COMMUNICATIONS IN SCIENCE AND TECHNOLOGY
Homepage: cst.kipmi.or.id

Modified adaptive support weight for stereo matching

Etik Irijanti*

Department of Information Technology, Faculty of Engineering, Universitas Muhammadiyah Yogyakarta, 55183, Indonesia

Article history:
Received: 1 April 2017 / Received in revised form: 15 May 2017 / Accepted: 27 May 2017

Abstract

Stereo matching using local algorithms are incredibly popular in the last years. The adaptive support weight algorithms can give high accuracy results such as global methods. This paper proposes a support aggregation approach for stereo matching that computes support weight in sparse support window mask. The improvement from the previous work is that the new support weight can reduce the computation time. At the end of the research, the result shows that the computation time decreases to approximately half to a quarter of the earlier work time without significant difference of bad pixel percentage and help to reach the optimum correspondence. It means sparse support weight affects the computation time that is needed in stereo matching and optimizes the disparity. This support weight isused to accomplish the stereo matching evaluation using this method. This approach is more reliable than the previous approach in the real implementation.

Keywords: Disparity; local stereo matching; stereo correspondence; adaptive support weight.

1. Introduction

Research in stereo vision has been an issue for several years, and it has again become very popular in recent years. In the stereo vision, the description of the stereo estimation difficulty is the method of evaluating a disparity map of two or more images of the scene. Numerous algorithms have been produced to estimate disparity, and the researcher has given a classification and evaluation for these algorithms in [1], which is presented based on disparity optimization, matching cost, and refinement phases of disparity. The main characteristic of a method typically based on optimization approach, that is the second step, therefore the expert classified it as local, semi-global, global and cooperative [2].

In local approaches [1-4], "winner-take-all" optimization is used to reach disparity map, by evaluating each candidate separately. The aggregation of matching cost through averaging or summation over a support area is calculated; therefore the disparity giving smallest cost is appointed as the corresponding pixel. The local method optimizations are simpler compared to the global optimization algorithms for example graph cuts, belief propagation [5,6], that are more complicated and more accurate. Researchers developed these methods in an energy optimization of the whole, and the aim is to minimize the estimated stereo correspondence energy. The semi-global algorithms [1-3], such as dynamic programming optimization are afforded to reduce computational complication of the global methods. Because there is a view, that global optimization is conducted for every scan line (row) individually in a polynomial time.

Other researchers combined benefits of global and local methods of managing occlusions, the boundaries of the object where un-textured areas and discontinuities of depth observed, using cooperative methods [7,8]. These methods, like area based global methods, depend on the supposition that scenes are consist of non-overlapping planar pieces all of which match to pixel clusters relating color-wise similar pixels.

The primary constraints of stereo matching algorithms need a speed of evaluation time, low memory necessity, and robust disparity map, particularly in electronics application devices. Furthermore, several applications require stereo estimation methods must provide less computational complexity, provide less precision loss and less memory in the extracted 3D models for robotic applications, surveillance systems, and the future generation 3D TVs [14-15]. In that behavior, the strongest applicants for faster disparity evaluations are local window-based approaches [5, 15-18]. The local methods normally do not occupy iterative works to give fast and simplicity executions; because they do not consume full cost volume and compared to other approaches, they need lower memory. Therefore, these methods are suitable for real-time applications on appropriate conditions. Yoon introduced adaptive support weight stereo matching (ADSW) in 2006 [3], with outperform result compared to traditional local methods. Because of its advantages, many stereo-matching algorithms are developed based on the ADSW [19-20].

This work proposes a development of a fast computational stereo estimation method based on ADSW aggregation. The aim is to optimize support weight computation in adaptive support-weight methods. The support weight is evaluated faster using sparse support window mask. The proposed method provides further evaluation time reduction. This work has a contribution to the stereo matching improvement,

* Corresponding author.
Email: etik.irijanti@ft.umy.ac.id.

particularly in speed up the computation.

