Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Image and Signal Processing (CIISP 2007)

A Multi-Window Stereo Vision Algorithm With Improved Performance at Object Borders*

Jun Zhao, Jayantha Katupitiya
ARC Centre of Excellence for Autonomous Systems
School of Mechanical and Manufacturing Engineering
The University of New South Wales, Sydney, NSW 2052, Australia
J.Katupitiya@unsw.edu.au

***Abstract*— Conventional correlation based stereo vision algorithms have poor performance at object borders due to occlusion. In this paper, a new matching sequence has been introduced. To detect left object border, a left shifted window is used with right image as the reference image and process from left to right. This method gives excellent left object border detection while detection of right object border is poor. The right object borders can be detected with equal success using a right shifted window with left image as reference and process from right to left. The combination result is good detection at both object borders.**

***Index Terms*— Occlusion, SMP, correlation, stereo vision.**

I. INTRODUCTION

Correlation-Based matching is an old but still widely used method for stereo vision due to its time efficiency compared to energy minimization methods. Stereovision systems determine depth from two images which are taken at the same time, but from slightly different viewpoints. The most important and time consuming task for a stereovision system is the registration of both images, i.e. the identification of corresponding pixels [1]. Considering only epipolar condition, in which there is only horizontal shift in left and right images, given a certain pixel in the right image, if we want to find the corresponding (or matching) pixel in the left image, we only need to search to the right in the same scan line for a certain range. Normally we compare the gray level difference not for a single pixel, but for a window around the pixel, and the pixel with least difference is selected as the matching point. This method is called Winner Takes it All (WTA). A difficulty arises in uniform grey level area in which there may exist several good matches. In these areas, the WTA is prone to errors because there isn't a way to optimally choose from among all good matches. A bigger window size can reduce this error. However, correlation assumes that the depth is equal for all pixels of a correlation window. This assumption is violated at depth discontinuities. The result is that, depending on the size of the correlation window [1], object borders are blurred and small details of objects are lost. Furthermore, WTA does not consider occluded areas, which are parts of stereo images that are only visible in one of the two images. This leads to poor matches within occluded areas.

*This work is supported in part by the ARC Centre of Excellence programme, funded by the Australian Research Council (ARC) and the New South Wales State Government.

Many methods were introduced to solve the problems caused by the fixed window size. They try to use different windows which are optimum at different points. In [2], the authors performed plausibility hypothesis testing and choose an arbitrarily shaped connected window. In [3], the authors adjust the support-weight of each pixel inside a given support window thus also form an arbitrary shaped connected window. This method also improve performance at depth discontinuities but too slow for real time operation because it requires exponent calculation. In [4], the author restrict the window to rectangle and computed the correlation value of several different window sizes for the pixel of interest. In [5], the authors use 9 windows that have the same size, but different positions with respect to the pixel of interest. The correlation is done with all 9 windows for every pixel and every possible disparity. In [1], the authors use a 5 window(1 center, 4 corner) configuration, correlation is done for all 5 windows for every pixel at every possible disparity. At every possible disparity the correlation value is the sum of the error of the central window and lowest 2 of the 4 surrounding windows. The last two approaches are time efficient and suitable for real-time tasks because the window size is fixed, however, as mentioned in [1], small details compared to the window size are lost.

All local area based methods discussed earlier has one feature in common. Although the final window may have different shape and located at different positions with respect to the pixel under consideration, in the beginning, the pixel is in the center of the initial support window. In this paper, we first analyze the characteristics at depth discontinuities, then we introduce a shifted window scheme accompanied by special matching direction. These are introduced in section II. In section III, we discuss a new method to assign disparities within the invalided areas. In section IV, we introduce a new Left Right Matching scheme to accompany the new matching window. In section V, we show the experimental results and draw some conclusions.

