

Stereo Matching by Using Self-distributed Segmentation and Massively Parallel GPU Computing

Wenbao Qiao^(✉) and Jean-Charles Créput

IRTES-SET, University of Technology of Belfort-Montbéliard, Belfort, France
rapidbao@gmail.com

Abstract. As an extension of using image segmentation to do stereo matching, firstly, by using self-organizing map (som) and K-means algorithms, this paper provides a self-distributed segmentation method that allocates segments according to image's texture changement where in most cases depth discontinuities appear. Then, for stereo, under the fact that the segmentation of left image is not exactly same with the segmentation of right image, we provide a matching strategy that matches segments of left image to pixels of right image as well as taking advantage of border information from these segments. Also, to help detect occluded regions, an improved aggregation cost that considers neighbor valid segments and their matching characteristics is provided. For post processing, a gradient border based median filter that considers the closest adjacent valid disparity values instead of all pixels' disparity values within a rectangle window is provided. As we focus on real-time execution, these time-consuming works for segmentation and stereo matching are executed on a massively parallel cellular matrix GPU computing model. Finally, we provide our visual dense disparity maps before post processing and final evaluation of sparse results after post-processing to allow comparison with several ranking methods top listed on Middlebury.

Keywords: Stereo · Image segmentation · SOM · Self-distributed segments

1 Introduction

Stereo matching from two 2D images is a process to estimate 3D scene but with acceptable error under the fact that we can not get really true disparity values for every pixel. Many solutions have been proposed for stereo to meet different requirements in various artificial intelligence research. These solutions can be categorized into four basic optimization approaches: local, global, cooperative and semi-global [1], among which the local methods are usually applied to real-time applications. Generally, local stereo methods work base on four steps: matching cost computation, cost aggregation, disparity computation and disparity refinement [2]. The cost aggregation step plays a more important influence on obtaining good results. For example, classical fixed window aggregation method using

matching cost such as color/grey absolute difference (AD) or mutual information [3] all suffers poor performance at depth discontinuities and in texture-less areas [4], leading to consequences like foreground fattening effect.

Researchers believe image segments can contribute border information to help distinguish background from foreground. Normally, these segments are small over-segments [5, 6] under the assumption that disparity values of pixels within the same segment are same or vary smoothly. Considering the same matching strategy, even though there are few side-effects at curved surface regions compared with pixel-based dense matching method, the advantages of segment-based techniques are well known: lessening computational complexity, enhancing noise tolerance [5] and border distinguishment.

Aiming at real-time execution, this paper proposes to use self-distributed image segmentation as main contribution to a newly proposed cost aggregation method. And because we train initial segment centers according to texture change, our self-distributed segmentation method maybe an interesting solution to the difficulty cited by Yoon and Kweon [7]: the segmentation-based stereo methods require precise color segmentation that is difficult when dealing with highly textured images [7]. Our stereo's cost aggregation strategies are inspired by another real-time algorithm listed in top 20 methods on Middlebury [8], wrote by Hirschmüller et al. [9]. Hirschmüller uses selected multiple rectangular windows to compute Sum of Absolute Differences (SAD) for each pixel and applies only two optimization steps (error disparity filter and border correction) to get state-of-art results. As far as we can see, the multiple small windows which Hirschmüller used can be substituted by the segments produced by our algorithm, because his small windows may across depth discontinuity areas and introduce errors [9], but most of our segments will not. And through using segments, we lessen the most heavy computation part of [9] to get acceptable stereo matching quality for a local stereo method with little complexity.

In most cases, stereo matching processes for each matching unit is same, that is why people can use CUDA programming [10] to do parallel computing instead of calculating one unit after another sequentially by CPU. However, that is not new for stereo, here we not only decompose the input data into parts as usual but also use a massive cellular matrix model to execute neural networks with jobs like: training data, spiral search [11], zooming in/out at different levels of an original image, stereo matching, in a parallel and non-conflict way.

The rest of this paper is organized as follows. Section 2 presents previous work about different strategies using various image segmentation methods for stereo. Section 3 provides our proposed approaches including brief introduction of our cellular matrix model, how we get self-distributed segmentation and the way we use these segments to do stereo matching. Section 4 includes results visualization and discussion.

