A New Basic Correlation Measurement for Stereo Matching

Bowen Shi,[1,3] Shan Shi,[2] Junhua Wu[1] and Musheng Chen[1,*]

[1] *School of Software, Jiangxi University of Science and Technology, Nanchang 330013, China*
[2] *School of Software, Jiangxi Agricultural University, Nanchang 330045, China*
[3] *hirobumi_shi@protonmail.com*
[*] *dreaminit@163.com*

Abstract: In this paper, we propose a new stereo matching algorithm to measure the correlation between two rectified image patches. The difficulty near objects' boundaries and textureless areas is a widely discussed issue in local correlation-based algorithms and most approaches focus on the cost aggregation step to solve the problem. We analyze the inherent limitations of sum of absolute differences (SAD) and sum of squared differences (SSD), then propose a new difference computation method to restrain the noise near objects' boundaries and enlarge the intensity variations in textureless areas. The proposed algorithm can effectively deal with the problems and generate more accurate disparity maps than SAD and SSD without time complexity increasing. Furthermore, proved by experiments, the algorithm can also be applied in some SAD-based and SSD-based algorithms to achieve better results than the original.

© 2019 Optical Society of America under the terms of the OSA Open Access Publishing Agreement

1. Introduction

Depth estimation is one of the fundamental problems in computer vision and has many applications in practice, *e.g.*, 3D reconstruction [1, 2], autonomous driving [3] and medical diagnosis [4–6]. It aims to extract 3D information from a single or a pair of stereo images. Given two rectified images I and I' from a binocular stereo vision system and a point V on a 3D object projected to I at pixel coordinate $(u, v)^T$ and I' at $(u, v + d)^T$, the depth z of the point V can be computed using the following relation:

$$z = \frac{bf}{d}, \tag{1}$$

where b is the length of the baseline and f is the focal length of the camera. The goal of stereo matching is to find the disparity d between two images.

Recent years, inspired by the successful applications of machine learning in various fields, more and more stereo matching algorithms based on machine learning [7–9] were proposed and achieved more accurate results than before. But these algorithms still need a long time to be applied in practice due to the demand of high computational power. As shown in [7], MC-CNN algorithm was implemented on a NVIDIA GeForce GTX Titian GPU. Training takes 5 hours and predicting a single image pair takes 100 seconds [7]. Therefore, considering the expensive price and high runtime, most researchers still use traditional algorithms in practice [2, 5, 6, 10].

Local correlation-based stereo matching algorithms are typically fast and require less memory, so that many real-time stereo vision systems use local algorithm to compute matching cost. The most common local algorithms include *sum of absolute differences* (SAD) and *sum of squared differences* (SSD), which are highly applied [5, 11]. According to the taxonomy of Scharstein and Szeliski [12], a typical stereo matching algorithm can be generally divided into four steps: cost computation, cost aggregation, disparity computation, and disparity refinement. Most SAD-based and SSD-based algorithms [13–17] focus on the last three steps to overcome the defects of SAD or SSD.

The difficulty near objects' boundaries and textureless areas is a widely discussed issue in

local correlation-based algorithms. The boundary problem is referred to as *factionalism* [18]. As discussed in [12], the size of matching window determines how many pixels will be used for correlation. A large window uses more pixels so it performs well in textureless areas, but it blurs objects' boundaries on the other hand. According to [14], boundaries' actual positions are usually within the half distance of the matching window's size. In contrast, a small window can produce more accurate results at boundaries but it frequently fails in textureless areas because of its lack of enough image intensity variations. The common approaches to solve the two problems can be divided into three categories: adaptive window method [13], multi-window method [14–16], and adaptive weight method [19].

In this paper, we focus on the cost computation step and analyze the limitations of absolute differences and squared differences, then propose a new difference computation method to deal with the factionalism and textureless problem of SAD and SSD. Our method only modifies the computation of differences, hence it won't possess higher time complexity than SAD and SSD. Moreover, proved by experiments, the new basic correlation measurement based on the new difference computation method can be applied to some SAD-based or SSD-based algorithms, and achieve better results than before. We use the new correlation to optimize the algorithm proposed in [15].

The contributions of this paper are:

- a new correlation measurement for stereo matching, which generates better performance than SAD and SSD without time complexity increasing;
- optimizing the algorithm proposed in [15].

2. Related Work

Many algorithms were proposed to overcome the difficulty near objects' boundaries and textureless areas. The common approaches focus on the cost aggregation step to decide how many and what kind of pixels are used to compute the matching cost. These approaches can be generally divided into three categories: adaptive window method, multi-window method, and adaptive weight method.