II. DEPTH DISCONTINUITY AND NEW MATCHING WINDOW

A. Characteristics at depth discontinuities

At a depth discontinuity, or depth jump, some pixel in one view can not be seen in the other view. This is the occluded area and so there should not be a match between the pixels of this area in this view with any pixel in the other view. If there is a distinctive feature present in one image and absent in the other, then it signifies an occluded area. In the absence of such distinctive features, the identification of occlusion is difficult. This situation occurs more often and it leads to a lot of incorrect matches.

Fig. 1 is a simplified example occluded area near a depth jump. L represents one left image line and R represents the corresponding epipolar line in right image. The thinner potion of the line represents lower intensity and the thicker portion of the line represents higher intensity. The lower intensity is the background and the disparity is D_1. The higher intensity is of the object at foreground and the disparity is D_2. If we use right view as the reference, and process from left to right, using a window centered at the pixel under consideration with width $2w+1$. Suppose up to A_R the matchings are correct and that A_R matches with A_L, correctly. Similarly, C_R is matched with C_L correctly. All pixels right of C_R also have correct matches. When we come to B_R, and given that B_R is close to the intensity change (due to depth jump), the support window which is placed centrally at B_R will have part of it crossed over to the higher intensity side as shown in this figure. If we now use SAD(Sum of Absolute Difference) as the window cost, then B_R would be wrongly matched to B'_L, instead of the correct pixel B_L. We can see that all the pixels in the left within distance w to C_R have the same problem. They will all be assigned the wrong disparity D_2. Of course, in the real scene the intensity change would not be so sharp, but this would happen to some extent. Fig. 1 shows a positive depth jump($D_2 > D_1$) or a left object border, in Fig. 2 there is a negative depth jump of($D_2 < D_1$) or a right object border. If we still use right image as reference and process from left to right in this situation, we can see that all the pixels within distance w to the right of C_R would be wrongly selected the disparity D_1, but in fact the area to the right of C_R is occluded and should have disparity D_2. In both cases of Fig. 1 and Fig. 2, some pixels that should have smaller disparity are wrongly assigned the larger disparity, thus causing the phenomena that the size of the object with bigger disparity are extended horizontally, or fattened.

B. A new matching window and scheme

Because the border error is due to the window which is centered at the pixel under consideration as analyzed in Section II-A , we put the window at the side of the pixel under consideration, like in Figure 3. Consider the situation in Figure 1 again. Suppose we use support window 1 shown in Figure 3, use right view as reference and process from left to right. At point B_R, this time the support window is to the left of this pixel and then it is correctly matched to B_L, and

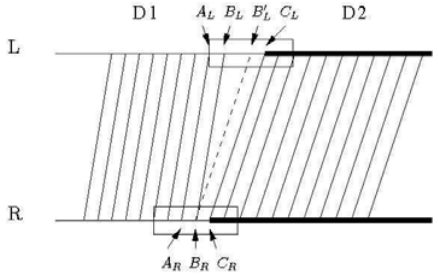


Fig. 1. Positive depth jump

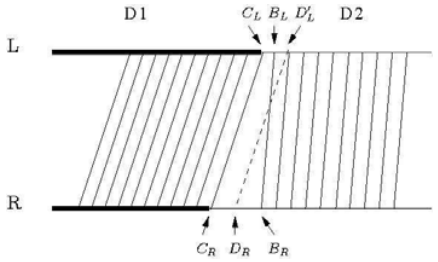


Fig. 2. Negative depth jump

we can see that all the pixels to the left of C_R can find good match, so are the pixels to the right of C_R. But in situations like in Figure 2, pixels to the right of C_R will have bad matches to the pixels to the right of C_L in left view, D_R would be wrongly matched to D'_L. However, if this time we use left view as reference, using support window 2 shown in Figure 3, and process from right to left, this time at point C_L we can detect a depth jump and therefore we can categorize the pixels between C_R and B_R in the right view into occluded area, and good disparity can be selected. Similarly, we can find that in situation shown in Figure 1, if we use left view as reference, using support window 2, and process from right to left, the occluded area in left view would be wrongly handled, but this area could be well handled previously. Above all, support window 1 can find a positive depth jump, if we use right view as reference, and process from left to right; symmetrically, support window 2 can find a negative depth jump, if we use left view as reference, and process from right to left. Actually, this negative depth jump from the view point of right image is a positive depth jump from the view point of left image. As can be seen, if we process from right to left, we first find D_2, then D_1, and $D_1 > D_2$.

