Stereo matching via selective multiple windows

Satyajit Anil Adhyapak
Nasser Kehtarnavaz
Mihai Nadin
University of Texas at Dallas
Department of Electrical Engineering
Richardson, Texas 75080
E-mail: kehtar@utdallas.edu

Abstract. *Window-based correlation algorithms are widely used for stereo matching due to their computational efficiency as compared to global algorithms. In this paper, a multiple window correlation algorithm for stereo matching is presented which addresses the problems associated with a fixed window size. The developed algorithm differs from the previous multiple window algorithms by introducing a reliability test to select the most reliable window among multiple windows of increasing sizes. This ensures that at least one window is large enough to cover a region of adequate intensity variations while at the same time small enough to cover a constant depth region. A recursive computation procedure is also used to allow a computationally efficient implementation of the algorithm. The outcome obtained from a standard set of images with known disparity maps shows that the generated disparity maps are more accurate as compared to two popular stereo matching local algorithms. © 2007 SPIE and IS&T.* [DOI: 10.1117/1.2711817]

1 Introduction

Stereo matching is used to generate disparity or depth maps for applications such as terrain mapping, robotics, and virtual studios. Generally, depth information is obtained from two broad categories of stereo matching algorithms: global and local.[1] Global algorithms, for example, those described in Refs. 2–9, yield accurate disparity maps but involve high computational costs. On the other hand, local algorithms, for example, those described in Refs. 10–17, are computationally efficient but do not produce results as accurately as global algorithms. This paper introduces a local algorithm that generates higher-accuracy disparity maps as compared to the commonly used local algorithms.

Local or area-based algorithms employ correlation techniques to calculate the disparity between a left and a right image. The disparity is calculated by determining a measure of similarity between the pixels within a window in the two images. In local algorithms, cross-correlation (CC),[19] sum of squared differences (SSD),[10,11,16,28] and sum of absolute differences (SAD)[12,13] are the most widely used techniques. However, in these techniques, the selection of window size plays a major role in determining the quality of the resulting disparity map.[14] This is because a fixed window size does not yield reliable disparity estimates for all the pixels in a stereo pair of images. The empirical selection of window size results in two major problems: a noisy disparity map if the selected window is small, covering a region of insufficient intensity variations, and a smoothed disparity map or boundaries if the selected window is too large, covering a region of varying disparities.

This paper presents a new multiple windows approach for stereo matching that allows us to correct the above problems. The most reliable disparity estimate is selected on the basis of quantitative scores obtained from SSD instead of the more commonly used *winner-takes-all* approach.[1] Before the developed algorithm is described in detail, an overview of similar stereo matching algorithms is mentioned in Section 2. A description of the standard area-based matching using the normalized SSD is then discussed in Section 3. In Section 4, the reason for using multiple windows of increasing sizes is mentioned, followed by a test, named the reliability factor, to select the most reliable disparity estimate from multiple disparity estimates. An efficient computation procedure is also discussed in this section. The experimental results are presented in Section 5 together with a comparison of the developed algorithm with two popular local algorithms, namely symmetric multi-window (SMW)[10] and single matching phase (SMP).[12] Finally, the conclusions are stated in Section 6.

2 Overview of Previous Algorithms

Local algorithms normally use window-based correlation to extract depth information from images. Generally, square or rectangular windows are used due to their ease of implementation.[5,10–13,16,17] However, the reliability of depth information is severely affected when a single window of fixed size is used.[14] For this reason, Kanade and Okutomi[14] proposed an adaptive window solution. They modified the window size and shape adaptively depending on the local intensity and disparity variations. Although this algorithm produced better results than the standard single-window algorithms, its final output depended on the choice of the initial disparity estimate. Also, as observed by Fusiello *et al.*,[10] this algorithm did not perform well in occluded regions due to not utilizing the *uniqueness constraint*.[15] Boykov *et al.*[7] also developed an adaptive window algorithm such that the shape of the window varied from pixel to pixel. This algorithm was considerably faster than the one in Ref. 14 as it did not employ an iterative scheme to compute disparities. However, the improvement in the accuracy of the final disparity map was not significant.

Paper 05171RR received Sep. 23, 2005; revised manuscript received Aug. 29, 2006; accepted for publication Oct. 6, 2006; published online Mar. 1, 2007.
1017-9909/2007/16(1)/013012/14/$25.00 © 2007 SPIE and IS&T.

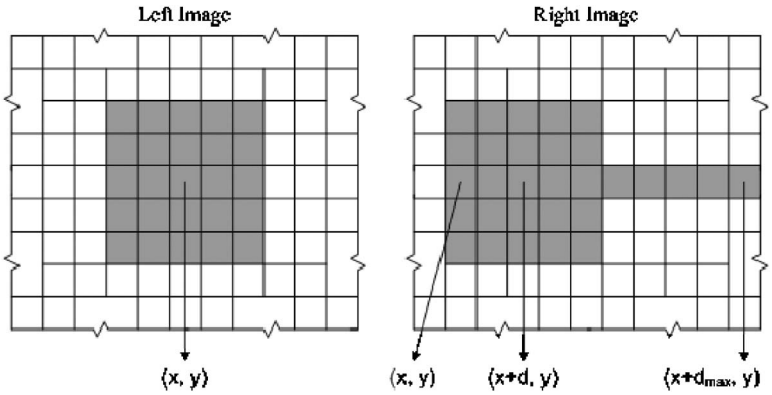


Fig. 1 Stereo matching by shifting correlation windows.