Kanade and Okutomi [13] proposed an adaptive window approach. It starts with an initial disparity map and chooses different size and shape of the matching windows according to the intensity and disparity variance, to update each disparity value till it converges, so the algorithm is sensitive to the initial disparity map.

Fusiello *et al.* [16] proposed an efficient multi-window method. It uses nine windows to compute the matching cost and the initial disparity value respectively, then chooses the initial disparity value of the smallest matching cost of the nine windows as the final. Though it uses nine windows, it is very fast because the matching cost and the initial disparity map can be computed for only one window and the final disparity value is obtained from the neighbouring pixels around the point.

Yoon and Kweon [19] proposed an adaptive support-weight approach. The algorithm gives the support weights to each pixel in the matching window according to the color dissimilarity and spatial distance between the pixels and the central pixel. The weights implicitly take the function of color segmentation so it can effectively restrain the influence of the points from other objects in the matching window. The algorithm can generate very accurate disparity maps but it is also computationally expensive.

Gupta and Cho [15] proposed a real-time approach to correct the disparities near the depth discontinuities. The method first uses a large window to generate the initial disparity map, which performs well in textureless areas, then uses a small window near the depth discontinuities to refine the boundaries blurred by the large window. The average matching error of the algorithm

with full steps is approximately one percentage point lower than the algorithm without small window matching.

3. The limitations of SAD and SSD

In general, the matching cost of SAD can be computed with the following equation:

$$C_{SAD}(u, v, d) = \sum abs(W_l(u, v) - W_r(u, v - d)), \tag{2}$$

where (u, v) is the pixel coordinate of an image point and $W(u, v)$ is a matrix containing the intensities in the matching window centered at (u, v). The function $abs()$ calculates absolute value for all the elements of a matrix. Like SAD, the cost function of SSD can be defined as following:

$$C_{SSD}(u, v, d) = \sum sqr(W_l(u, v) - W_r(u, v - d)), \tag{3}$$

where $sqr()$ calculates the square of all elements of a matrix.

As mentioned in Section 1, factionalism and textureless problem are the defects that are widely discussed in SAD and SSD. Many algorithms focus on the cost aggregation step to solve the problems. But here, we want to show the inherent limitations of SAD and SSD to figure out why they result in the problems. SSD usually generates more accurate results than SAD with the same matching window in textureless areas, because the intensity values change little in textureless areas and square function enlarges the intensity variations to capture the dominant differences which are from the unrelated points in a matching window. The ground in the image Motorcycle [20] is a typically textureless area. Motorcycle image and its disparity maps computed by SAD and SSD with a 9×9 matching window are shown in Fig. 1. From the disparity maps we can see that the results computed by SSD in textureless areas are better than those by SAD.

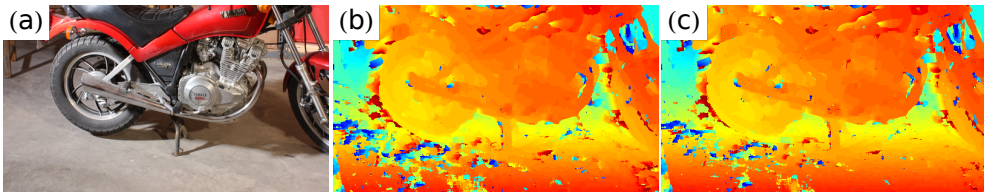

Fig. 1. Textureless areas in Motorcycle image. (a) Motorcycle image; (b) disparity map computed by SAD; (c) disparity map computed by SSD.

Though SSD takes advantage in textureless areas, it is more sensitive to noise than SAD. Noise points in matching windows usually result in high differences. SSD will enlarge them several times to unrelated points' differences and lead to mismatch. Both SAD and SSD are influenced by noise points, especially near depth discontinuities which commonly occur at objects' boundaries. In depth discontinuities, the pixels on the other side usually bring larger differences. These differences will raise the matching cost of correctly matched windows and lead to mismatch, then blur boundaries.

4. Proposed algorithm

4.1. New Difference Computation Method

The value of absolute differences is between 0 and 255. According to the constraint of photometric compatibility [21], the intensity difference between two related pixels won't change too much. We divide the interval $[0, 255]$ into three subintervals, $[0, t_1]$, $(t_1, t_2]$, and $(t_2, 255]$. The differences within $[0, t_1]$ are very small, which are usually from the correctly matched points. The differences

within $(t_1, t_2]$ should be computed like square function in order to enlarge the intensity variations and capture the dominant differences which are from the unrelated points. Finally, the differences within $(t_2, 255]$ generally belong to two categories, one from unrelated points and another from noise. Both of them should have similar differences in the matching cost, because enlarging the noise's differences will raise the correctly matched windows' matching cost and lead to mismatch. Related points' differences rarely appear in the interval, thus we can identify the mismatched windows by the quantity of unrelated points, but the difference's value.