Windows shown in Fig. 3 can detect vertical object borders. If the object border is not vertical, each window will comes across object and background. Thus we use a 4 window configuration, shown in Fig. 4. In this figure, if we use right image as reference, processing from left to right, we use support window w1 and w2; on the contrary, if we use left image as reference, processing from right to left, we use support window w3 and w4. For example, if there is a left object border like that shown in Fig. 4, support window w1 would find a perfect match, but support window w2 fails,

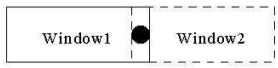


Fig. 3. New window

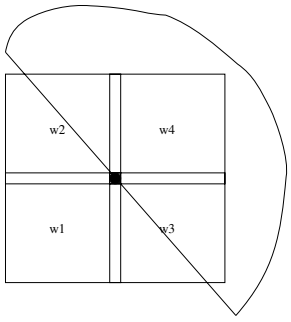


Fig. 4. 4 window configuration

because w2 comes across both foreground and background, while w1 do not have this problem. Thus in each matching phase, we choose from the 2 support windows and select the one with smaller matching error.

The matching window introduced above is different from other multiple window method such as [5]: in our method, when using right image as reference, left shifted windows are selected and process from left to right, in favor of detecting left object border; when using left image as reference, right shifted windows are selected and process from right to left, in favor of detecting right object border. While in normal multiple window method, no matter which image is used as reference, for each pixel all the support windows are selected and process in the same direction. This will cause difference in the resulted disparity images. As will be illustrated in Section III-B.

Next we discuss the procedure we use to find the positive depth jump in each matching phase.

III. SMP AND ITS DISPARITY MAP

A. SMP

In [6], the authors introduced a method to establish depth map using a *Single Matching Phase*(SMP). We use SMP to establish depth map from left and right view respectively. The following constraints are utilized:

- uniqueness constraint [6], enforces a one-to-one mapping between pixels in two images.
- ordering constraint [7], preserves order along scanlines in both input images.
- smoothness constraint [8], states that disparity do not vary much on object surfaces.

Detailed explanation of techniques in the SMP procedure can be seen in [8].

1) Fill in the gaps: Error filter introduced in [8], together with uniqueness constraint and order constraint, will invalidate many incorrect matches in SMP, then pixels considered as having incorrect matches will have no disparities assigned to them. Such an area is shown in Figure 5. In this figure, L identifies the left image line and R identifies the epipolar line in right image. A pixel on the image line is identified with the notation X_i where the subscript refers to the pixel number and X labels L or R depending on the left image or right image respectively. Note that in this figure we have used the right image as the reference, that is, the matching cost $S'(x, y, d)$ is calculated based on the right image. Suppose R_0 has valid disparity d_1 with matching point L_0 in left image and R_{n+1} has valid disparity d_2 with matching point L_{n+m+1}, then $d_2 - d_1 = m$. Pixels R_k which lie somewhere in between R_0 and R_{n+1} $(0 < k < n + 1)$ has no disparities assigned because a confident way of calculating disparities is not yet available. Thus we call this range the *invalid area* in the right image. Depending on the relative magnitudes of d_1 and d_2 there exists three possibilities. First, $d_1 = d_2$, as in Figure 5(a), second, $d_1 < d_2$, as in Figure 5(b), third, $d_1 > d_2$, as in Figure 5(c). In the first case, all pixels that has no disparities assigned must have the same disparity right across. However, in the other two cases, where $d_1 \neq d_2$, previously only the lower disparity is propagated throughout the invalid area [1]. This method is subjective in that there are possibilities that the depth jump may occur at any pixel within the invalid area. In the following discussion we introduce a method that assign the depth jump to any pixel within the invalid area.