Fusiello *et al.*[10] and Jeon *et al.*[11] used multiple windows to overcome the drawbacks of single-window methods without using an adaptive scheme. The former algorithm employed nine windows, each with a different center. Although the use of multiple windows with different centers ensured that at least one window covered a constant-depth region, the empirically selected fixed window size did not generate accurate disparity estimates when the matching was applied to low textured images. The latter algorithm used eight windows to preserve edge information, thus eliminating the blurring of boundaries. The windows were expanded uniformly in all directions. This demanded a high computational cost because of the use of eight simultaneously expanding windows. Both algorithms used the SSD correlation and selected the disparity estimate of the window giving the least SSD, i.e., the winner-takes-all approach.

Efforts were also made to get improved results with a single window by including certain modifications. The bidirectional matching algorithm introduced by Fua[16] addressed the low textured region problem by identifying and removing wrong matches. In this algorithm, every pixel in the left image was first matched to its best match in the right image. Then the images were reversed and the matching was repeated. Finally, the uniqueness condition was checked to mark unmatched pixels. Stefano *et al.*[12] discussed a matching algorithm that produced disparity maps of more or less the same quality with respect to the bidirectional matching algorithm by carrying out the matching process only once. In their algorithm, older matches were rejected when more reliable matches were found, thus satisfying the uniqueness constraint.[15] Although this considerably improved the matching efficiency, the window size used was still selected empirically, which sometimes resulted in loss of details in disparity maps, especially in those images containing small and fine objects. Muhlmann *et al.*[13] described an efficient algorithm for stereo matching of color images. They showed that the color information could improve the quality of disparity estimates in low textured regions.

In general, local algorithms are plagued by the problem of blurring of edge boundaries, known in the literature as *boundary overreach* or *border localization*.[17,22] Okutomi *et al.*[17] addressed this problem with the help of a multibaseline stereo algorithm[18] and using multiple windows described in Ref. 10. This problem occurs when correlating windows overlap depth discontinuities. Hirschmüller *et al.*[22] introduced a border correction filter to improve matches at object borders. The overall reliability of matches was improved by using an error correction filter and multiple supporting windows. This algorithm can be considered to belong to the real-time class of algorithms. The algorithms developed by Faugeras *et al.*,[19] Kanade *et al.*,[20] and Forstmann *et al.*[21] are some of the other real-time algorithms that deployed a window-based approach.

Global algorithms have been developed to deal with the problems associated with local algorithms. These algorithms, such as the ones described in Refs. 2–5, remove the dependency of the disparity map on the window size. Geiger *et al.*[5] and Veksler[6] used shifting windows to compute a matching cost and then a global optimization method to find the disparity map. Global algorithms rely on the minimization of a global cost function, thereby satisfying most of the constraints imposed by the stereo geometry.[1] Due to their global support nature, these algorithms provide reliable disparity estimates even for regions containing low texture and occluded points.[24] Many global algorithms such as graph cuts,[2] belief propagation,[8] and maximum flow[9] generate dense and highly accurate disparity maps as discussed in Ref. 1. However, due to their high computational costs, their applicability was limited in real-time constraint applications. For example, as discussed in Refs. 19 and 20, only local algorithms were used to achieve computationally efficient implementations.

Lastly, there is another category of algorithms known as the cooperative algorithms. These algorithms use iterative techniques to select the best disparity estimate instead of the winner-takes-all policy. One of the best performing algorithms in this category is developed by Zitnick and Kanade.[23]

3 Area-based Correlation

Area-based correlation is the technique deployed by local algorithms to compute dense disparity maps. In this section, we briefly describe this technique based on a single window to set the stage for our multiple-window algorithm presented in the next section.

Without loss of generality, let us assume that the stereo images are obtained from two cameras with parallel optical

axes. This stereo geometry assumption prevents getting projective distortion and reduces the matching complexity as the search process is limited to one dimension.[26] However, it should be noted that this assumption can be eased by utilizing suitable algorithms, for example, the one in Ref. 27, to make the epipolar lines parallel to image rows.

To compute a disparity value, a window is placed and kept fixed over a specific region in the reference (left) image, while it is shifted horizontally over a finite range in the test (right) image. The range over which the window is shifted is limited by the maximum disparity in the two images. Figure 1 illustrates the window shifting process over the disparity range. The shaded region corresponds to the window placed at a pixel (x,y) in the left image and at $(x+d,y)$ in the right image. The window is shifted only along the x direction in the right image, indicated by the shaded yth row. Correlation is then performed as the window is moved to $(x+d_{\max},y)$, where $d_{\max}$ denotes the maximum disparity. For matching, the normalized SSD function—see Eq. (1)—is used:

$$\mathrm{SSD}_W(x,y,d) = \frac{\sum_{i,j\in W}[\hat{L}(x+i,y+j)-\hat{R}(x+d+i,y+j)]^2}{\sqrt{\sum_{i,j\in W}[\hat{L}(x+i,y+j)]^2}\times\sqrt{\sum_{i,j\in W}[\hat{R}(x+d+i,y+j)]^2}} \tag{1}$$

where

$$\hat{L}(x+i,y+j) = L(x+i,y+j) - \overline{L(x+i,y+j)},$$

$$\hat{R}(x+d+i,y+j) = R(x+d+i,y+j) - \overline{R(x+d+i,y+j)}.$$

W denotes a window of size $(2w+1)\times(2w+1)$, $i, j\in[-w,w]$, and $\hat{L}$ and $\hat{R}$ are the mean subtracted images of the left and right images, respectively. The advantage of using this equation is that it makes the result invariant to any nonuniform lighting by removing the dc component in the images. However, it significantly increases the computational burden.