2. Methodology

2.1. Stereo Images Data Source

This work used stereo images obtained from Middlebury stereo images test set which can be downloaded from (http://vision.middlebury.edu/stereo/data/). The images used are 'cones', 'teddy', 'venus', 'books' ,'sawtooth', 'art', 'laundry', and 'computer'. The size of images are approximately about 370×463 pixels and 383×434 pixels.

2.2. Overview of Algorithm

The method is consist of several main steps which are determining the different distance weighting algorithm, cost aggregation evaluation, disparity selection for the left to right and right to left, and right to left check for cross checking the both results.

Gaussian function of the color distance is calculated among the midpoint pixel c of the window and a pixel i in the support window to define the original color weight. The pixel i regarding to pixel c has color weight w_{ic},

$$w_{i,c} = \exp\left(\frac{\Delta C_{i,c}}{\gamma_c}\right) \quad (1)$$

where the color distance value among two pixels, c and i is a constant. This weight allows the pixel with a similar color to the midpoint pixel. They will give more effect on the last matching cost. In this paper, the color space used is HSV instead of Lab color space that assigned in ADSW. The color distance is calculated utilizing Manhattan rather than Euclidean color distance to minimize calculation.

The mask size of support window for weighting is maintained as large and as symmetric as possible. The support window mask uses only the alternate pixels in each row and each column of the support weights window, as shown in Fig. 1 for 7 x 7 mask.

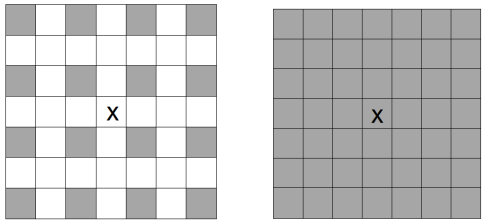

Fig. 1. Support weight window mask

The shaded squares are the pixels used in the support weight calculation in sparse and full mask. This paper shows that the sparse mask can reduced processing time and get similar result compared to the full mask. The difference between pixel x and $\bar{x}_d$, $E(x,\bar{x}_d)$, is expressed as

$$E(x,\bar{x}_d) = \frac{\sum_{y\in N_x,\bar{y}_d\in N_{xd}} w(x,y)w(\bar{x}_d,\bar{y}_d)e(y,\bar{y}_d)}{\sum_{y\in N_x,\bar{y}_d\in N_{xd}} w(x,y)w(\bar{x}_d,\bar{y}_d)} \quad (2)$$

x and y in the reference image have a disparity value d, $\bar{x}_d$ and $\bar{y}_d$ are the corresponding pixels in the target image. $e(y,\bar{y}_d)$ expresses the raw matching cost based on pixel evaluated by the colors of y and $\bar{y}_d$. When using the truncated AD (absolute difference), it is expressed as

$$e(y,\bar{y}_d) = \min\left\{\sum_{c\in\{r,g,b\}} |I_c(y) - I_c(\bar{y}_d)|, T\right\} \quad (3)$$

where I_c is the color group intensity of c and T represents the truncation limit number for controlling the matching cost. After the difference calculation, the every pixel disparity is just selected by using the Winner-Takes-All (WTA) method with no any global calculation as

$$d_x = \arg\min_{d\in S_d} E(x,\bar{x}_d) \quad (4)$$

where s_d = $\{d_{min},\ldots,d_{max}\}$ is the set of whole probable disparities.

