulate the size of the neighborhood of a node: the closest θ_γ nodes in terms of y-coordinate are considered in the neighborhood. We use the node neighborhood to enhance vertical consistency for the depth map construction.

Let $N^T_{n_L}$ ($N^B_{n_L}$) denote the vector of neighbours of $n_L \in M_L$ above (or below) n_L, and $N^T_{n_R}$ ($N^B_{n_R}$) the vector of neighbours of $n_R \in M_R$ above (or below) n_R. Let $N(i)$ denote the i-th element in N. Both in N^B and N^T the distance between $N(i)$ and n increases as i is increased, therefore $N(0) = n$. We define the aggregated cost of matching the node pair (n_L, n_R) as:

$$C(n_L, n_R) = \sum_{s=\{T,B\}} \left(\frac{1}{\min(\#N^s_{n_L}, \#N^s_{n_R})} \sum_{i=0}^{\min(\#N^s_{n_L}, \#N^s_{n_R})} \times \left(\alpha \; C_{grad}\left(N^s_{n_L}(i), N^s_{n_R}(i)\right) \; + (1-\alpha) \; C_{context}\left(N^s_{n_L}(i), N^s_{n_R}(i)\right) \right) \right), \tag{4}$$

where $0 \le \alpha \le 1$ controls the weight of individual costs.

2.4. Disparity search range determination

Our method considers the full disparity search range during the matching of *coarse top nodes* in the first iteration. In subsequent iterations, after *coarse top nodes* have been matched with reasonable confidence, only descendants of matched *coarse top nodes* are matched. The disparity of a pair of segments can be derived by calculating the difference in x-coordinate of the left-side endpoints, or by calculating the difference in x-coordinate of the right-side endpoints. To determine the disparity search range of a node, we compute the median disparity in the neighborhood of the ancestor of the node matched in the previous iteration on both sides resulting in the median disparities d_{left} and d_{right}. At most θ_γ nodes above and below a node which are part of the node neighborhood, and have been matched to another node are included in the median disparity calculations. A node n_L in the left image is only matched with node n_R in the right image if:

$$\begin{aligned} &left(n_R) \le left(n_L) \wedge right(n_R) \le right(n_L) \\ &\quad \wedge left(ctn(n_L)) - d_{left} \le left(n_R) \le right(ctn(n_L)) - d_{right} \\ &\quad \wedge left(ctn(n_L)) - d_{left} \le right(n_R) \le right(ctn(n_L)) - d_{right}, \end{aligned} \tag{5}$$

where $ctn(n)$ denotes the *coarse top node* ancestor of node n which was matched in the previous iteration. Nodes touching the left or right image border are not matched, as predictions in such regions are not reliable.

After each iteration we perform the left-right consistency check by Weng et al. [30], which detects occlusions and incorrect matches. Given a matching of two pixels, disparity values are only assigned when both pixels have minimal matching cost with each other. Let $match(n)$ denote the node matched to node n. The nodes which pass the left-right consistency check are contained in the set:

$$\{(n_L, n_R) \mid match(n_L) = n_R \wedge match(n_R) = n_L\}. \tag{6}$$

2.5. Disparity refinement and map computation

During the tree matching process, it is not ensured that all *fine top nodes* are correctly matched: some nodes may be incorrectly matched, while others may not be matched due to the left-right consistency check (Eq. (6)). We derive a disparity map from matched node pairs in such a way that a disparity value is assigned in the majority of regions corresponding to a *fine top node*, and incorrect disparity value assignment is limited. To compute the disparity of a region corresponding to a *fine top node n*, we compute the median disparity at the left and right endpoints (i.e. the difference in x-coordinate of the same-side endpoints of matched nodes) in the neighborhood of n. At most, the θ_γ nodes above and θ_γ nodes below n that are already matched to another node are included in the median disparity calculation. The output of our method can be a semi-dense or sparse disparity map. We generate semi-dense disparity maps by assigning the minimum of said left and right side median disparities to all the pixels of the region corresponding to the node, while for sparse disparity maps the left (right) side median disparity is assigned at the left (right) endpoint only.