A pixel in the test image is matched if the correlation window centered on it produces a minimum value as compared to the other values. However, the disparity map so formed exhibits discrete disparity levels, which appear as bands of varying intensities on the disparity map. To diminish such bands and generate a smooth disparity map, a subpixel interpolation procedure is carried out, which involves fitting a curve, e.g., a parabola, to the SSD function in the vicinity of the minimum disparity. That is,

$$d_{\mathrm{sub}} = d_m + \frac{\mathrm{SSD}_W(x,y,d_m-1)-\mathrm{SSD}_W(x,y,d_m+1)}{2(\mathrm{SSD}_W(x,y,d_m-1)-2\mathrm{SSD}_W(x,y,d_m)+\mathrm{SSD}_W(x,y,d_m+1))}, \tag{2}$$

where d_{sub} denotes the subpixel disparity and d_m the disparity producing the minimum SSD value. The subpixel interpolation procedure refines the disparities, i.e., generates a gradual transition from one disparity level to another.

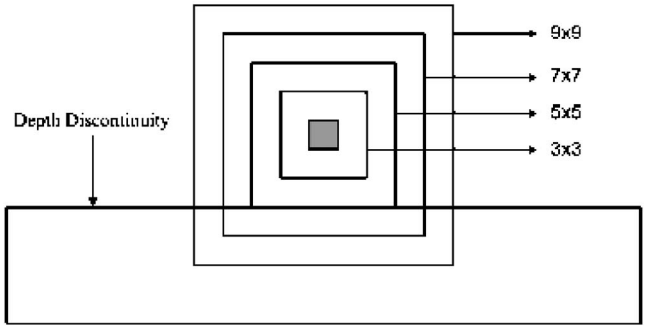


Fig. 2 Multiple windows with increasing sizes.

4 Selective Multiple-Window (SEL) Algorithm

Area-based correlation can be performed using multiple windows. The use of multiple windows allows one to improve the accuracy of the disparity map, albeit at the expense of a higher computational cost. Let us first describe the advantage of using multiple windows.

As illustrated in Fig. 2, consider a possible scenario where four single windows of different sizes are used to match an arbitrary point near a depth discontinuity, where there is a higher chance of incorrect matching due to either occlusions or windows covering a nonconstant disparity region. The windows shown are considered to be of size 3×3, 5×5, 7×7, and 9×9, respectively. Reliable disparity estimation requires the windows to cover a region of sufficient intensity variations as well as constant disparity. Assume that out of the four windows, only the size 3×3 and 5×5 windows satisfy the above condition, that is, the size 7×7 and 9×9 windows go across the depth discontinuity. The disparity estimate obtained from the 3×3 window may not be as accurate as the 5×5 window due to the presence of noise. However, this does not imply that the 5×5 window would yield reliable estimates for all the points. In other words, at another point, a different window size could perform better. This makes the selection of the window size of critical importance. The idea here is to utilize an adaptive scheme to determine an appropriate window size automatically. Since conventional adaptive techniques, such as the one described in Ref. 14, are computationally inefficient, the focus of this work has been on a computationally efficient multiwindow technique capable of yielding accurate disparity maps.

In our multiple-window algorithm, the windows grow in size progressively, keeping their center pixel fixed, unlike Fusiello *et al.*'s[10] algorithm, which uses windows with different center pixels. The fixed-center pixel approach allows having a uniform contribution from top, bottom, left, and right pixels in the computation of correlation. Apart from the position of the center pixel, three other issues need to be addressed here: (1) the number of windows to use; (2) the criterion for selecting a reliable disparity among the windows; and (3) the computational complexity. The first issue can be addressed by making the largest window equal to the maximum disparity ($d_{\max}$) and the smallest window of size 3×3.

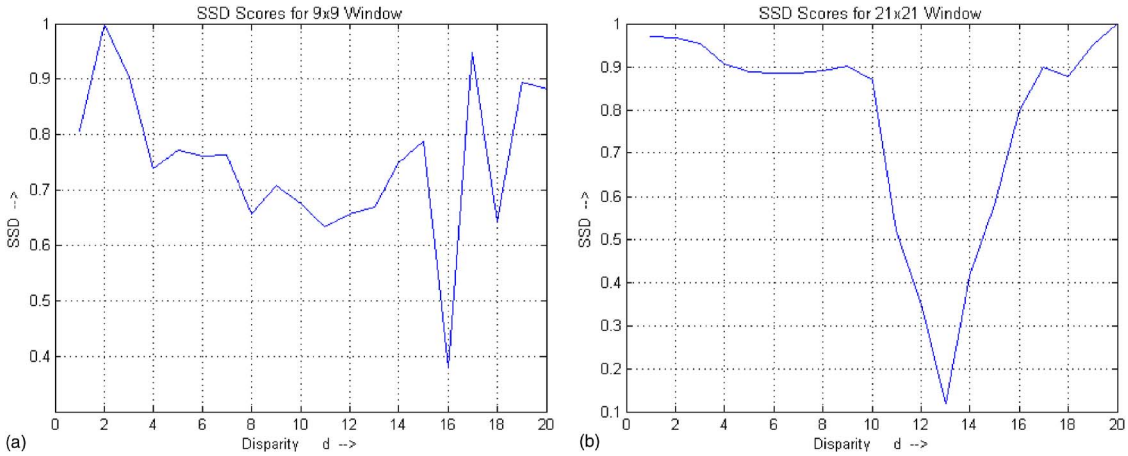


Fig. 3 SSD curves of a point in the Venus image pair (see Fig. 12).