2 Previous Work

Various image segmentation methods have been proposed for stereo matching and researchers use different strategies to contribute segments to stereo.

Probabilistic-based methods such as Zitnick and Kang [5] use iterative K-means algorithm to produce color-based segmentation, which evolve to be SLIC [12] later. Then a Markov Random Field(MRF) is constructed by using segments as probability events. Through computing all pixels' Sum of Squared Differences(SSD) within one segment, Zitnick constructs that segment's matching probability for the MRF.

Global optimization methods such as Klaus et al. [6] applies Mean shift color segmentation, but the preliminary disparity values before post processing are calculated by computing color and gradient SAD of all pixels within a fixed pixel-level window. Segments are mainly used in post processing steps to refine each pixel's disparity value.

Another bottom-up segmentation method proposed by Bruzzone and Carlin [13] is also used in Xiao et al. [14]. However, Xiao et al. [14] only computes disparity values of segment's edge line points, counts their disparities and selects the most frequently appeared value as disparity for the segment region [14]. In fact, the edge line points used in [14] may turn out to belong to a slanting surface or curved surface whose disparity values vary hugely from center pixels to edge line pixels.

State-of-art local stereo methods such as Gerrits and Bekaert [15] additionally considers image segmentation information within rectangular fixed window support region. The author segments the reference image and considers all window pixels outside the image segment that contains the pixel under consideration as outliers and greatly reduces their weight [15]. Tombari et al. [16] does similarly with Gerrits, the differences are that Tombari segments both reference and target images and introduces a modified weight function similar to adaptive weight method presented by Yoon and Kweon [7].

3 Proposed Approach

3.1 Overview

Under the fact that left and right images' segmentation maps are not exactly same and problems such as big segments may represent slanting or curved surfaces, it can bring matching errors when we only consider the border information extracted from segments. To tackle these problems when using segmentation to do stereo matching, we propose following two parts base on the assumption that objects' surfaces are piece-wise smooth like people treat a circle as a polygon with n sides:

- An image self-distributed over-segmentation method that allocates segments according to texture density or color change where depth discontinuities usually appear.
- A stereo cost aggregation strategy that not only uses segments as matching units but also involves neighboring segments' matching characteristic to help detect some occluded regions.

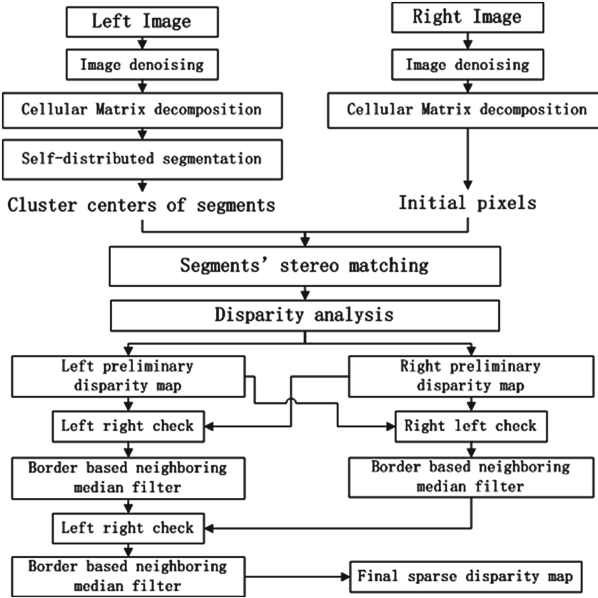


Fig. 1. Overall algorithm flow of our self-distributed segmentation-based stereo algorithm.

Further more, we provide an improved gradient border-based median filter that considers the closest valid disparity values instead of involving all neighboring disparity values within a fixed window. And also, a cellular matrix parallel GPU computing model is provided to execute time-consuming jobs during processes of segmentation and stereo matching. We provide our algorithm's overall flow chart in Fig. 1.