In figure 5(b), there is a positive depth jump, or left object border. If all the invalid area is assigned disparity d_1, it corresponds to the depth jump occurring at R_{n+1}. The matching error $E_{R_{n+1}}$

$$E_{R_{n+1}} = \sum_{x=1}^{n} S'(x, y, d_1) \tag{1}$$

Here the window size to calculate the matching cost (for individual pixels) $S'(x, y, d)$ is limited to 1×1, that is, just containing the pixel under consideration. This way a window stretching across a depth jump can be avoided. The matching error associated with any pixel R_k is calculated as:

$$E_{R_k} = \sum_{x=1}^{k-1} S'(x, y, d_1) + \sum_{x=k}^{n} S'(x, y, d_2) \tag{2}$$

To decide at which pixel the depth jump occurs, we compare the matching error associated with every pixel, and select the pixel with the smallest matching error. If E_{R_k} is the smallest, then for the pixels from R_1 to R_{k-1}, d_1 is selected and for pixels from R_k to R_n, d_2 is selected.

In Figure 5(c), there is a negative depth jump, or right object border, thus some area visible in the right view is occluded in the left view. Not all of the pixels in the invalid area of the right image can find their counterparts in the left image, but all the pixels in left image from L_1 to $L_{n+m}(m < 0)$ which are in the invalid area in the left view, can find their matching points in right view. Thus the right view here is like the left view in Figure 5(b), and the left

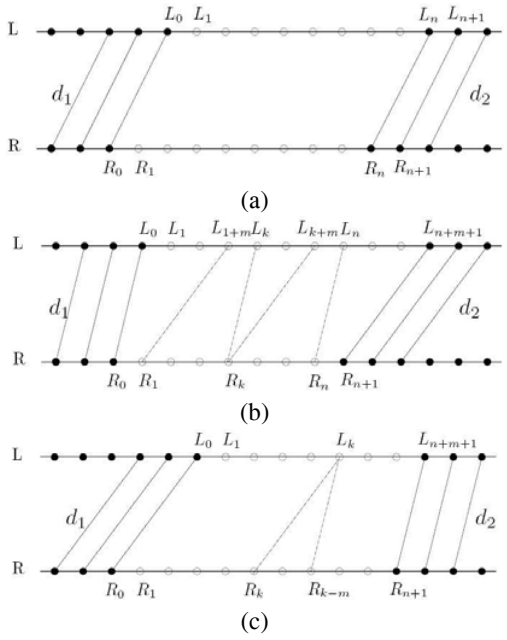


Fig. 5. Invalid areas (a) $d_1 = d_2$ (b)$d_1 < d_2$ (c)$d_1 > d_2$

view here is like the right view in Figure 5(b). So this time we match from left view to right view, and find the depth jump in the left view. The matching error associated with a pixel L_k $(0 < k < n + m + 1)$ which lies somewhere in between L_1 and L_{n+m}, inclusive is:

$$E_{L_k} = \sum_{x=1}^{k-1} S'(x, y, d_1) + \sum_{x=k-m}^{n} S'(x, y, d_2) \qquad (3)$$

Because we use right view as reference, we need to match the depth jump in left view to depth jump in the right view. Suppose L_k is selected as the depth jump in left view, then pixels from R_1 to R_{k-1} should have disparity d_1 and pixels from R_{k-m} to R_n should have disparity d_2. The area from R_k to R_{k-m-1} is occluded, and the occluded area belongs to the background. Therefore for this area the smaller disparity d_2 should be selected.

Intensive calculation of matching error to find the position of depth jump is needed in this method, however, using SIMD(*Single Instruction Multiple Data*) technique thanks to the modern high performance computing, the calculation can be done very quickly.