The reason behind this choice is the requirement of local algorithms to use windows covering a constant disparity region. For an arbitrary point, not lying on a depth discontinuity, with disparity d, the constant disparity region must extend to at least d pixels. Since the search for the correct disparity is limited to $d_{\max}$, the extent of this constant disparity region is therefore upper-bounded by $d_{\max}$ pixels. This dictates the selection of the size of the largest window equal to the maximum disparity. This disparity is identified by inspection. That is to say, we identify some distinct points on the foreground and background and find the shift between those points in the two images. This way the disparity range is approximated. $d_{\max}$ is found from the foreground object closest to the camera and $d_{\min}$ from the background object farthest from the camera. The drawbacks of this kind of a window selection process include the generation of incorrect disparity estimates for points near a depth discontinuity and high computational complexity. The former brings us to the second issue, i.e., how to select a reliable disparity estimate. This is a challenging task in the absence of any prior knowledge about the nature of the image. The methods to select a reliable estimate from multiple windows and reduce computational complexity are explained in detail in the subsections that follow.

4.1 *Reliability Test*

Multiple windows result in multiple disparity estimates out of which the most reliable estimate needs to be selected. The selection cannot be based on the window giving the least SSD value, as done in Refs. 10 and 11, because the windows are of different sizes, and the smallest window always yields the smallest SSD value. Another method to select the disparity estimates of multiple windows is to normalize the error score by dividing it by the number of pixels in the window. This approach will remove the dependence on window size, but the selection of the disparity is still based on the winner-takes-all policy and thus suffers from the drawbacks associated with it.[1,28] Hence, the decision must be made by taking into account the nature of the SSD curve for each window over the entire disparity range. For this reason, a quantitative test is introduced here that analyzes the SSD curve for each window to assign a weight, named the *reliability factor* (RF), to the disparity estimate of that window. The estimate corresponding to the largest RF is then selected. This factor performs refining disparities similar to the refinement reported in Ref. 25 with one difference—the reliability calculations are part of the algorithm and not a postprocessing step.

Figure 3 shows the SSD curves for a sample point lying in a region of low-intensity variations using two windows: one small (9×9) and one large (21×21). The correct disparity for the point is 13 pixels, and the SSD curves are normalized with respect to the maximum SSD value. Matching is affected by the presence of regions having low or repetitive texture. Such regions generally produce jagged curves with large peak-to-peak variations, as shown in Fig. 3(a). To identify and discard such estimates, the RF is made proportional to the local variation around the minimum value. Hence, if there is a high local peak-to-peak variation, the corresponding RF would be small. The local variation (lv) is defined, similar to the one defined in Ref. 28 as follows:

$$lv = \sum_{k\in E}\left[\frac{e(k)-e(k-1)}{\max\limits_{k\in E} e(k) - \min\limits_{k\in E} e(k)}\right]^2; \quad E = d_m - 2{:}d_m + 2, \tag{3}$$

where e denotes the SSD score, d_m is the position of the minimum SSD score, and E is a five-pixel-wide region around the position of the minimum. The denominator represents the difference between the maximum and minimum values in this five-pixel region.

In order to assign high weights to the estimates of the windows with a distinct global minimum [see Fig. 3(b)], the following factor, e_d, is included that signifies the distinctiveness of the global minimum:

$$e_d = \sum_{i=1}^{nlm}(e_i - e_m), \tag{4}$$

where nlm denotes the number of local minima and e_i a local minimum. Moreover, RF is made inversely proportional to the number of local minima, noting that the ambiguity in a disparity estimate increases with an increase in

the number of local minima. Hence, the expression used to compute the reliability factor is given by Eq. (5),

$$\mathrm{RF} = \frac{e_d}{nlm} \sum_{k \in E} \left[\frac{e(k) - e(k-1)}{\max\limits_{k \in E} e(k) - \min\limits_{k \in E} e(k)} \right]^2. \tag{5}$$

The window with the highest RF is selected, and the disparity estimate associated with it is taken as the disparity of the current pixel.