To demonstrate our points, we make a histogram for all absolute differences in the 9×9 correctly matched windows in the training images that satisfy photometric compatibility constraint in Middlebury 2014 dataset [20], which is illustrated in Fig. 2.

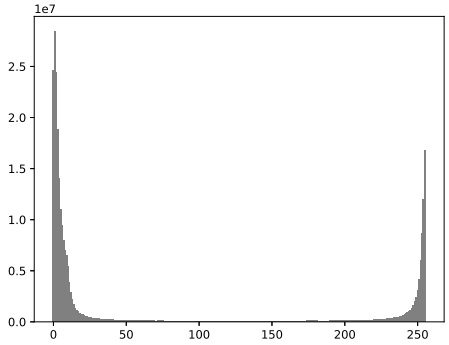

Fig. 2. The histogram for all absolute differences in correctly matched windows.

It is obvious that substantial differences are less than 10 approximately. It demonstrates that the matching windows are correctly matched. And there are also many differences greater than 230. These differences are from the noise points and they shouldn't be computed in the matching cost.

According to the statistics, we define a new function to replace the square function and absolute function in the difference computation, as shown in Eq. (4).

$$f(x) = \frac{s}{1 + \exp\left(-\frac{|x|-t}{0.14t}\right)}, \tag{4}$$

where s is an arbitrary scale factor, and t is the extreme value of the function's derivative, where the function changes most intensely. In addition, t is also the median of the interval $[t_1, t_2]$, where the differences should be computed like the square function to enlarge the intensity variations. Fig. 3 shows the image of the function and its derivative, where we set s to 255 and t to 12.5. From that we can see the absolute differences greater than 25 tend to a same result, which can restrain the influence of noise.

We refer the function to as X function, and its matching cost function is called SXD which is defined as below,

$$C_{SXD}(u, v, d) = \sum X(W_l(u, v) - W_r(u, v - d)). \tag{5}$$

The matching window size of SXD can be set to the same as SAD and SSD use, hence the time complexities of the three algorithms are equal. The runtime of the three difference computation function are approximate, so that the differences between the total runtime of the three algorithms are negligible.

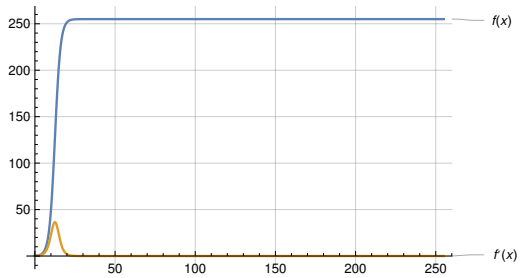

Fig. 3. The image of the function and its derivative

4.2. Application

In this section, we use SXD to optimize the algorithm proposed in [15]. The processing steps of the algorithm are:

1. left disparity map and right disparity map are computed using SAD with a large matching window;
2. use SAD with a small matching window to correct the disparity values near the depth discontinuities;
3. unreliable pixel detection with left-right consistency check;
4. disparity interpolation using neighbouring reliable pixels;
5. disparity refinement.

According to [15], the matching cost $C(u, v, d)$ computed for a large window is given as follows,

$$C(u, v, d) = \sum abs(W_l(u, v) - W_r(u, v - d)) + \xi, \tag{6}$$

where ξ is a penalty term based on gradient and the disparity of neighbouring pixel in the same epipolar line, which makes the disparity estimation more accurate in non-textured areas, and it is defined as below,

$$\xi = T \times |d - d'| \times \left(1 - \frac{|I_l(u, v) - I_l(u, v)'|}{255}\right), \tag{7}$$

where T is the constant set to 8 in [15], and d' is the disparity of the left or the right neighbouring pixel (u, v').

We remove the second step of the algorithm and replace SAD in the first step with SXD to compute left and right disparity maps. SXD performs well near depth discontinuities so it is no use to correct the disparity values there with small window matching. Though SXD also performs well in textureless areas, The penalty term ξ in the matching cost function is still necessary, because it introduces the constraint of disparity smoothness [21]. And in order to be compatible with the penalty term ξ of the large window matching step with the same parameter T, we set the parameter s of SXD to 255, which makes the range of SXD same as SAD. Furthermore, for the rest processing steps, we don't make any changes. The purpose is to show that SXD can achieve better results not only than SAD and SSD, but also in some SAD-based and SSD-based algorithms.