3.2 Self-distributed Segmentation

We provide a cellular matrix model to execute neural networks and do parallel CUDA computation, this model contains following basic concepts:

- A concept of low level topological grid (neural network) with the same size as input image and its node (neuron) contains all attributes of its corresponding original pixel, as shown in Fig. 2 upper (a), here, too small to show details;
- Extracting nodes with a topological radius r on the low level grid, these nodes compose a higher level grids named base level shown in Fig. 2 upper (b, c). Here, each hexagon delimits one cell containing the decomposed low level nodes for GPU to do parallel jobs;
- Neighborhood search operators that can do local spiral search centered by each node within different topology radius of different levels' grid.

To produce self-distributed segments, firstly, we initialize K cluster centers for each final segment. Actually, these cluster centers are same with hexagonal

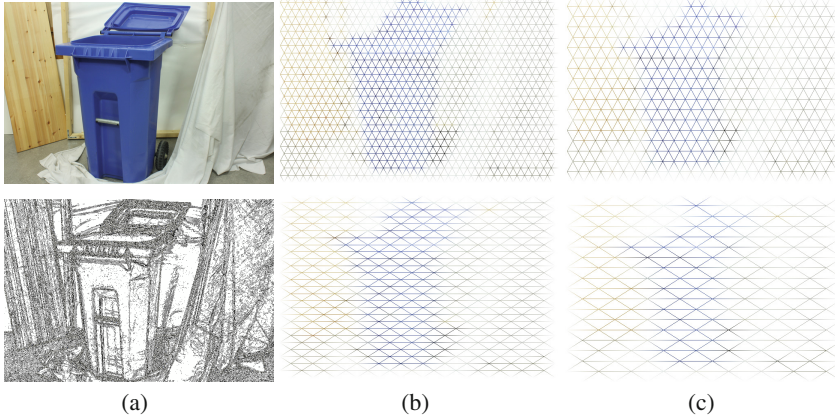


Fig. 2. Cellular matrix model for GPU parallel computation. Up (a) input left image of Recycle dataset from Middlebury [8], each pixel corresponds to one node belonging to the low level hexagonal grid; Up (b) base level with topological radius $r = 20$; Up (c) base level with topological radius $r = 30$; Down (a) gradient information of the input image; Down (b, c) second higher level grid extracted from Up (b, c) separately, namely geometric dual level.

nodes on base level shown in Fig. 2 upper (b, c). And the topological radius r decides the number of nodes, which finally influences stereo matching quality. Then, we use online-SOM [17] to train these initial cluster centers (neurons), aiming at allocating more neurons to areas where exists more color change. Because the cluster centers are on the base level topological grid, the decomposition of these nodes for parallel computation should be on geometric dual level shown in Fig. 2 lower (b, c), in which each dual hexagon can be considered as a cell S_i containing equal number of cluster centers at beginning. And the learning direction of these cluster centers is controlled by randomly determining which cell can do the training step of SOM by using an activity possibility stated in Eq. (1).

$$p_i = \frac{S_i}{\max\{S_1, S_2, \dots, S_{num}\}} \quad (1)$$

Here, p_i is the activity probability of cell i ; S_i is sum of initial gradient values of all the input pixels in cell i ; and num is the quantity of cells.

After that, we slightly “exchange” these cluster centers to mean color position of all its similar color neighbor pixels as K-means does and get cluster center pixels. Here we only concern color difference when choosing the closest neighbor pixels for K-means because of the assumption that depth discontinuities always appear with color change. Then these neighboring pixels are marked belong to that cluster center pixel and we get final self-distributed segmentation shown in Fig. 3(b, d). And the cluster center pixel’s color information can represent all its subordinate pixels’ color information. Comparing Fig. 3(b, d) with Fig. 2 upper (a), we can figure out that the allocation of self-distributed segments