Points near a depth disparity in one image are generally occluded in the other image. Finding a match for such points is difficult and often left unmatched. To reliably identify points near a depth discontinuity, first the variance is computed for all windows. From our experimentations, it is observed that the variance of points near depth discontinuities exhibits a sudden rise or a peak. Figure 4 shows the variance plots of a row of the Venus image for increasing window sizes. This can be attributed to the fact that at this point the window covers a region of a depth discontinuity. As a result, the window has contributions from the region in the background as well as the region in the foreground. The magnitude of the peak depends on the area of each region covered by the window. A valid peak is considered to be the one with magnitude greater than 0.5. Then, the consistency of the locations of these peaks is determined. For instance, at pixel 150 near a depth discontinuity, there is a peak seen in Figs. 4(c) and 4(d); however, no peak is found at that location in Figs. 4(a) and 4(b). Thus, the variances for pixel 150, computed using all windows, exhibits a step-like pattern: low values for windows covering a constant disparity region and a sudden step change for windows covering varying disparity regions. This suggests that the window used in Fig. 4(b) is the largest window that will give a reliable estimate of disparity at this point. Note that at pixel 200, which is sufficiently away from the depth discontinuity, all windows exhibit a low value. Finally, the window sizes are compared based on the RF and the variance check. If the window given by the variance check is already discarded by the RF, this point is marked as unmatched. Thus, it is important to note that our algorithm explicitly identifies occluded pixels near a depth discontinuity.

4.2 *Recursive Computation*

Since the number of windows used for correlation depends on the disparity range, for images with large disparity ranges, the algorithm becomes computationally expensive. To overcome this problem, the recursive technique introduced in Ref. 12 is utilized and extended here to achieve an efficient computation of SSD by eliminating the dependency of the computation on the window size.

Before proceeding to the analysis of the recursive computation, let us rewrite Eq. (1) in a simpler form as follows:

$$\mathrm{SSD}_W(x,y,d) = \frac{N_W(x,y,d)}{D1_W(x,y) \times D2_W(x,y,d)},$$

$$N_W(x,y,d) = \sum_{i,j \in W} [\hat{L}(x+i,y+j) - \hat{R}(x+d+i,y+j)]^2,$$

$$D1_W(x,y) = \sqrt{\sum_{i,j \in W} [\hat{L}(x+i,y+j)]^2},$$

$$D2_W(x,y,d) = \sqrt{\sum_{i,j \in W} [\hat{R}(x+d+i,y+j)]^2}. \tag{6}$$

The recursive procedure is now described with respect to the numerator, as the denominators can be computed by applying the same process with slight modifications. Once the numerator and denominators are computed, the correlation score can be easily obtained.

Consider a single window of size $W=(2w+1)\times(2w+1)$ positioned at the coordinates (x,y) in the reference image and at $(x+d,y)$ in the test image. When the window is shifted from a point $(x,y-1)$ to a point (x,y), the SSD at the new point can be computed from the SSD at the old point as stated below:[12]

$$N_W(x,y,d) = N_W(x,y-1,d) + RD_W(x,y,d), \tag{7}$$

where

$$RD_W(x,y,d) = \sum_{i=-w}^{w} [\hat{L}(x+i,y+w) - \hat{R}(x+d+i,y+w)]^2 - \sum_{i=-w}^{w} [\hat{L}(x+i,y-1-w) - \hat{R}(x+d+i,y-1-w)]^2,$$

$RD_W(x,y,d)$ denotes the difference between the SSD of the $(y+w)$th and $(y-1-w)$th rows, and $\hat{L}$, $\hat{R}$ are the mean subtracted images of the reference and test images, respectively.

To have a better understanding of Eq. (7), Fig. 5 provides a graphical description of it. When the window, indicated by the thick black lines, is moved one pixel down, the pixels along the row $(y+w)$ (shaded dark) are the only ones that get included in the area enclosed by the window, those along the row $(y-1-w)$ (shaded light) are left out. Hence, the SSD value at the previous point can be updated based on the SSD difference of these two rows. This immediately reduces the number of operations per window from $(2w+1)^2$ to $(2w+1)$, where each operation is the squared difference of the pixel intensity in the reference and test image.

Next, when the window is shifted horizontally, it is observed that the SSD value of each row can be computed by adding the difference of the SSD values of only two pixels to the previous value. The two pixels correspond to the one included in the row and the one excluded when the window shifts by one pixel. The result of this operation corresponds to the second level of recursion, which computes the row difference $RD_W(x,y,d)$ from the row difference $RD_W(x-1,y,d)$, as illustrated in Fig. 6. The lightly shaded pixels