5. Experimental Results

We evaluate the performance of the algorithms with the Middlebury's evaluation SDK and its training dataset [20], which is called MiddEval3. The images of the training dataset we use to compute the average error are only the quarter resolution images that satisfy the constraint of photometric compatibility. The disparity estimations that differ from the ground truth by more than one pixels are marked as bad pixels. For the parameters of SXD, we set s to 255 and t to 12.5.

5.1. Experiment on Different Windows

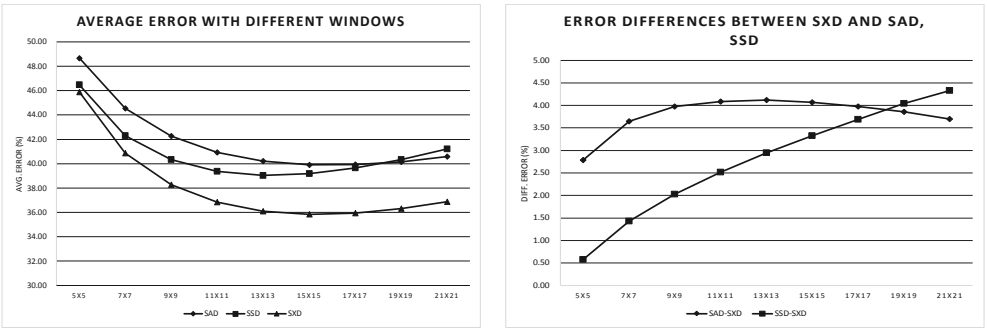

Fig. 4. The average errors of different window sizes, and the error differences between SXD and SAD, SSD.

Fig. 4 shows that the average errors of SAD, SSD and SXD, which are evaluated by MiddEval3, and the error differences between SXD and SAD, SSD. From the figure, we can see that the average errors of SXD are lower than SAD and SSD no matter what size windows are used. And with the increase of matching window's size, the error of SSD grows faster than SAD and SXD. It is because the large window contains more noise than the small one so that square function will enlarge the noise's influence on the matching result. It demonstrates that SXD can effectively deal with the noise points and is less affected by window size.

5.2. Experiment on Different Post-Processing Algorithms

For demonstrating that SXD is compatible with the post-processing algorithms that SAD and SSD commonly use. We choose the three most common post-processing algorithms to test, which are subpixel enhancement, disparity interpolation, and median filter.

Subpixel enhancement is a useful approach to make the disparity estimation more accurate. It fits a quadratic curve through the three neighbouring matching cost values to obtain a new disparity value.

$$d(u,v) = d(u,v) - \frac{C(u,v,d(u,v)+1) - C(u,v,d(u,v)-1)}{2(C(u,v,d(u,v)+1) - 2C(u,v) + C(u,v,d(u,v)-1)}. \tag{8}$$

Disparity interpolation is a step following left-right consistency check. The left-right consistency check is a very effective way to detect the unreliable points, especially those in occluded regions. Once a point is marked as unreliable, its disparity value will be recomputed with the neighbouring pixels. The reliable pixels must satisfy the condition:

$$|d_l(u,v) - d_r(u,v-d)| \leq t, \tag{9}$$

where t is a threshold to distinguish the reliable points and the unreliable points. In this experiment, we set it to 1.0. For the unreliable pixels, we move left until find a reliable pixel and replace the unreliable pixels' disparity value with the reliable pixel's own.

Median filter can restrain the noise in images. For disparity map refinement, it replaces a pixel's disparity with the median of the window centered at the pixel, so that it can eliminate the extreme values which usually are the mismatched points. In this experiment, we set the size of the window to 5×5.

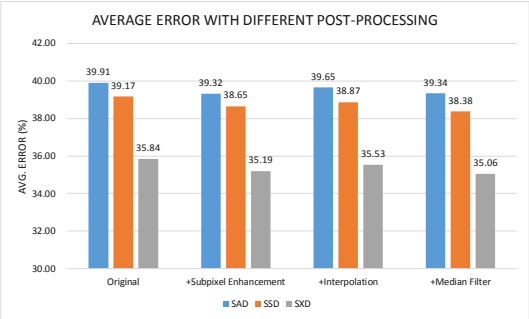

Fig. 5. The average errors with different post-processing algorithms.