When a sparse disparity map is created, we remove disparity map outliers in an additional refinement step. Let $d(x, y)$ denote a disparity map pixel. We set $d(x, y)$ as invalid when it is an outlier in local neighbourhood $ln(x, y) = \{(c, r) \mid valid(d(c, r)) \wedge (x - 21) \le c < (x + 21) \wedge (y - 21) \le r < (y + 21)\}$ consisting of valid (i.e. having been assigned a disparity value) pixel coordinates. We define the set of pixels in $ln(x, y)$ similar to $d(x, y)$ as $sim(x, y) = \left\{(c, r) \in \mathrm{ln}(x, y) \;\middle|\; |d(c, r) - d(x, y)| \le \theta_\omega\right\}$. We define the outlier filter as

$$d(x, y) = \begin{cases} d(x, y) & \textbf{if} \quad \#sim(x, y) \ge \#(ln(x, y) \backslash sim(x, y)) \\ invalid & \textbf{else} \end{cases}.$$

3. Evaluation

3.1. Experimental setup

We carried out experiments on the Middlebury 2014 data set [22], KITTI 2015 data set [15] and the TrimBot2020 3DRMS 2018 data set of synthetic garden images [28]. We evaluate the performance of our algorithm in terms of computational efficiency and accuracy of computed disparity maps.

The Middlebury training data set contains 15 high resolution natural stereo pairs of indoor scenes and ground truth disparity maps. The KITTI 2015 training data set contains 200 natural stereo pairs of outdoor road scenes and ground truth disparity maps. The Trimbot2020 training data set contains 5×4 sets of 100 low-resolution synthetic stereo pairs of outdoor garden scenes with ground truth depth maps. They were rendered from 3D synthetic models of gardens, with different illumination and weather conditions (i.e. clear, cloudy, overcast, sunset and twilight), in the context of the TrimBot2020 project [25]. The (vcam_0, vcam_1) stereo pairs of the Trimbot2020 training data set were used for evaluation.

For the Middlebury and KITTI data sets, we compute the average absolute error in pixels (avgerr) with respect to ground truth disparity maps. Only non-occluded pixels which were assigned a disparity value (i.e. have both been assigned a disparity value by the evaluated method and contain a disparity value in the ground truth) are considered. For the Trimbot2020 data set, we compute the average absolute error in meters (avgerr$_m$) with respect to ground truth depth maps. Only pixels which were assigned a depth value (i.e. have been assigned a depth value by our method and contain a non-zero depth value in the ground truth) are considered. Furthermore, we measure the algorithm processing time in seconds normalized by the number of megapixels (sec/MP) in the input image. We do not resize the original images in the datasets. For all data sets, we compute the average density (i.e. percentage of pixels with a disparity estimation w.r.t. total number of image pixels) of the disparity maps computed by the considered methods (d%). We performed the experiments on an Intel® Core^{TM} i7-2600K CPU running at 3.40GHz with 8GB DDR3 memory. For all the experiments we set the value of the parameters as $q = 5$, $S = \{1, 0\}$, $\theta_\gamma = 6$, $\alpha = 0.8$, $\theta_\alpha = 3$, $\theta_\omega = 3$. For the Middlebury and KITTI data sets, θ_β is 1/3 of the input image width. For the Trimbot2020 data set, θ_β is 1/15 of the input image width.

3.2. Results and comparison

In Fig. 4, we show example images from the Middlebury (a), synthetic TrimBot2020 (e), and KITTI (i,m) data sets, together with their ground truth depth images ((b), (f) and (j,n), respectively). In the third column of Fig. 4, we show the output of our sparse reconstruction approach, while in the fourth column that of the semi-dense reconstruction algorithm. Our semi-dense method makes the assumption that regions with little texture are flat because information can not be extracted from a uniformly colored region which allows to recover its disparity. We observed that the proposed method estimates disparity in texture-less regions with satisfying robustness (e.g. the table top and the chair surface in Fig. 4d). When semi-dense reconstruction is applied, in the case of an object containing a hole, the foreground disparity is sometimes assigned to the background when the background is a texture-less region. This is seen in the semi-dense output shown in Fig. 4h. In what way our method behaves when faced with uniformly colored regions can be altered through parameter θ_β. Due to inherent ambiguity, this parameter should be set based on high level knowledge about the dataset. A dataset containing more (less) objects with a hole that are in front of a uniformly colored background than objects that do not contain a hole but have a uniformly colored region on their surface should use a smaller (larger) θ_β value.