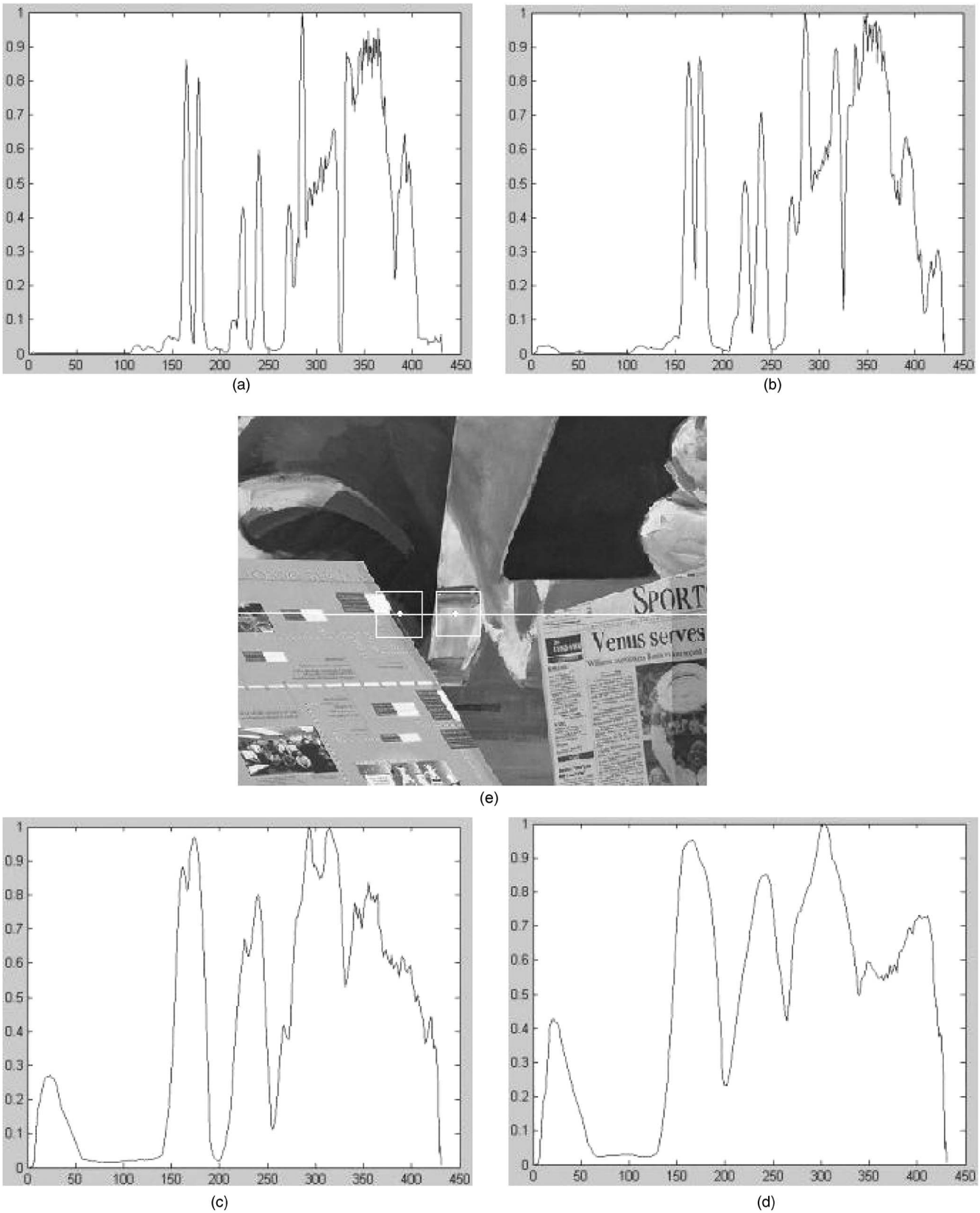


Fig. 4 Determining occluded pixels using variance of pixels.

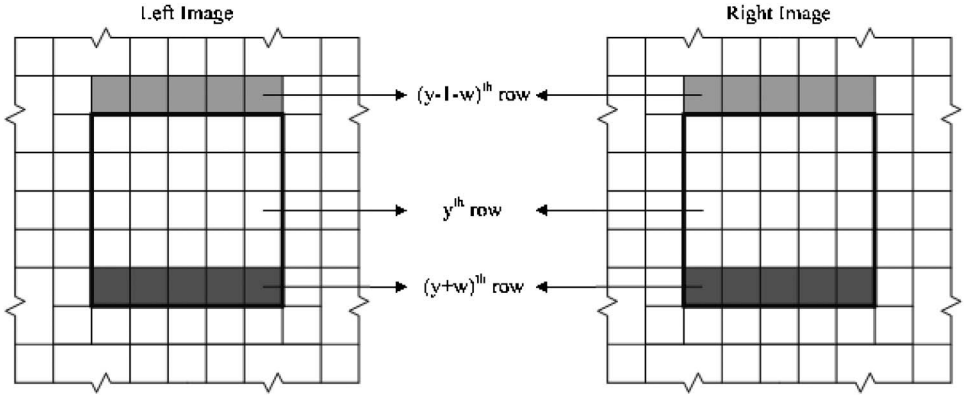


Fig. 5 First stage of recursive computation.

represent the pixels excluded when the window moves to its current position. Equation (8) provides the second stage of the recursive computation:[12]

$$\begin{aligned} RD_W(x,y,d) = RD_W(x-1,y,d) &+ [\hat{L}(x+w,y+w) \\ &- \hat{R}(x+d+w,y+w)]^2 - [\hat{L}(x-1-w,y+w) \\ &- \hat{R}(x-1+d-w,y+w)]^2 \\ &- [\hat{L}(x+w,y-1-w) \\ &- \hat{R}(x+d+w,y-1-w)]^2 \\ &+ [\hat{L}(x-1-w,y-1-w) \\ &- \hat{R}(x-1+d-w,y-1-w)]^2. \end{aligned} \tag{8}$$