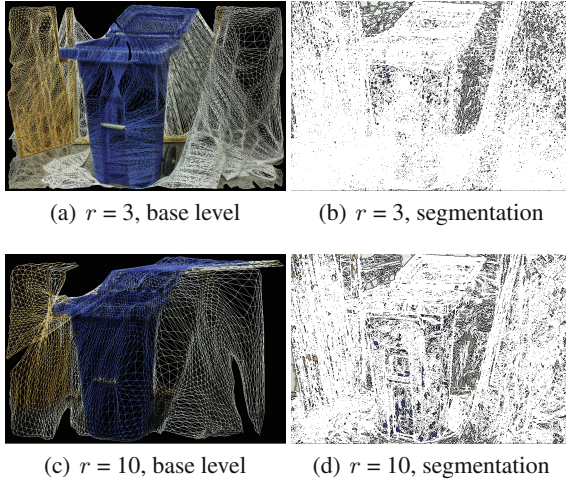


Fig. 3. Self-distributed cluster centers and the final segmentation for stereo matching. (a, c) Online-som results with topological radius $r = 3$ and $r = 10$ separately; (b, d) final self-distributed segmentation, the white lines indicate borders of these segments.

reflects texture change of the input image, where higher color change regions have more and smaller segments while segments in texture-less regions can also support our assumptions.

3.3 Stereo Matching by Using Segments

As we use the cluster center pixels as representatives of the self-distributed segments, the basic strategy that we contribute segments to stereo matching is to match these cluster center pixels of left image to pixels of right image. Of course, image denoising step is indispensable to reduce influence of noise pixels. This strategy works base on the fact that cluster centers pixel has mean color value of its segment and the segment are very small, just like people treat a circle as a polygon with n sides.

Then, only considering left image's segmentation, for each current cluster center pixel p_{iL} which we are dealing with, we spiral search all its neighbor cluster center pixels p_{jL} within another topological radius r_s . Under the assumption that depth discontinuity appears with color discontinuity, here we set an empirical threshold (β) of color AD between p_{iL} and its neighbor cluster center pixel p_{jL} using Eq. (2), to decide n_{jL} valid neighbor segments that can contribute to p_{iL} 's aggregation cost in Eq. (4). We also count the number (n_{jR}) of valid projection pixels P'_{jR} whose color AD with p_{iL} less than the same threshold (β) in the right image, the bigger the difference between n_{jL} and n_{jR} , the smaller chance this candidate disparity value is correct.

$$AD(p_{iL}, p_{jL}) = \begin{bmatrix} R_{p_{iL}} - R_{p_{jL}} \\ G_{p_{iL}} - G_{p_{jL}} \\ B_{p_{iL}} - B_{p_{jL}} \end{bmatrix} \quad (2)$$

Here p_{jL} is the closest neighbor cluster center pixels surrounding p_{iL} in the left image.

$$SAD(p_{iL}, s) = \sum_{p_0}^{p_{n_{jL}}} \left(\begin{bmatrix} R_{P_{jL}}(p_{jx}, p_{jy}) - R_{P'_{jR}}(p_{jx} - s, p_{jy}) \\ G_{P_{jL}}(p_{jx}, p_{jy}) - G_{P'_{jR}}(p_{jx} - s, p_{jy}) \\ B_{P_{jL}}(p_{jx}, p_{jy}) - B_{P'_{jR}}(p_{jx} - s, p_{jy}) \end{bmatrix} \right) \quad (3)$$

Here $P_{jL} = (p_0, p_1 \dots p_{n_{jL}})$ are neighbor valid cluster center pixels of p_{iL} , n_{jL} represents the quantity; s is candidate disparity value within the disparity range given by Middlebury; P'_{jR} are the projection pixels of P_{jL} on the right image with s .

$$C(p_{iL}, s) = \beta * SAD(p_{iL}, s) + (1 - \beta) \left| n_{jL} - n_{j'R} \right| \quad (4)$$

Here, n_{jR} is the quantity of valid neighbor projection pixels P'_{jR} in the right image.

Then, the candidate disparity value that minimize the aggregation cost $C(p_{iL}, s)$ is considered as the preliminary disparity result of p_{iL} . We adapt this disparity value to all the subordinate pixels belong to p_{iL} and get disparity map shown in Fig. 4(c) without post processing steps.