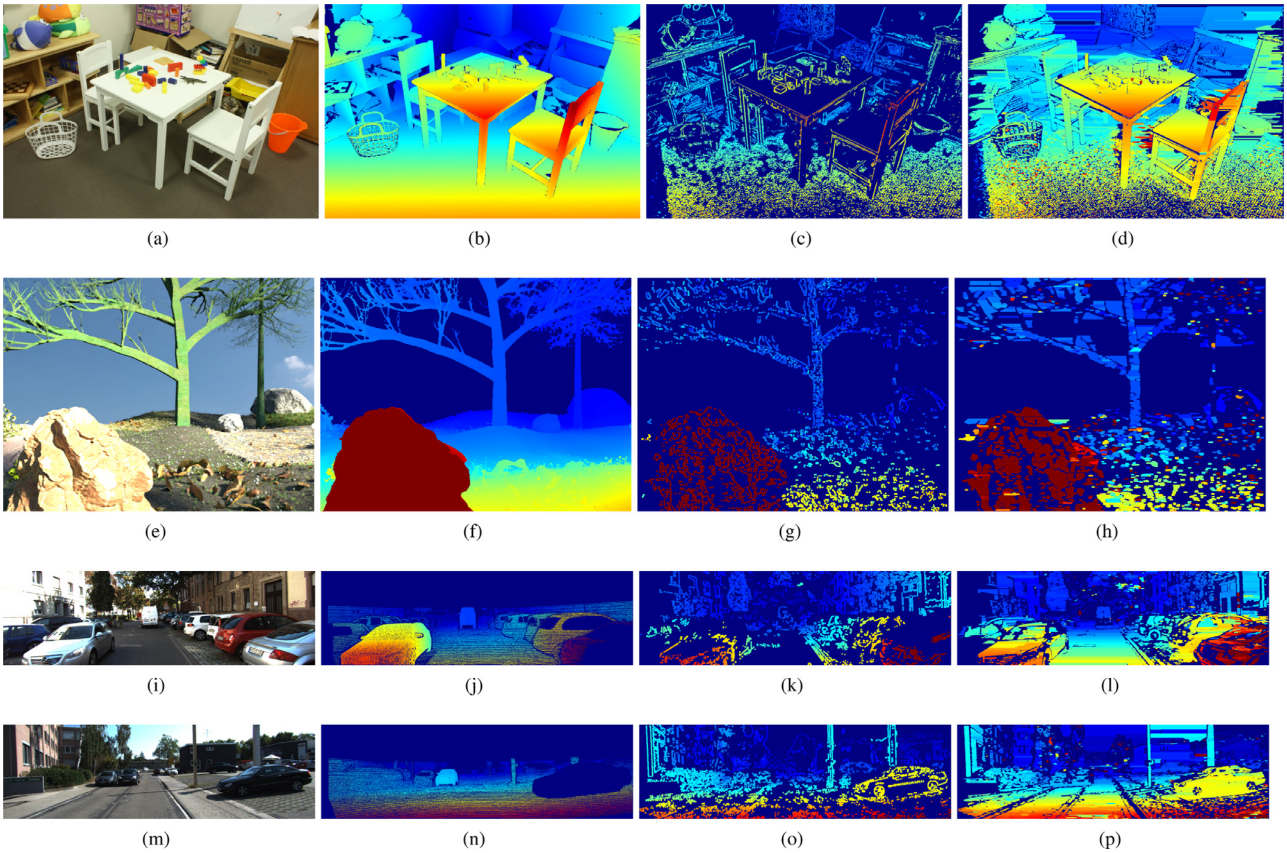


Fig. 4. Example images from the Middlebury (a), TrimBot2020 (e), and KITTI 2015 (i,m) data sets, with corresponding (b,f,j,n) ground truth disparity images. The sparse and semi-dense results are shown in (c,g,k,o) and (d,h,lp), respectively. Morphological dilation was applied to disparity map estimates for visualization purposes only.

Table 1
Comparison of the processing time (sec/MP) achieved on the Middlebury data set. Methods are ordered on avgtime. Our methods are rendered bold.