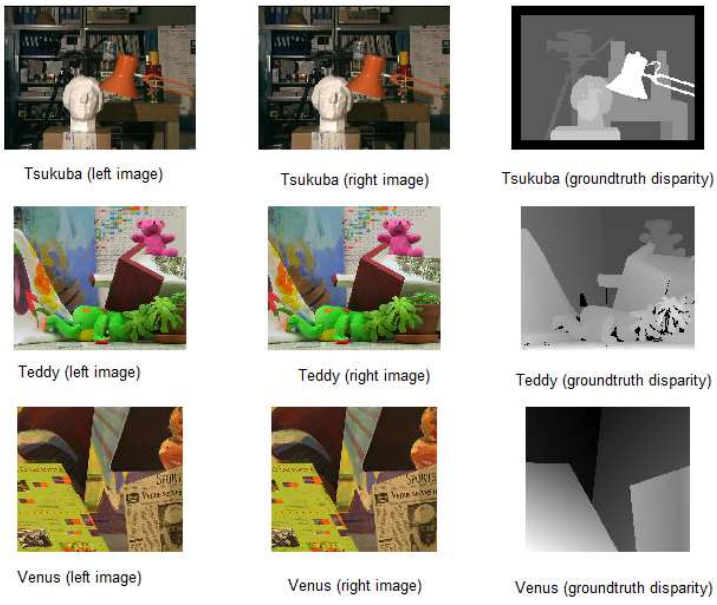

Fig. 2. Tsukuba, Teddy and Venus stereo images: Left image, right Image and ground truth

3. Results and Discussion

This paper used the Tsukuba, Teddy, and Venus as the example for tested images. Fig. 2 shows the images: (a) left image, (b) right image and (c) ground truth. Fig. 3 shows the disparity result using both full support weight (a) and sparse support weight windows for Tsukuba. The size of full support windows in Fig 4 (a) 25 × 25 full support weight windows, computed from left to right, (b) 25 × 25 full support weight windows, computed from right to left (c) 25 × 25 sparse support weight windows, computed from left to right, (d) 25 × 25 sparse support weight windows, computed from right to left, T = 40 for Tsukuba.

The full support weight needs more calculation compared to a sparse window because it calculates whole pixels in the support windows. Support weight computation for n × n full support windows size needs evaluation of n × n pixels, however in sparse support window needs [(n-1)/2 + 1] × [(n-1)/2 + 1] pixels. Regarding to weight formula, it will reduce many calculations, because each pixel will be calculated. Computation time also decreases because the evaluated pixels number is also reduced as in weight computation. Both calculation indicated that the whole stereo matching process using sparse window mask will give faster result compare to the previous method proposed in [5].

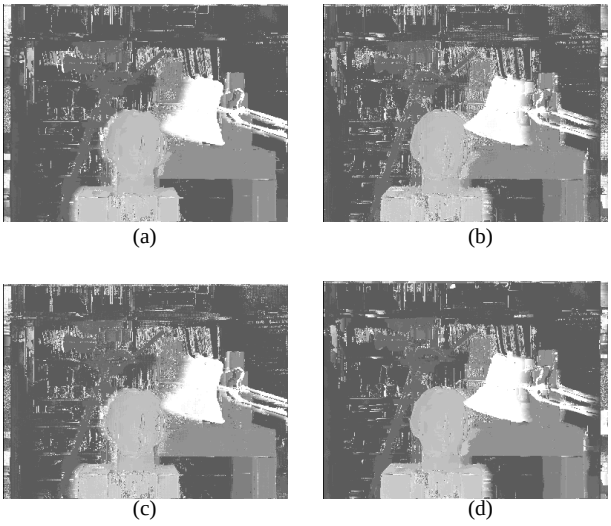

Fig. 3. Dense disparity of Tsukuba using:
(a) 25 x 25 full support weight windows, compute from left to right,
(b) 25 x 25 full support weight windows, compute from right to left
(c) 25 x 25 sparse support weight windows, compute from left to right,
(d) 25 x 25 sparse support weight windows, compute from right to left.