From Eq. (8), we can see that the correlation score of each pixel can be computed in only four operations, where each operation is the squared difference of the pixel intensity in the reference and test images. However, to initiate the recursion, the SSD value of the first pixel must be computed in a direct manner. This implies that for each window, the SSD value of the first pixel must be computed before the process is initiated. Thus, a third stage of recursion is introduced here to reduce the computational cost arising from the above operation. Matching begins when the largest window is placed over a particular pixel and the SSD associated with it is computed. When the window size is reduced by one, the SSD computation for the new window ($W1$) is performed by simply subtracting the SSD values of the two outermost pairs of rows and columns (Wo) from the SSD value of the larger window (W) as illustrated in Fig. 7. Equation (9) provides the third stage of the recursive computation:

$$\begin{aligned} N_{W1}(x,y,d) = N_W(x,y,d) &- \sum_{i=-w}^{w} [\hat{L}(x+i,y-w) \\ &- \hat{R}(x+d+i,y-w)]^2 - \sum_{i=-w}^{w} [\hat{L}(x+i,y+w) \\ &- \hat{R}(x+d+i,y+w)]^2 - \sum_{j=-w+1}^{w-1} [\hat{L}(x-w,y+j) \\ &- \hat{R}(x+d-w,y+j)]^2 \times \sum_{j=-w+1}^{w-1} [\hat{L}(x+w,y \\ &+ j) - \hat{R}(x+d+w,y+j)]^2. \end{aligned} \tag{9}$$

For a single window, after the first pixel, the recursion makes the matching process almost independent of the window size. For multiple windows, the recursive computation lowers the complexity, in terms of the number of SSD op-

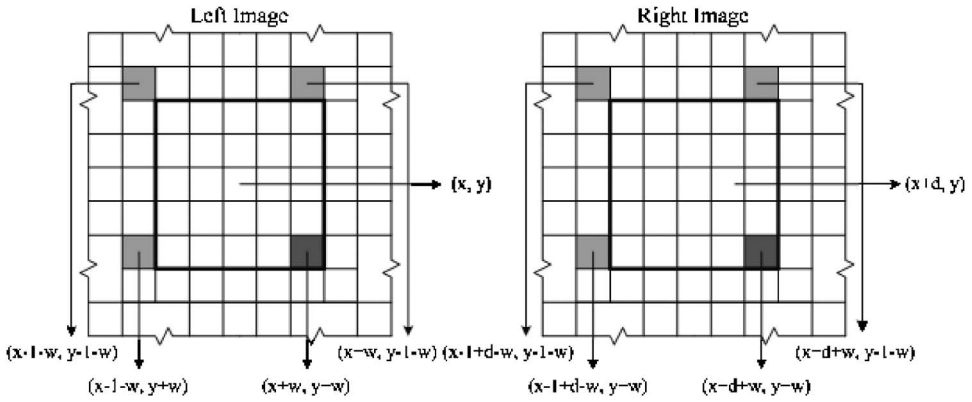


Fig. 6 Second stage of recursive computation.

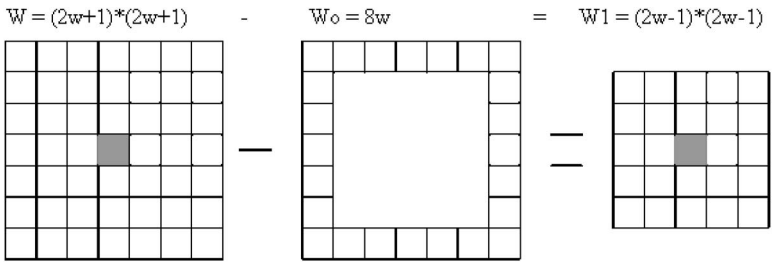


Fig. 7 Third stage of recursive computation.

erations, from $O[4MN*d_r*\Sigma_{i=1}^{d_{\max}} w_i^2]$ when no recursion is used to $O[4MN*d_r*d_{\max}]$ when using the recursions, where MN represents the number of pixels and d_r the disparity range. For example, on a Pentium 4, 1 GHz PC, for the Tsukuba image pair of size 288×384 (see Fig. 8), the recursive computation reduced the processing time from 1066 seconds to 22 seconds; that is a speed up by a factor of 49.

4.3 *Algorithm Summary*

Basically, our stereo matching algorithm consists of the following four steps:

1. The input images are first compensated for photometric distortions by subtracting the means instead of the Laplacian-of-Gaussian (LoG) approach.[14,20] In this step, the variance is also computed, to be used during the reliability test. The input images are assumed to be rectified so that the disparity only varies in the horizontal direction.
2. Stereo matching is carried out using the SSD correlation and multiple windows as described in Sections 3 and 4. The number of windows depends on the disparity range between the two stereo images. The maximum window size is selected to be equal to the largest disparity. This means $d_{\max}$ number of windows is deployed. This selection is based on the observation that windows larger than $d_{\max}$ do not provide any new information. Furthermore, a window of this size induces an effect of global matching due to its size.
3. Once the matching process using the multiple windows is completed, the most reliable match is found by applying the reliability test. The results of the reliability test and the variances computed in step 1 aid in identifying occluded pixels without the need to perform any bidirectional matching.
4. Finally, a subpixel interpolation is performed on the disparity estimate of the selected window to refine the match and generate a smooth map. A second-degree curve is then used to interpolate the SSD scores in the vicinity of the minimum disparity found by the above steps.

5 Results and Discussion

This section includes the experimental results obtained by our SEL algorithm on a set of standard stereo images from the Middlebury College database[29] with known ground truths. The results are compared with the two popular local algorithms, symmetric multi-window (SMW) and single matching phase (SMP), to illustrate the effectiveness of the SEL algorithm.