Fig. 5 shows the results of this experiment. In this experiment, the size of the matching window is set to 15×15. We are told from the figure that these post-processing algorithms are compatible with SXD, and no matter what post-processing SXD use, the results of SXD are better than SAD and SSD.

5.3. Experiment on Application

The algorithm proposed in [15] and its SXD edition are referred to as *WBA* and *WBA-SXD* in this section. As shown in [15], the best choice of the size of the large window is 9×9, and the small window is 3×3. Therefore, we use the sizes for WBA in this experiment. And for the algorithms without small window matching, we set their matching window size to the same as WBA's large window size. Fig. 6 shows the comparison of the average error of the algorithms: SAD, SSD, SXD, WBA, and WBA-SXD.

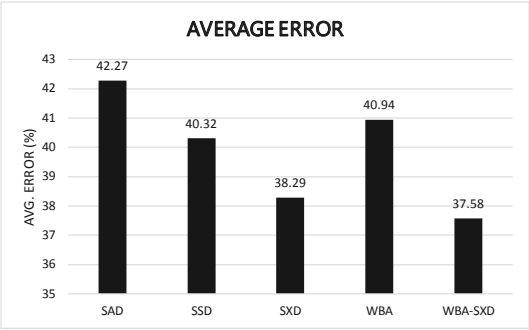

Fig. 6. Average error comparison.

It can be seen clearly from the figure that WBA-SXD performs best among these. The average error of it is obviously lower than the others. It indicates that SXD can effectively raise the accuracy near depth discontinuities without small window matching and is compatible with these post-processing steps of WBA. Fig. 7 shows the disparity maps of image Backpack [20], which are generated by SAD, SSD, SXD, WBA, and WBA-SXD. From the figure, we can see that the

disparity map of WBA-SXD is more smooth than WBA. That's because SXD can generate good results in textureless areas as well.

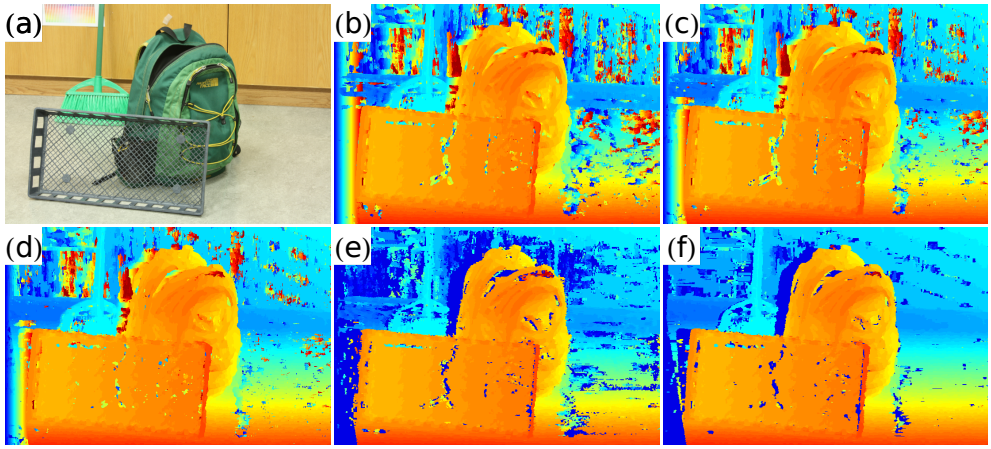

Fig. 7. The disparity maps of Backpack image. (a) Left image; (b) disparity map generated by SAD; (c) disparity map generated by SSD; (d) disparity map generated by SXD; (e) disparity map generated by WBA, (f) disparity map generated by WBA-SXD.

5.4. Experiment on Runtime

We implement these algorithms including SAD, SSD, SXD, WBA, and WBA-SXD on a computer with a NVIDIA GTX 860M GPU. Fig. 8 shows the runtime of these algorithms on the Backpack's quarter resolution image whose resolution is 737×497. All algorithms use a 9×9 matching window and the small matching window of WBA is 3×3.

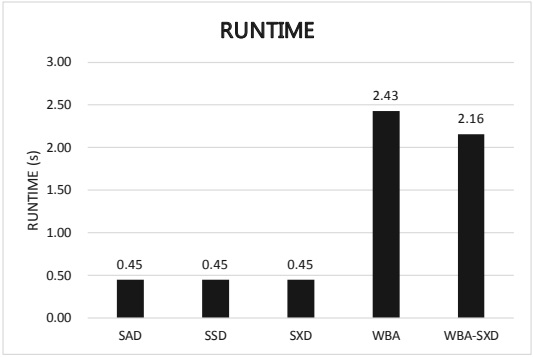

Fig. 8. Runtime comparison.