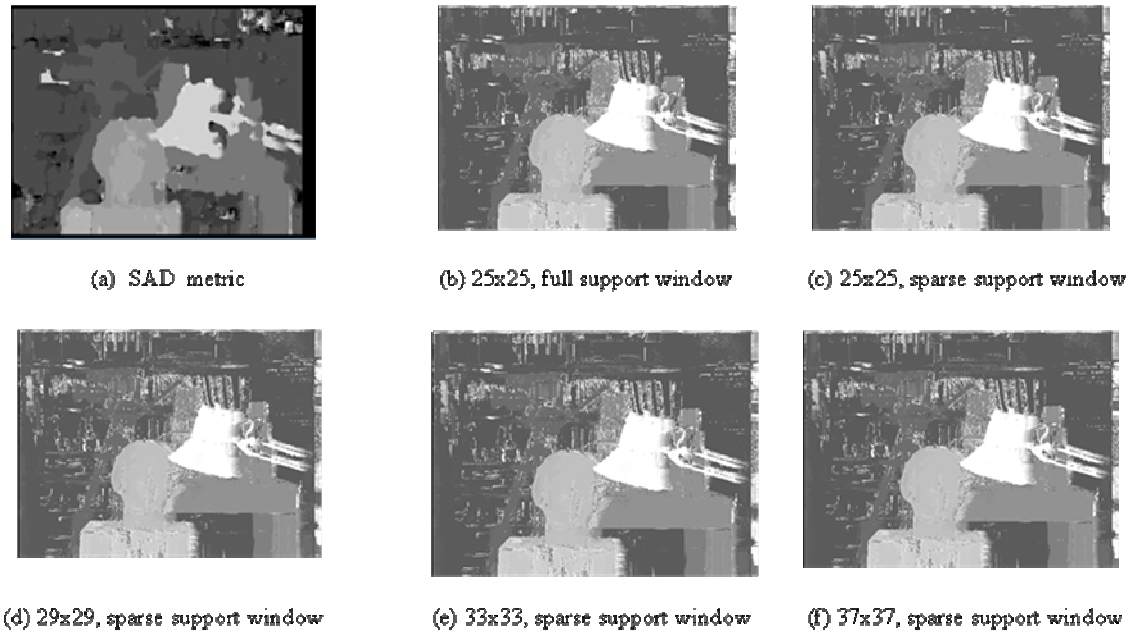

Fig. 4. Dense disparity of Tsukuba using: (a) SAD metric, (b) 25 x 25 full support weight windows (c) 25 x 25 sparse support weight windows, (d) 29 x 29 sparse support weight windows, (e) 33 x 33 sparse support weight windows, (f) 37 x 37 sparse support weight windows

Table 1. Execution time and error for disparity computation

Windows size and support	Execution Time (s)	Bad pixels (%)
25 × 25, full support window	8081.5	30.86
29 × 29, full support window	9404.2	29.56
25 × 25, sparse support window	2428.2	34.14
29 × 29, sparse support window	3750.3	19.52
33 × 33, sparse support window	4850.5	18.56
37 × 37, sparse support window	6587.1	17.88