Method	avgtime	Adiron	ArtL	Jadepl	Motor	MotorE	Piano	PianoL	Pipes	Playrm	Playt	PlaytP	Recye	Shelvs	Teddy	Vintge
r200high	0.01	0.01	0.03	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
MotionStereo	0.09	0.07	0.26	0.08	0.07	0.07	0.07	0.07	0.07	0.08	0.08	0.08	0.07	0.07	0.15	0.07
ELAS_ROB	0.36	0.37	0.34	0.37	0.37	0.37	0.37	0.36	0.34	0.38	0.39	0.37	0.36	0.37	0.34	0.37
LS-ELAS	0.50	0.50	0.51	0.48	0.52	0.50	0.49	0.47	0.48	0.49	0.50	0.48	0.49	0.51	0.51	0.50
Semi-Dense	0.52	0.33	0.41	0.43	0.46	0.49	0.45	0.47	0.57	0.35	1	0.92	0.92	0.33	0.27	0.44
SED	0.52	0.48	0.40	0.72	0.62	0.62	0.58	0.53	0.64	0.54	0.46	0.34	0.43	0.34	0.48	0.57
Sparse	0.54	0.37	0.47	0.49	0.51	0.48	0.44	0.43	0.58	0.36	1	0.92	0.92	0.36	0.27	0.5
ELAS	0.56	0.54	0.49	0.61	0.57	0.57	0.54	0.56	0.55	0.58	0.64	0.57	0.59	0.54	0.51	0.57
SGBM1	0.56	0.61	0.46	0.89	0.52	0.52	0.51	0.50	0.52	0.60	0.51	0.51	0.52	0.46	0.46	1.03
SNCC	0.77	0.72	0.62	1.27	0.71	0.74	0.60	0.60	0.75	0.81	0.71	0.72	0.68	0.64	0.62	1.49
SGBM2	0.91	0.84	0.74	1.55	0.82	0.82	0.82	0.82	0.82	1.03	0.85	0.82	0.83	0.74	0.74	1.81
Glstereo	0.98	0.90	1.17	1.40	0.84	0.84	0.84	1.01	0.90	0.96	0.93	0.92	0.84	0.78	0.92	1.53

Table 2
Comparison of the average error achieved on the Middlebury data set. Methods are ordered on avgerr. Our methods are rendered bold.

Method	avgerr	d%	Adiron	ArtL	Jadepl	Motor	MotorE	Piano	PianoL	Pipes	Playrm	Playt	PlaytP	Recye	Shelvs	Teddy	Vintge
MotionStereo	1.25	48	0.95	1.48	1.69	1.15	1.09	0.90	0.95	1.27	1.30	4.61	0.90	0.70	1.77	0.77	0.90
SNCC	2.44	64	1.95	1.96	4.28	1.51	1.38	1.07	1.24	2.05	2.17	17.9	1.55	1.06	2.75	0.89	1.40
Sparse	3.17	2	2.31	3.65	4.53	2.36	4.07	1.88	7.19	3.88	3.23	3.87	1.5	3.65	2.84	1.24	3.95
LS-ELAS	3.30	61	3.26	1.66	5.58	2.22	2.08	2.65	4.42	2.11	3.41	8.34	1.64	3.03	6.55	1.16	8.98
ELAS	3.71	73	3.92	1.65	7.38	1.80	2.21	3.63	6.07	2.70	3.44	5.50	2.05	4.44	10.1	1.74	4.57
SED	3.82	2	4.51	5.28	5.88	4.22	3.97	2.54	5.26	4.20	3.72	3.53	2.78	4.23	3.40	1.35	1.75
SGBM2	4.97	83	2.90	6.37	11.7	2.54	6.26	3.59	13.0	4.55	4.03	3.24	2.63	2.07	8.32	2.30	5.76
SGBM1	5.35	68	3.56	5.57	12.4	2.78	4.45	5.50	15.5	5.04	4.55	3.55	3.17	2.31	8.35	2.85	6.61
ELAS_ROB	7.19	100	3.09	4.72	29.7	3.28	3.31	4.37	8.46	5.62	6.10	21.8	2.84	3.10	8.94	2.36	9.69
Glstereo	7.36	100	3.33	4.28	36.9	4.48	4.92	2.73	4.67	9.60	5.95	7.19	3.82	3.15	8.63	1.36	8.30
r200high	12.90	23	10.7	11.9	16.0	12.9	10.8	7.29	11.8	5.52	17.3	35.5	11.6	13.3	12.2	7.45	31.7
Semi-Dense	13.8	58	11.3	10.8	34.9	9.3	12.6	9.97	20.4	16.9	12.3	11.7	7.3	18.2	8.31	5.11	18.9