As can be seen from the figure that SAD, SSD, SXD take the same amount of time, and WBA-SXD is less than WBA. It is mainly because SXD has the same time complexity with SAD and SSD, and the differences between the cost computation step's runtime are negligible. WBA-SXD don't use the small window matching step, so it is faster than WBA.

5.5. Experiment on Depth Discontinuities

Fig. 9 shows the disparity maps which are computed by SAD, SSD, and SXD with a 21×21 matching window, and their bad pixel maps of the image Map [12]. Map is a pair of stereo

images with typical occluded regions and object's boundaries. We are told from the bad pixel (nocc) maps that the errors of SXD in non-occluded regions are smaller than SAD and SSD, because of its restraint for noise. It demonstrates that SXD can effectively deal with factionalism. Furthermore, the errors of SXD in all areas are also smaller than SAD and SSD, especially in the occluded regions. The results may point out that SXD are better in occluded regions than SAD and SSD, but it needs more experiments to verify.

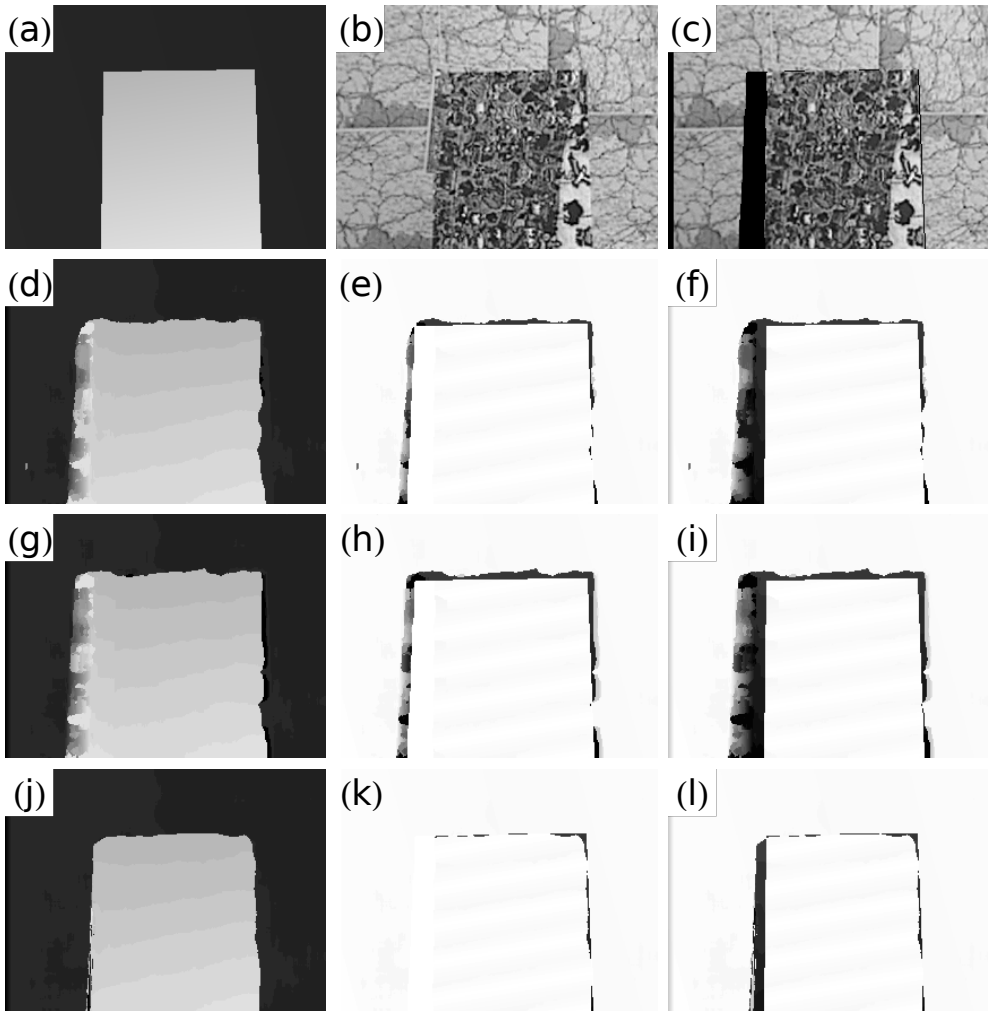

Fig. 9. The disparity maps of Map image. (a) Ground truth; (b) left image; (c) occlusion areas marked as black in the left image; (d) disparity map generated by SAD; (e) bad pixels (nocc) of SAD; (f) bad pixels (all) of SAD; (g) disparity map generated by SSD; (h) bad pixels (nocc) of SSD; (i) bad pixels (all) of SSD; (j) disparity map generated by SXD; (k) bad pixels (nocc) of SXD; (l) bad pixels (all) of SXD.