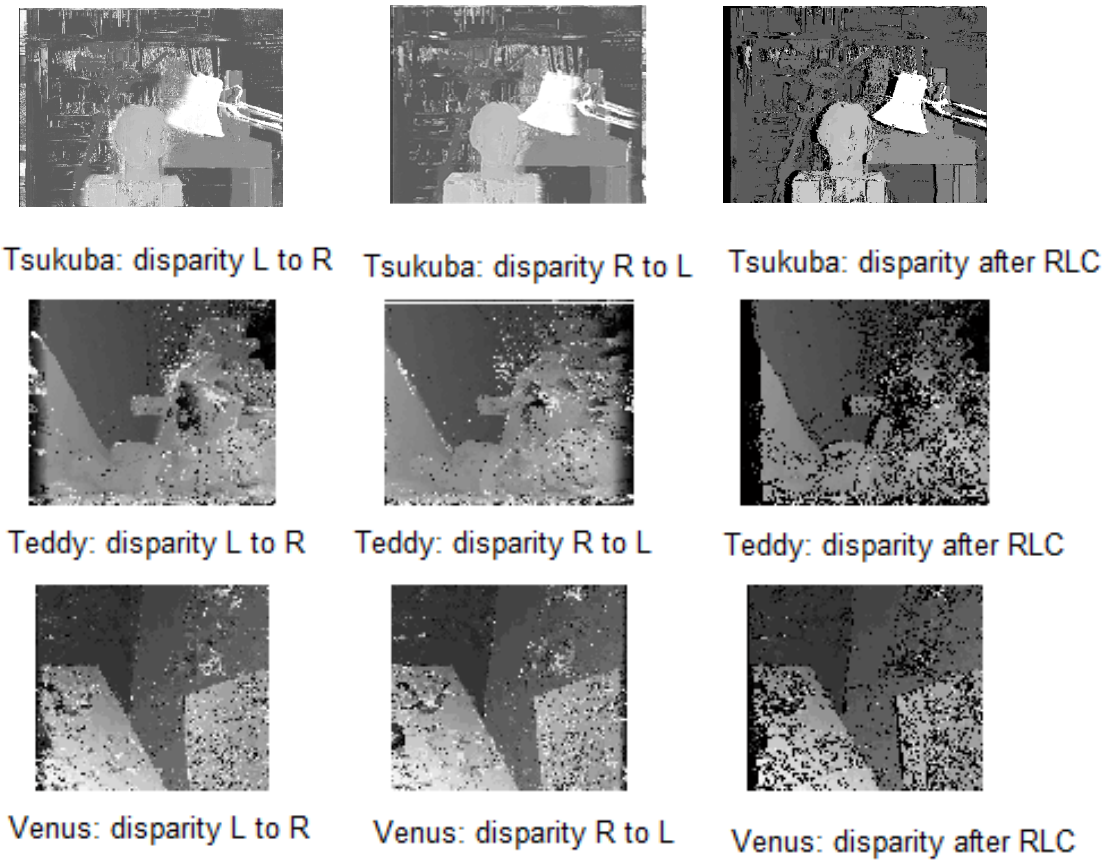

Fig. 5. Disparity map left to right (L to R), right to left (R to L) and disparity after right to left check (RLC) of Tsukuba, Teddy and Venus

Furthermore, the matching process is performed twice: at first with the left image kept as a reference; then the process is reversed, and the right image of a stereo pair constitutes a reference image. Then, the two disparity maps are checked as follows: cross-checking, left–right checking (LRC). Fig. 5. shows the result of disparity using 33x33 spared support windows which are LR, RL, and the RLC disparity. This paper calculated the disparity estimation performance by comparing to the disparity from the ground truth. Table 1. shows execution time and bad pixels for disparity computation from several windows size and the support type. The execution time and bad pixels percentage can be used to consider optimum window size.

The table shows the execution time and error for disparity computation using 25 × 25, 29 × 29, 33 × 33 and 37 × 37 windows size and support. The 25 × 25 and 29 × 29 full support window give 8081.5; 9404.2 seconds execution time; and 30.86; 29.56 error percentage respectively. Furthermore, 25 × 25 and 29 × 29 sparse support window give 2428.2; 3750.3 seconds execution time; and 34.14; 19.52 bad pixels percentage in that order. In the size of 25 × 25 presents 8081.5 seconds calculation time for full support window and 2428.2 seconds evaluation time for sparse support window. The comparison of process time is around 3.3:1, where the error is 1:1.33. That means this approach reduces processing time into fourth without reducing the quality significantly.

The computer cannot perform the stereo matching evaluation to get the result using 33 and 37 window size by full support window method, because of the limitation of computer's memory, so we do not present in Table 1. While if we use sparse support window, the calculation can still be done, even yielding better results, lower bad pixels percentage. The 33 × 33 sparse support window gives 4850.5 seconds execution time and 18.56% bad pixels. The 37 × 37 sparse support window gives 6587.1 seconds execution time and 17.88% bad pixels. It shows the limitation of full support window method that consumes high memory and computation. Disparity estimation process that needs a large number of calculations gives too much time and slow speed execution. Hence, the sparse support window can evaluate the stereo matching with lower error.

The result is still having big bad pixels because occlusion handling and refinement as the post-processing step have not been conducted in this paper. The work aim is to contribute optimum computation for local stereo matching based on support weight aggregation schemes. Disparity estimation has not given expected result yet. Thus, reduction of time execution and quality improvement is needed.

4. Conclusion

This paper has proposed a new support aggregation approach for stereo matching and has computed support weight in sparse support window mask. The improvement from the previous work is that the new support weight mask can reduce computation time while maintain the performance. It means sparse support weight can improve the full mask support weight. The occlusion handling and post-processing

filter can be explored in the next step to make the disparity map close to ground truth and has smaller bad pixels.