Our approach makes errors in the case of very small repetitive texels which are not surrounded by a strong edge. The sparse stereo reconstruction output shown in Fig. 4g demonstrates the effectiveness of the proposed method on garden images, which contain highly textured regions: disparity is computed for sparse pixels and disparity borders are well-preserved.

We compare our algorithm on the Middlebury (evaluation - version 3) data set directly with those of existing methods that run on low average time/MP and do not use a GPU. These methods are r200high [10], MotionStereo [29], ELAS and ELAS_ROB [7], LS-ELAS [9], SED [18], SGBM1 and SGBM2 [8], SNCC [4] and Glstereo [6]. The reported processing time for these methods, however, was registered on different CPUs than that used for our experiments. Details are reported on the Middlebury benchmark website.[1]

In Tables 1 and 2, we report the average processing time and average error (avgerr), respectively, achieved by the proposed sparse and semi-dense methods on the Middlebury data set in comparison with those achieved by existing methods. The methods are listed in the order of the average processing time (average error) in Table 1 (Table 2). We considered in the evaluation the best performing algorithms that run on CPU or embedded systems. We do not aim at comparing with approaches based on deep and convolutional networks that need a GPU to be executed. These methods, indeed, achieve very high accuracy but have large computational requirements which are not usually available on embedded systems, mobile robots or unmanned aerial vehicles. Among existing methods, MotionStereo is the only method that performs better than our approach, while SNCC and ELAS-based methods achieve comparable accuracy-efficiency trade-off. Other approaches, instead, achieve much lower results and efficiency than that of our algorithm. The average error of our semi-dense method is relatively higher than that of the sparse version. This is mostly caused by the assignment of a single disparity value to entire fine top nodes. By design, the disparity values in-between the endpoints of *fine top nodes* are frequently in error, although not by large margin. Our Semi-Dense method generates disparity maps with competitive density. Our sparse method generates, by design, highly accurate disparity maps with a density that is sufficient for many applications.

Table 3
Processing time (sec/MP) of our method on the Trimbot2020 data set.

Method	avgtime	Clear	Cloudy	Overcast	Sunset	Twilight
Semi-Dense	0.33	0.31	0.29	0.33	0.35	0.37
Sparse	0.38	0.35	0.33	0.38	0.40	0.42

Table 4
Average error of our method on the Trimbot2020 data set.

Method	avgerr$_m$	d%	Clear	Cloudy	Overcast	Sunset	Twilight
Sparse	0.34	2	0.35	0.38	0.39	0.30	0.30
Semi-Dense	0.64	14	0.67	0.70	0.73	0.55	0.54

In Tables 3 and 4 we report the average processing time and average error (avgerr$_m$) that we achieved on the TrimBot2020 synthetic garden data set. The sparse reconstruction version of our method obtains a generally higher accuracy, although it requires a slightly longer processing time than the semi-dense version. The computational requirements of our method do not strictly depend on the resolution of input images as we match top nodes as a whole. This is in contrast with patch-based match methods which make extensive use of sliding-windows. The efficiency gain obtained by our approach is particularly evident for scenes with fewer edges. This is due to the assumption on which our approach is based, i.e. the top nodes represent regions comprised between strong edges.

In Table 5, we report the average error (avgerr), density (d%) and processing time (sec/MP) achieved on the KITTI data set. We compare our algorithm with the methods listed in Tables 1 and 2

[1] http://vision.middlebury.edu/stereo/eval3.

Table 5
Comparison of the average error (avgerr), density (d%) and processing time (sec/MP) achieved on the Kitti2015 data set.

	Semi-dense	Sparse	SGBM1	SGBM2	ELAS_ROB	SED
avgerr	4.4	1.53	1.36	1.20	1.46	1.22
d%	44	2	84	82	99	4
sec/MP	0.36	0.39	1.47	2.45	0.57	1.28

of which an official implementation is publicly available. We used the same parameters of the experiments on the Middlebury data set. Existing methods achieve slightly higher accuracy, while our method achieves competitive results with lower processing time.