However, only selecting the minimum aggregation cost by Eqs. (2)–(4) can not solve the occluded region marked in Fig. 4(a), because this region corresponds to nothing correctly in the right image. To detect this kind of occluded region, at process of computing Eq. (3), we set another threshold (θ) of color AD between the valid left neighbor cluster center pixels and its candidate projection pixels on the right image, shown in Eq. (5). This means we check the neighbor segments' matching characteristics when we compute the aggregation cost for p_{iL} . Then, our segmentation-based aggregation cost can detect that occluded region as shown in Fig. 4(d).

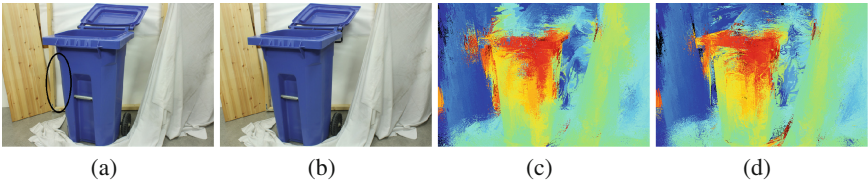


Fig. 4. Preliminary disparity maps with radius parameters $r = 5$ and $r_S = 3$ before post-processing. (a) input left image with one kind of occluded region being marked with black ellipse; (b) input right image; (c) disparity map got by using Eqs. (2)–(4) for the left image; (d) improved disparity map got by adding Eqs. (5) to Eq. (3), the occluded region marked in (a) has been detected.

$$AD(P_{jL}, P'_{jR}) = \begin{bmatrix} R_{P_{jL}}(p_{jx}, p_{jy}) - R_{P'_{jR}}(p_{jx} - s, p_{jy}) \\ G_{P_{jL}}(p_{jx}, p_{jy}) - G_{P'_{jR}}(p_{jx} - s, p_{jy}) \\ B_{P_{jL}}(p_{jx}, p_{jy}) - B_{P'_{jR}}(p_{jx} - s, p_{jy}) \end{bmatrix} \quad (5)$$

Here, P_{jL} are the neighbor valid cluster center pixels surrounding p_{iL} ; P'_{jR} are the candidate correspondences of P_{jL} in the right image with candidate disparity value s .

4 Post Processing and Result Evaluation

As we put emphasis on the quality of preliminary disparity map before post-processing, we make a comparison of our preliminary dense results with three top ranking results after post-processing, shown in Fig. 6. We can discern the smoothness of our results compared with BSM in some regions where disparity values seem correct and the clear distinguishment of foreground with background compared with R-NCC. Though our preliminary results before post processing can not compete with these two methods on Middlebury, our results can be improved by adding many other optimization steps that have been proved by many other researchers.

For example, before post-processing, we borrow idea from Hirschmüller et al. [9] and use his disparity analysis strategy. After left-right cross check (LRC), we use a gradient border-based median filter as post-processing step, which can avoid fuzzy borders that usually happen with normal window-based median filter: we firstly exclude small gradient values from initial gradient image and get the gradient boundaries similar as Fig. 2 down (a); then we spiral search neighboring valid disparities surrounding the bad pixel detected by LRC instead of using fixed rectangle window surrounding that bad pixel; if the searcher touches the gradient boundaries or bigger than a certain searching radius, it stops; also we do not fix these bad pixels locating on gradient boundaries. The final evaluation of our sparse disparity maps (SDSWTA) on Middlebury shows

Set: [training dense](#) [training sparse](#)
 Metric: [bad 0.5](#) [bad 1.0](#) [bad 2.0](#) [bad 4.0](#) [avgerr](#) [rms](#) [A50](#) [A90](#) [A95](#) [A99](#) [time](#) [time/MP](#) [time/GD](#)
 Mask: [nonocc](#) [all](#)
 plot selected show invalid [Reset sort](#)