References

1. D. Scharstein and R. Szeliski, *A Taxonomy and evaluation of dense two-frame stereo correspondence algorithms.* Int. J. Comput. Vision. 47 (2002) 7-42.
2. C. Cigla and A. A. Alatan, *Information permeability for stereo matching.* Signal Process. Image Commun. 28 (2013) 1072-1088.
3. K.-J. Yoon and I. S. Kweon, *Adaptive support-weight approach for correspondence search.* IEEE Trans. Pattern. Anal. Mach. Intell. 28 (2006) 650-656.
4. L. Wang, M. Gong, M. Gong and R. Yang, *How far can we go with local optimization in real-time stereo matching. 3D Data Processing, Visualization, and Transmission, Third International Symposium on, 2006, pp. 129-136.*
5. E. Irijanti, M. Nayan and M. Yusoff, *Fast stereo correspondence using small-color census transform, Intelligent and Advanced Systems, 4th International Conference on. 2012. pp. 685-690.*
6. E. Irijanti, M. Y. Nayan and M. Z. Yusoff, *Local stereo matching algorithm: Using small-color census and sparse adaptive support weight, National Postgraduate Conference (NPC), 2011. pp 1-5.*
7. P. F. Felzenszwalb and D. P. Huttenlocher, *Efficient belief propagation for early vision, Computer Vision and Pattern Recognition, Proceedings of the IEEE Computer Society Conference on. 2004. pp. 41-54.*
8. M. Bleyer, and M. Gelautz, *Graph-based surface reconstruction from stereo pairs using image segmentation, Electronic Imaging. 2005. pp. 288-299.*
9. T. Hai and H. S. Sawhney, *Global matching criterion and color segmentation based stereo, Applications of Computer Vision, Fifth IEEE Workshop on. 2000. pp. 246-253.*
10. M. Bleyerand and M. Gelautz, *A layered stereo algorithm using image segmentation and global visibility constraints, Image Processing, International Conference on. 2004. pp. 2997-3000.*
11. O. Veksler, *Stereo correspondence by dynamic programming on a tree, Computer Vision and Pattern Recognition, IEEE Computer Society Conference on. 2005. pp. 384-390.*
12. L. Wang, M. Liao, M. Gong, R. Yang and D. Nister, *High-quality real-time stereo using adaptive cost aggregation and dynamic programming, 3D Data Processing, Visualization, and Transmission, Third International Symposium on. 2006. pp. 798-805.*
13. D. Hernandez-Juarez, A. Chacón, A. Espinosa, D. Vázquez, J.C. Moure, and A.M. López. *Embedded real-time stereo estimation via semi-global matching on the GPU. Procedia Computer Science, 2016. pp. 143-153.*
14. Y.-H. Seo, J.-S. Yoo and D.-W. Kim, *A new parallel hardware architecture for high-performance stereo matching calculation.* Integration, the VLSI Journal, 51 (2015) 81-91.
15. D. Zha, X. Jin, and T. Xiang, *A real-time global stereo-matching on FPGA.* Microprocessors and Microsystems Part B, 47 (2016) 419-428.
16. M. Humenberger, C. Zinner, M. Weber, W. Kubinger and M. Vincze, *A fast stereo matching algorithm suitable for embedded real-time systems.* Comput. Vision Image Understanding, 114 (2010) 1180-1202.
17. L. De-Maeztu, A. Villanueva and R. Cabeza, *Near real-time stereo matching using geodesic diffusion.* IEEE Trans Pattern Anal Mach Intell., 34 (2012) 410-416.
18. G. Li, X. Zhang, C. Li, H Jin and J. Zhao, *Design and application of parallel stereo matching algorithm based on CUDA.* Microprocessors and Microsystems Part A, 47 (2016) 142-150.
19. A. Hosni, M. Bleyer, M. Gelautz and C. Rhemann, *Local stereo matching using geodesic support weights, Image Processing, 16th IEEE International Conference on. 2009. pp. 2093-2096.*
20. E.T. Psota, J. Kowalczuk, J. Carlson, and L.C.Pérez. *A local iterative refinement method for adaptive support-weight stereo matching. International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV),* 2011.