3.3. *Resolution independence*

We evaluated the effect of image resolution on the runtime of our methods, compared with that of a patch match method. This method computes a cost volume and aggregates cost using 2D Gaussian blur. To highlight the efficiency of our method, we kept the same blurring kernels although we changed the input image resolution, and no disparity refinement is performed. We resized the images in the Middlebury data set. We measured the unweighted average processing time of our methods and Patch match when given an image with specific width. We used the same set of parameters as for other experiments on the Middlebury data set. The average running time, in seconds, of our semi-dense (sparse) method divided by the running time of the patch match method for the images with a resolution of 2000px to 750px, in steps of 250px was 0.14 (0.16), 0.15 (0.18), 0.17 (0.19), 0.17 (0.2), 0.2 (0.24), 0.26 (0.31).

4. Conclusion

We proposed a stereo matching method based on a Max-Tree representation of stereo image pair scan-lines, which balances efficiency with accuracy. The Max-Tree representation allows us to restrict the disparity search range. We introduced a cost function that considers contextual information of image regions computed on node sub-trees. The results that we achieved on the Middlebury and KITTI benchmark data sets, and on the TrimBot2020 synthetic data set for stereo disparity computation demonstrate the effectiveness of the proposed approach. The low computational load required by the proposed algorithm and its accuracy make it suitable to be deployed on embedded and robotics systems.

Declaration of Competing Interest

On behalf of all authors, Michael H.F. Wilkinson certify that there are no conflicts of interest.

Acknowledgements

This research received support from the EU H2020 programme, TrimBot2020 project (grant no. 688007).

References

[1] D. Chen, M. Ardabilian, X. Wang, L. Chen, An improved non-local cost aggregation method for stereo matching based on color and boundary cue, in: IEEE ICME, 2013, pp. 1–6.

[2] Z. Chen, X. Sun, L. Wang, Y. Yu, C. Huang, A deep visual correspondence embedding model for stereo matching costs, in: IEEE ICCV, 2015, pp. 972–980.

[3] L. Cohen, L. Vinet, P.T. Sander, A. Gagalowicz, Hierarchical region based stereo matching, in: IEEE CVPR, 1989, pp. 416–421.

[4] N. Einecke, J. Eggert, A two-stage correlation method for stereoscopic depth estimation, in: DICTA, IEEE, 2010, pp. 227–234.

[5] J. Engel, J. Stückler, D. Cremers, Large-scale direct slam with stereo cameras, in: IEEE/RSJ IROS, IEEE, 2015, pp. 1935–1942.

[6] Z. Ge., A global stereo matching algorithm with iterative optimization, China CAD & CG 2016 (2016).

[7] A. Geiger, M. Roser, R. Urtasun, Efficient large-scale stereo matching, in: Asian conference on computer vision, Springer, 2010, pp. 25–38.

[8] H. Hirschmuller, Stereo processing by semiglobal matching and mutual information, IEEE Trans. Pattern Anal. Mach. Intell. 30 (2) (2008) 328–341.

[9] R.A. Jellal, M. Lange, B. Wassermann, A. Schilling, A. Zell, Ls-elas: Line segment based efficient large scale stereo matching, in: IEEE ICRA, IEEE, 2017, pp. 146–152.

[10] L. Keselman, J. Iselin Woodfill, A. Grunnet-Jepsen, A. Bhowmik, Intel realsense stereoscopic depth cameras, in: IEEE CVPRW, 2017, pp. 1–10.

[11] W. Luo, A.G. Schwing, R. Urtasun, Efficient deep learning for stereo matching, in: IEEE CVPR, 2016, pp. 5695–5703.

[12] X. Luo, X. Bai, S. Li, H. Lu, S.-i. Kamata, Fast non-local stereo matching based on hierarchical disparity prediction, arXiv preprint arXiv:1509.08197 (2015).

[13] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, T. Brox, A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation, in: IEEE CVPR, 2016, pp. 4040–4048. ArXiv:1512.02134

[14] G. Medioni, R. Nevatia, Segment-based stereo matching, Comput. Vision Graph. Image Process. 31 (1) (1985) 2–18.