Date	Name	Res	Avg	bad 0.5 (%)															
				Adiron	Artl	Jadep1	Motor	MotorE	Piano	PianoL	Pipes	Playrm	Playt	PlaytF	Recyc	Shelvs	Teddy	Vintge	
04/08/15	REAF	H	66.6	66.7	63.8	64.6	63.5	62.3	66.1	72.3	61.1	71.7	79.3	64.9	66.1	80.3	62.3	79.8	
03/03/16	SDSWTA	Q	71.4	71.0	82.0	58.4	72.1	71.7	70.9	77.8	69.7	68.7	66.2	68.3	74.8	73.8	70.7	73.6	
01/21/15	TSGO	F	78.1	76.8	69.2	86.2	74.7	76.2	80.6	86.9	74.2	85.7	81.9	73.6	75.6	85.0	76.1	87.9	
08/31/14	BSM	Q	82.2	80.6	79.8	82.9	79.4	79.1	82.3	88.6	80.4	84.5	87.5	81.9	81.5	88.0	84.1	87.6	

Fig. 5. Evaluation of our sparse disparity results after post processing (SDSWTA). This evaluation only concerns the percentage of bad pixel 0.5 without occluded parts of the left image.

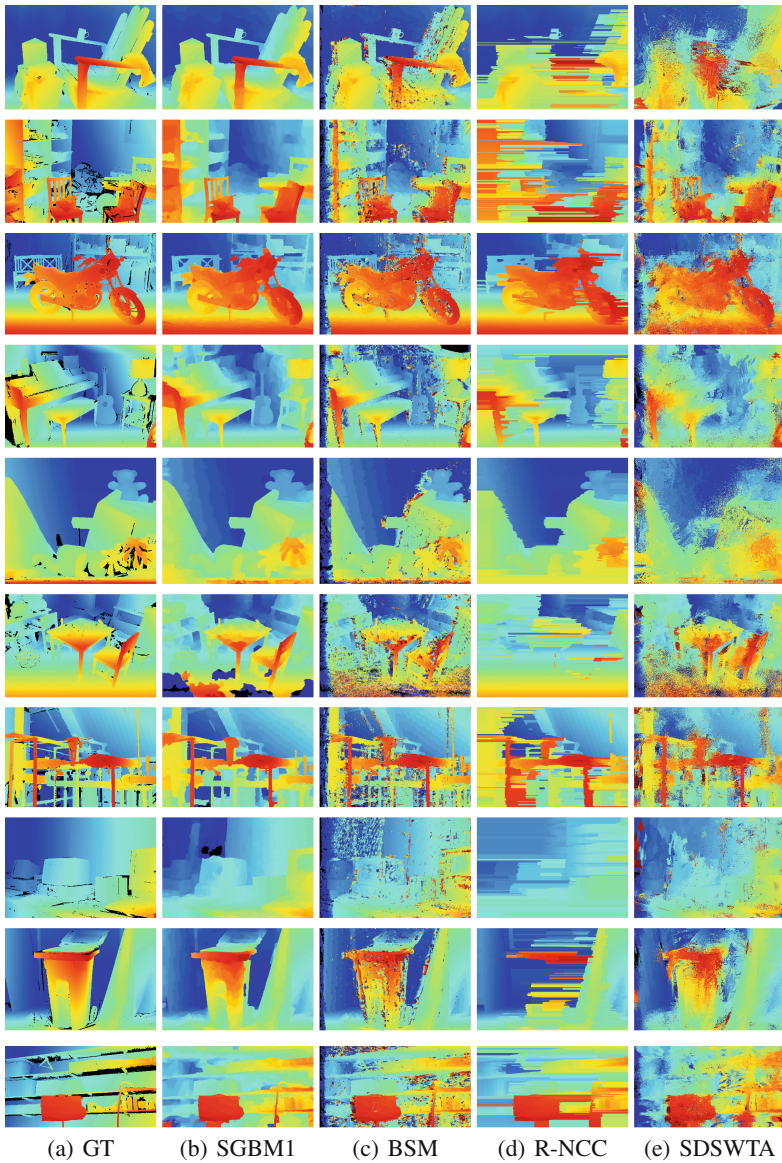


Fig. 6. Comparison of our preliminary dense results before post processing with three top ranking methods listed on Middlebury. (a) Ground Truth; (b) SGBM1; (c) BSM; (d) R-NCC; (e) Our method, SDSWTA.

that our method can reach an interesting accuracy. One of the best evaluations is shown in Fig. 5.