B. Disparity image from SMP

Using the techniques mentioned above, we can build depth map using SMP. In Left Matching Phase(LMP), we use left image as reference, selecting right shifted windows, and process from right to left; symmetrically, in the Right Matching Phase(RMP), we use right view as reference, selecting left shifted windows, and process from left to right. Figure 6(a) and (b) are the original Tsukuba left and right images, (c) and (d) show the depth map built from SMP, Figure 6(c) use LMP, (d) uses RMP. From Figure 6 we can see that in LMP, we can detect right object borders, or negative depth jumps successfully, but were inaccurate in left object border or positive depth jump detection. However, on the contrary, in RMP, we can detect left object borders successfully, but were inaccurate in detecting right object borders. The inaccuracy is caused by the support windows extending over the object borders and the discontinue penalty function we used.

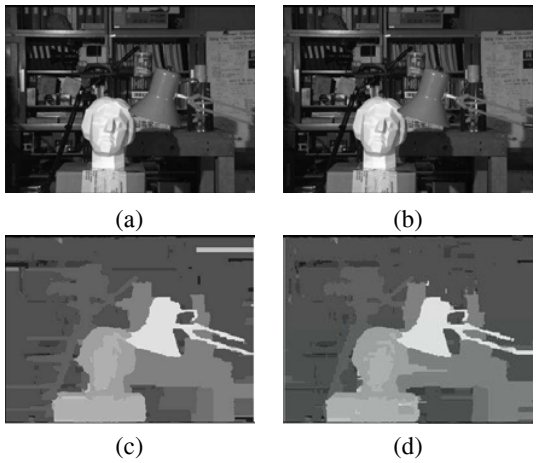


Fig. 6. Tsukuba image

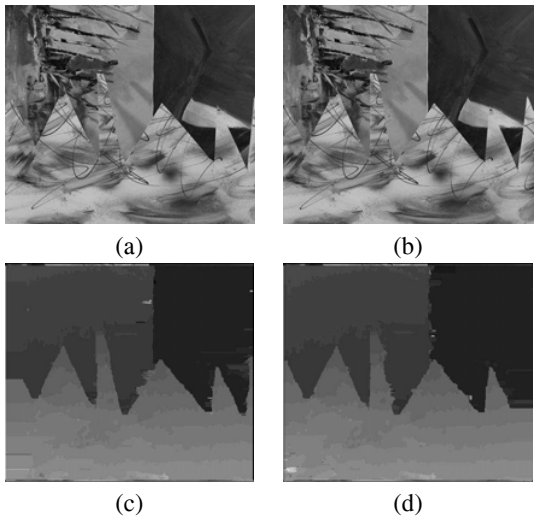
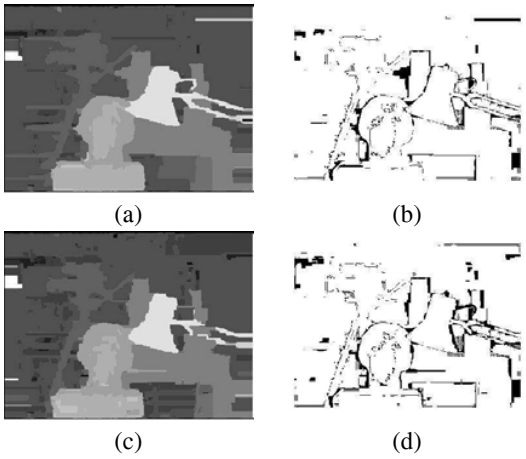


Fig. 7. Sawtooth image

Comparison with normal multiple windowing method are shown in Fig. 8 using left Tsukuba image. Fig. 8(a) is the results using new method: processing from right to left, Fig. 8(b) is the corresponding error images compared with the ground truth; Fig. 8(c) is the results using normal method: processing from left to right and old "Fill in the gaps" method, Fig. 8(d) is the corresponding error images.

Difference between Fig. 8(a) and (c) shows that processing in different direction will result in different disparity images. Comparing Fig. 8(b) with Fig. 8(d) we can see that new



(a) (b) (c) (d)

Fig. 8. Comparison with old method using Tsukuba image

method performs better at right object border, in the cost of performing poorly at left object border.

If we combine these two matching phases, and only select the correct object border from each matching phase, we should be able to build a depth map that is correct at both left and right object borders. In the next section, we will discuss the new Left Right Matching scheme.