5.6. Experiment on Textureless Areas

As shown in Fig. 7, the ground and the wardrobe are typically textureless areas, and SXD, WBA-SXD generates more smooth results than SAD, SSD, and WBA there. That's because SXD enlarges the intensity variations in the same size matching window, and restrain the noise

meanwhile. Moreover, with the increase of the matching window size, SXD and WBA-SXD can generate better results than the small window. The disparity maps computed with a 21×21 matching window are shown in Fig. 10.

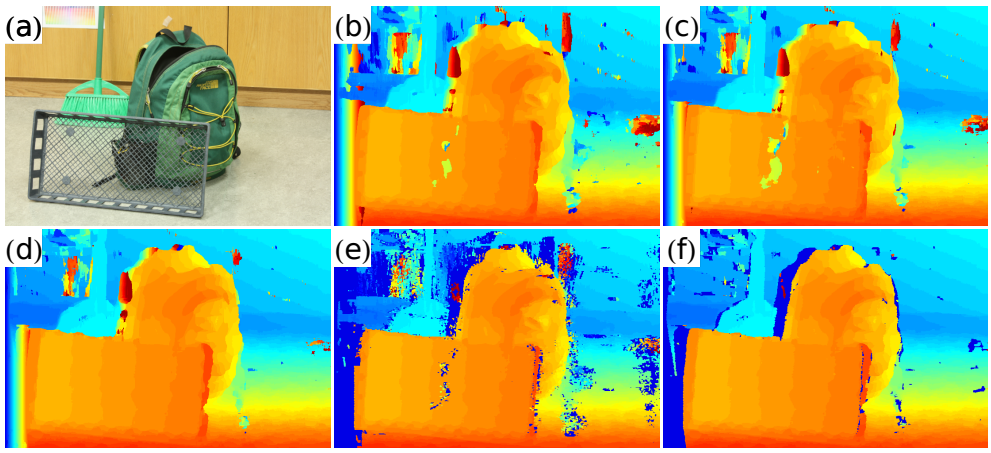

Fig. 10. Backpack image's disparity maps computed with 21×21 window. (a) Left image; (b) disparity map generated by SAD; (c) disparity map generated by SSD; (d) disparity map generated by SXD; (e) disparity map generated by WBA, (f) disparity map generated by WBA-SXD.

5.7. Experiment on Best t Value

The parameter t in Eq. (4) represents the extreme value of X function's derivative. It determines what interval of differences are important for stereo matching. The best value of t is not a fixed number in any situation. Fig. 11 shows the standardized errors of the different values of t with a fixed window size 15×15 in different images. We can see that the best value of t varies in different images. But according to the curve of the mean errors, and the points of intersection of all curves, it is easy to find an appropriate value of t for the use case.

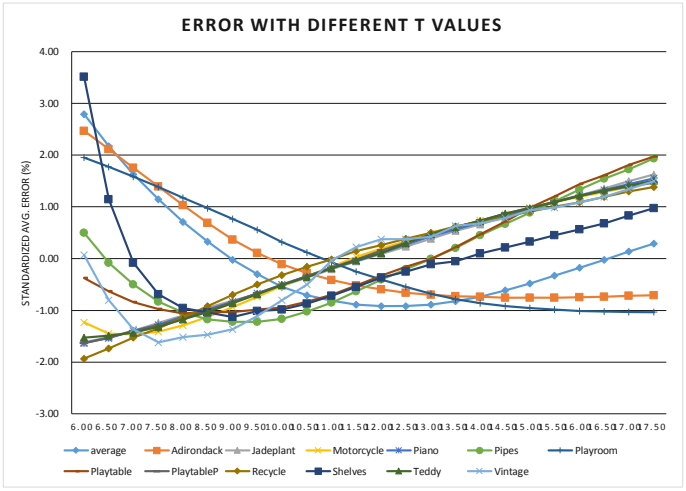

Fig. 11. The best t value for different images.