Three parameters in this paper play important roles to the final results: the topological radius r for SOM and r_S for searching neighboring segments, the threshold θ to select valid neighbor correspondences.

5 Conclusion

To give an objective assessment of our methods, we provide a self distributed segmentation method that is suitable for stereo and a matching strategy that adapts to this segmentation. The aggregation cost may not be suitable for pixel-based dense matching method, but we have proved that it is adequate for our segments-based strategy. The reasonable assumption that depth discontinuity happens with color change has its drawbacks at cases where the color of background is very close to the foreground's. Just as shown in Fig. 4(c, d), matching errors appear near the borders of white wall and white drape in Fig. 4(a, b). This is a challenging problem for many stereo algorithms.

Acknowledgments. This paper is sponsored by China Scholarship Council(CSC) and laboratory IRTES-SET of UTBM.

References

1. Cigla, C., Alatan, A.A.: Information permeability for stereo matching. *Sig. Process. Image Commun.* **28**(9), 1072–1088 (2013)
2. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comput. Vis.* **47**(1–3), 7–42 (2002)
3. Egnal, G.: Mutual information as a stereo correspondence measure. Technical reports (CIS), p. 113 (2000)
4. Kim, J., Kolmogorov, V., Zabih, R.: Visual correspondence using energy minimization and mutual information. In: *Proceedings of the Ninth IEEE International Conference on Computer Vision*, pp. 1033–1040. IEEE (2003)
5. Zitnick, C.L., Kang, S.B.: Stereo for image-based rendering using image over-segmentation. *Int. J. Comput. Vis.* **75**(1), 49–65 (2007)
6. Klaus, A., Sormann, M., Karner, K.: Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In: *18th International Conference on Pattern Recognition, ICPR 2006*, vol. 3, pp. 15–18. IEEE (2006)
7. Yoon, K.-J., Kweon, I.S.: Adaptive support-weight approach for correspondence search. *IEEE Trans. Pattern Anal. Mach. Intell.* **4**, 650–656 (2006)
8. Scharstein, D., Hirschmüller, H., Kitajima, Y., Krathwohl, G., Nešić, N., Wang, X., Westling, P.: High-resolution stereo datasets with subpixel-accurate ground truth. In: Jiang, X., Hornegger, J., Koch, R. (eds.) *GCPR 2014*. LNCS, vol. 8753, pp. 31–42. Springer, Heidelberg (2014)
9. Hirschmüller, H., Innocent, P.R., Garibaldi, J.: Real-time correlation-based stereo vision with reduced border errors. *Int. J. Comput. Vis.* **47**(1–3), 229–246 (2002)
10. NVIDIA: CUDA C Programming Guide 4.2, CURAND Library, Profiler User's Guide (2012). <http://docs.nvidia.com/cuda>

11. Bentley, J.L., Weide, B.W., Yao, A.C.: Optimal expected-time algorithms for closest point problems. *ACM Trans. Math. Softw. (TOMS)* **6**(4), 563–580 (1980)
12. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Susstrunk, S.: Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(11), 2274–2282 (2012)
13. Bruzzone, L., Carlin, L.: A multilevel context-based system for classification of very high spatial resolution images. *IEEE Trans. Geosci. Remote Sens.* **44**(9), 2587–2600 (2006)
14. Xiao, J., Xia, L., Lin, L.: Segment-based stereo matching using edge dynamic programming. In: 2010 3rd International Congress on Image and Signal Processing (CISP), vol. 4, pp. 1676–1679. IEEE (2010)
15. Gerrits, M., Bekaert, P.: Local stereo matching with segmentation-based outlier rejection. In: The 3rd Canadian Conference on Computer and Robot Vision, 2006, p. 66. IEEE (2006)
16. Tombari, F., Mattoccia, S., Di Stefano, L.: Segmentation-based adaptive support for accurate stereo correspondence. In: Mery, D., Rueda, L. (eds.) *PSIVT 2007*. LNCS, vol. 4872, pp. 427–438. Springer, Heidelberg (2007)
17. Kohonen, T.: The self-organizing map. *Proc. IEEE* **78**(9), 1464–1480 (1990)