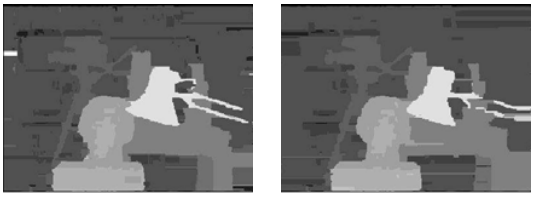
IV. New LRM scheme

In Section II-B, we have introduced a new matching scheme to correctly detect positive depth jumps in each matching phase, but it fails to correctly detect negative depth jumps. So we can think of a procedure that combines the 2 matching phases, we also call it Left Right Matching(LRM).

In normal LRM scheme, a valid correspondence must match in both directions [9]. From Fig. 6 and Fig. 7 we can see, normally near depth jumps, the disparities we get from different matching phases do not agree. If we still use the normal scheme, not only bad matches are discarded, but also the good ones.

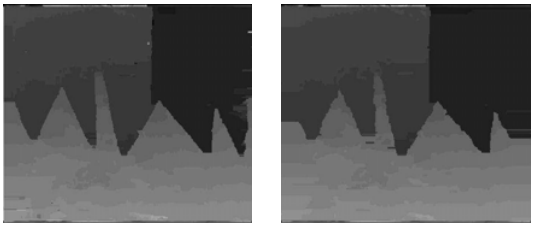
Suppose d_l is the disparity established in LMP, d_r in RMP. In the following we will check d_r according to d_l. Because in each matching phase, we can find a positive depth jump from each point of view, then we will trust this positive depth jump. Like in Fig. 2, in LMP, at point C_L we find a positive depth jump, then according to the jump distance, we can get some occluded area in the right image, that is from C_R to B_R. We can invalidate all the disparities established in RMP inside this area, but that is insufficient. In Fig. 6 (d), we can see from the right side of the statue that sometimes in RMP the depth jump is assigned to a point quite far from the true depth jump. So after B_R in Fig. 2, we will keep searching to the right. For each pixel we compare its disparity d_r obtained from RMP with d_l obtained from LMP, if $d_r > d_l$, we change d_r to be d_l, until $d_r \leq d_l$.

Symmetrically, we can invalidate and change many disparities in the left image according the positive depth jump in RMP. After that, if a pixel still gets different disparities from the two matching phases, we invalidate such match and use the "Fill in the gaps" method introduced in Section III-A.1 to assign a disparity to it.



(a) (b)

Fig. 9. Tsukuba image results (a) new method (b) old method



(a) (b)

Fig. 10. Sawtooth image results (a) new method (b) old method

V. Experimental Results

A. Comparison with traditional methods

Figures labeled (a) in Figure 9 and Figure 10 are results for real image pairs using the new Left Right Matching Scheme and the new Filling the Gaps introduced in this paper. The window size we used is 5×5. Compared with Figure 6, Figure 9 shows that correct detection of both left and right object borders can be achieved. Figures labeled (b) are the results for the same image pairs using traditional Left Right Matching Scheme and traditional Filling the Gaps. The window size is also 5×5.

The percentage of wrongly matched pixels over the entire image and in the depth discontinuity regions [10] for both new and traditional methods are listed in Table I. From this table we can see that the new algorithm performs better in both regions.

TABLE I

PERCENTAGE OF BAD PIXELS FOR BOTH NEW AND OLD METHOD

Image Name	New		Old	
	Entire image	Depth discontinuity	Entire image	Depth discontinuity
Tsukuba	5.29	12.31	9.00	21.31
Sawtooth	4.17	9.82	5.64	21.22
Map	1.48	6.45	4.07	29.44
Venus	5.27	9.74	7.36	12.85

The processing time is a little longer than other correlation-based stereo vision algorithms, partly because of the matching cost we used [11], instead of absolute difference, which is most time efficient.