6. Conclusion

In this paper, we present a new correlation measurement for stereo matching, which is based on the analysis on the limitations of SAD and SSD. The analysis shows the reason why SAD and SSD encounter the difficulty near objects' boundaries and textureless areas. According to the analysis, we propose a new difference computation method to enlarge the intensity variations in textureless areas and restrain the noise near objects' boundaries. The correlation measurement using the new difference computation method can effectively deal with the problems with the same time complexity as SAD and SSD, and can be applied to some SAD-based and SSD-based algorithms to achieve better results than the original. We demonstrate the points by experiments, and the source code implemented for the experiments is available at https://github.com/GreatestCapacity/SXD.

References

1. C. Zhang, Z. Li, Y. Cheng, R. Cai, H. Chao, and Y. Rui, "Meshstereo: A global stereo model with mesh alignment regularization for view interpolation," in *Proceedings of IEEE International Conference on Computer Vision,* (IEEE, 2015), pp. 2057–2065.
2. D. Wang, H. Liu, and X. Cheng, "A miniature binocular endoscope with local feature matching and stereo matching for 3d measurement and 3d reconstruction," Sensors **18**, 2243–2264 (2018).
3. A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,* (IEEE, 2012), pp. 3354–3361.
4. F. Zafar, Y. Yesha, and A. Badano, "Computational observers and visualization methods for stereoscopic medical imaging," Opt. Express **22**, 22246–22267 (2014).
5. I. T. Comlekciler, S. Gunes, and C. Irgin, "Three-dimensional repositioning of jaw in the orthognathic surgery using the binocular stereo vision," Sci. Iran. **0**, 1–33 (2017).
6. A. T. Gabriel, C. Quaresma, M. F. Secca, and P. Vieira, "Development and clinical application of vertebral metrics: using a stereo vision system to assess the spine," Med. & Biol. Eng. & Comput. **56**, 1435–1446 (2018).
7. J. Zbontar and Y. LeCun, "Computing the stereo matching cost with a convolutional neural network," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,* (IEEE, 2015), pp. 1592–1599.
8. J. Zbontar and Y. LeCun, "Stereo matching by training a convolutional neural network to compare image patches," J. Mach. Learn. Res. **17**, 1–32 (2016).
9. A. Shaked and L. Wolf, "Improved stereo matching with constant highway networks and reflective confidence learning," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition,* (IEEE, 2017), pp. 6901–6910.
10. L. Ge, Z. Yang, Z. Sun, G. Zhang, M. Zhang, K. Zhang, C. Zhang, Y. Tan, and W. Li, "A method for broccoli seedling recognition in natural environment based on binocular stereo vision and gaussian mixture model," Sensors **19**, 1132–1149 (2019).
11. X. Zhang and Z. Chen, "Sad-based stereo vision machine on a system-on-programmable-chip (sopc)," Sensors **13**, 3014–3027 (2013).
12. D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," Int. J. Comput. Vis. **47**, 7–42 (2002).
13. T. Kanade and M. Okutomi, "A stereo matching algorithm with an adaptive window: Theory and experiment," IEEE Transactions on Pattern Analysis Mach. Intell. **16**, 920–932 (1994).
14. H. Hirschmuller, P. R. Innocent, and J. Garibaldi, "Real-time correlation-based stereo vision with reduced border errors," Int. J. Comput. Vis. **47**, 229–246 (2002).
15. R. K. Gupta and S.-Y. Cho, "Window-based approach for fast stereo correspondence," IET Comput. Vis. **7**, 123–134 (2013).
16. A. Fusiello, V. Roberto, and E. Trucco, "Efficient stereo with multiple windowing," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,* (1997), pp. 858–863.
17. S. Adhyapak, N. D. Kehtarnavaz, and M. Nadin, "Stereo matching via selective multiple windows," J. Electron. Imaging **16**, 1–14 (2007).
18. R. Zabih and J. Woodfill, "Non-parametric local transforms for computing visual correspondence," in *Proceedings of European Conference on Computer Vision,* (Springer, 1994), pp. 151–158.
19. K.-J. Yoon and I. S. Kweon, "Adaptive support-weight approach for correspondence search," IEEE Transactions on Pattern Analysis Mach. Intell. **28**, 650–656 (2006).
20. D. Scharstein, H. Hirschmuller, Y. Kitajima, G. Krathwohl, N. Nesic, X. Wang, and P. Westling, "High-resolution stereo datasets with subpixel-accurate ground truth," in *Proceedings of German Conference on Pattern Recognition,* X. Jiang, J. Hornegger, and R. Koch, eds. (Springer, 2014), Lecture Notes in Computer Science, pp. 31–42.
21. M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis, and Machine Vision* (Cengage Learning, 2008), 3rd ed.