[15] M. Menze, C. Heipke, A. Geiger, Joint 3d estimation of vehicles and scene flow, ISPRS Workshop on Image Sequence Analysis (ISA), 2015.

[16] H. Oleynikova, D. Honegger, M. Pollefeys, Reactive avoidance using embedded stereo vision for mav flight, in: IEEE ICRA, IEEE, 2015, pp. 50–56.

[17] H. Park, K.M. Lee, Look wider to match image patches with convolutional neural networks, IEEE Signal Process. Lett. 24 (12) (2017) 1788–1792.

[18] D. Peña, A. Sutherland, Disparity estimation by simultaneous edge drawing, in: ACCV 2016 Workshops, 2017, pp. 124–135.

[19] G. Ros, S. Ramos, M. Granados, A. Bakhtiary, D. Vazquez, A.M. Lopez, Vision-based offline-online perception paradigm for autonomous driving, in: IEEE WCACV, IEEE, 2015, pp. 231–238.

[20] P. Salembier, A. Oliveras, L. Garrido, Antiextensive connected operators for image and sequence processing, IEEE Trans. Image Process. 7 (4) (1998) 555–570.

[21] P. Salembier, M.H.F. Wilkinson, Connected operators, IEEE Signal Process. Mag. 26 (6) (2009) 136–157.

[22] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, P. Westling, High-resolution stereo datasets with subpixel-accurate ground truth, in: GCPR, Springer, 2014, pp. 31–42.

[23] D. Scharstein, R. Szeliski, A taxonomy and evaluation of dense two-frame stereo correspondence algorithms, Int. J. Comput. Vis. 47 (1–3) (2002) 7–42.

[24] S. Sengupta, E. Greveson, A. Shahrokni, P.H. Torr, Urban 3d semantic modelling using stereo vision, in: IEEE ICRA, IEEE, 2013, pp. 580–585.

[25] N. Strisciuglio, R. Tylecek, M. Blaich, N. Petkov, P. Biber, J. Hemming, E. v. Henten, T. Sattler, M. Pollefeys, T. Gevers, T. Brox, R.B. Fisher, Trimbot2020: an outdoor robot for automatic gardening, in: ISR 2018; 50th International Symposium on Robotics, 2018, pp. 1–6.

[26] C. Sun, A fast stereo matching method, in: DICTA, Citeseer, 1997, pp. 95–100.

[27] S. Todorovic, N. Ahuja, Region-based hierarchical image matching, Int. J. Comput. Vis. 78 (1) (2008) 47–66.

[28] R. Tylecek, T. Sattler, H.-A. Le, T. Brox, M. Pollefeys, R.B. Fisher, T. Gevers, The second workshop on 3d reconstruction meets semantics: Challenge results discussion, in: ECCV 2018 Workshops, 2019, pp. 631–644.

[29] J. Valentin, A. Kowdle, J.T. Barron, N. Wadhwa, M. Dzitsiuk, M. Schoenberg, V. Verma, A. Csaszar, E. Turner, I. Dryanovski, et al., Depth from motion for smartphone ar, in: SIGGRAPH Asia, ACM, 2018, p. 193.

[30] J. Weng, N. Ahuja, T.S. Huang, et al., Two-view matching., in: ICCV, 88, 1988, pp. 64–73.

[31] M.H.F. Wilkinson, A fast component-tree algorithm for high dynamic-range images and second generation connectivity, in: IEEE ICIP, 2011, pp. 1021–1024.

[32] X. Ye, J. Li, H. Wang, H. Huang, X. Zhang, Efficient stereo matching leveraging deep local and context information, IEEE Access 5 (2017) 18745–18755.

[33] K.-J. Yoon, I.S. Kweon, Adaptive support-weight approach for correspondence search, IEEE Trans. Pattern Anal. Mach. Intell (4) (2006) 650–656.

[34] J. Zbontar, Y. LeCun, et al., Stereo matching by training a convolutional neural network to compare image patches., J. Mach. Learn. Res. 17 (1–32) (2016) 2.

[35] K. Zhang, J. Lu, G. Lafruit, Cross-based local stereo matching using orthogonal integral images, IEEE Trans. Circuits Syst. Video Technol. 19 (7) (2009) 1073–1079.