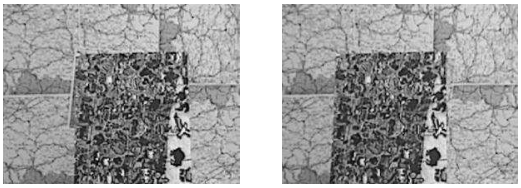


Fig. 11. map image

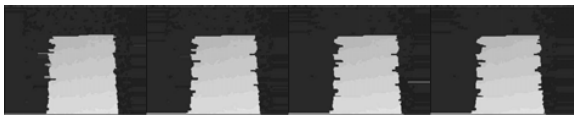


Fig. 12. Results of map image using traditional method when changing widow size from 5×5 to 11×5

B. Comparison with traditional methods when changing window width

The performance, especially at object borders, using traditional correlation method relies heavily on the window size. This can be easily seen from the results of map image(Fig. 11) in Figure 12. Here the window sizes are ranging from 5×5 to 11×5 in steps of 2 pixels. We can see that when the window size becomes wider, the traditional method performs better at the inside-object area; however, it performs worse at object-border area. Figure 13 shows the result of map image using the new Left Right Matching Scheme and new Filling the Gaps introduced in this paper. We can see that when the window size becomes wider, the new method again performs better at the inside-object area, but it does not get worse at object-border area.

VI. Conclusion

Unlike previous correlation-based stereo vision algorithms, which put emphasis on the size and shape of support window for pixels under consideration in different situations, we analyzed the characteristics at each object border or depth jump, based on which we presented a lopsided window scheme accompanied by special matching direction. It is different from normal multiple windowing. Also we present a new idea to assign disparity within invalid areas, which computes the position of depth jump. The experimental results show that the new idea can achieve, with time ef□ciency, correct disparity at object borders, which has not been as successful using other correlation-based stereo vision algorithms.

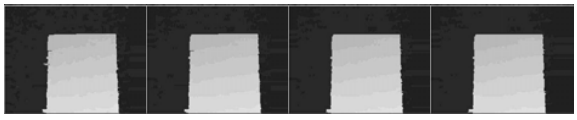


Fig. 13. Results of map image using new method when changing widow size from 5×5 to 11×5

References

[1] H. Hirschmuller, P. R. Innocent, and J. Garibaldi, "Real-time correlation-based stereo vision with reduced border errors," *International Journal of Computer Vision*, vol. 47, pp. 229–246, June 2002.

[2] Y. Boykov, O. Veksler, and R. Zabih, "Variable window approach to early vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 1283–1294, December 1998.

[3] K.-J. Yoon and I.-S. Kweon, "Locally adaptive support-weight approach for visual correspondence search," *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, vol. 2 IS -, pp. 924–931, 2005.

[4] O. Veksler, "Fast variable window for stereo correspondence using integral images," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1 IS -, pp. I/556–I/561, 2003.

[5] A. Fusiello, V. Roberto, and E. Trucco, "Ef□cient stereo with multiple windowing," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition VL - IS -*, pp. 858–863, 1997.

[6] L. D. Stefano, M. Marchionni, and S. Mattoccia, "A fast area-based stereo matching algorithm," *Image and Vision Computing*, vol. 22, pp. 983–1005, October 2004.

[7] J. Sun, Y. Li, S. B. Kang, and H.-Y. Shum, "Symmetric stereo matching for occlusion handling," *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, vol. II, pp. 399 – 406, 2005.

[8] J. Zhao and J. Katupitiya, "A fast stereo vision algorithm with improved performance at object borders," *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, vol. 1 IS -, 2006.

[9] K. Muhlmann, D. Maier, J. Hesser, and R. Manner, "Calculating dense disparity maps from color stereo images, an ef□cient implementation," *International Journal of Computer Vision*, vol. 47, no. 1-3, pp. 79 – 88, 2002.

[10] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision*, vol. 47, pp. 7–42, June 2002.

[11] S. Birch□eld and C. Tomasi, "Depth discontinuities by pixel-to-pixel stereo," *International Journal of Computer Vision*, vol. 35, pp. 269–293, December 1999.