A preprocessing step was carried out, which consisted of subtracting the image means to make the matching process invariant to different lighting conditions seen by the cameras. Some methods, for example, Refs. 14 and 20, use Laplacian-of-Gaussian filtering for such a preprocessing. The displayed results consist of six images: left input image, right input image, ground truth disparity map, disparity map generated by the SEL algorithm, disparity map generated by the SMW algorithm, and, finally, disparity map generated by the SMP algorithm. For the SMW and SMP algorithms, the images shown correspond to a 9×9 correlation window, although the comparison tables provided here include the outcomes for three window sizes: 7×7, 9×9, and 11×11.

The disparity maps of the Tsukuba stereo image pair are shown in Fig. 8. This image pair contains objects at varying depths and regions with low and repetitive texture. Comparing the map with the ground truth, it is observed that the SEL algorithm recovers the disparity in the images well. Even the disparity of fine objects, such as the legs of the tripod, the handle of the camera, and the lamp wire, is recovered. However, the map contains some wrong matches caused by occlusions and poor texture. These points are represented in white for visual identification. On the other hand, the SMW and the SMP algorithms recover the overall 3D structure of the objects but exhibit more errors in recovering finer objects. The SMW algorithm recovers some of the fine objects, but some others are com-

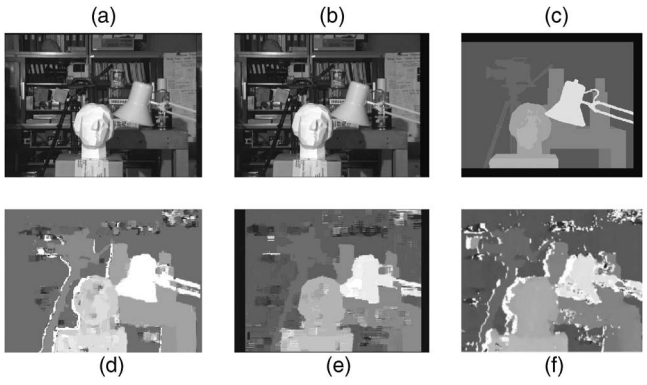


Fig. 8 Tsukuba image pair: (a) left image; (b) right image; (c) ground truth; disparity maps using (d) SEL; (e) SMW; and (f) SMP algorithms.

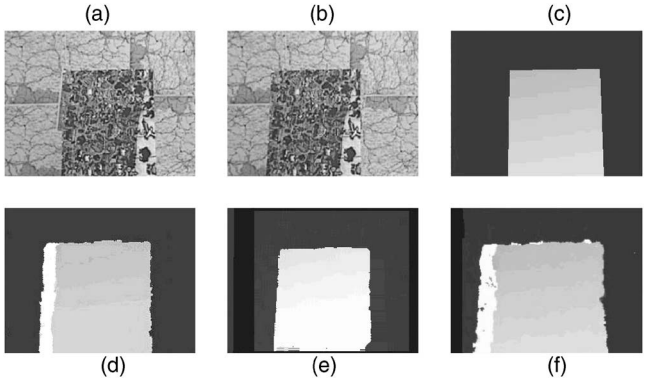


Fig. 9 MAP image pair: (a) left image; (b) right image; (c) ground truth; disparity maps using (d) SEL; (e) SMW; and (f) SMP algorithms.

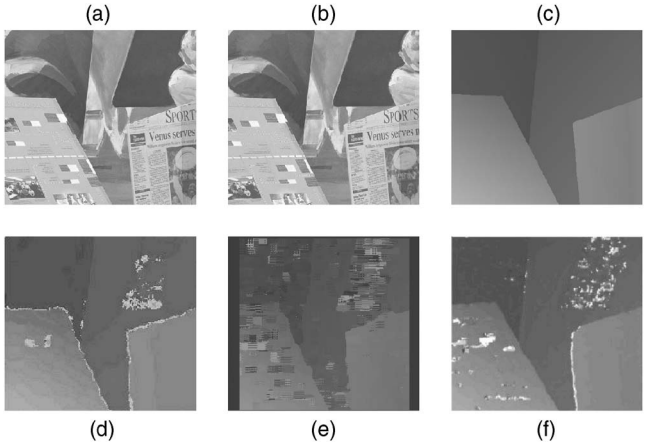


Fig. 12 Venus image pair: (a) left image; (b) right image; (c) ground truth; disparity maps using (d) SEL; (e) SMW; and (f) SMP algorithms.

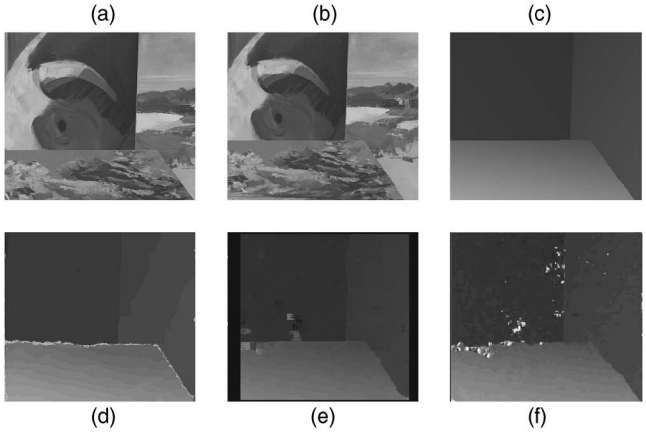


Fig. 10 Bull image pair: (a) left image; (b) right image; (c) ground truth; disparity maps using (d) SEL; (e) SMW; and (f) SMP algorithms.

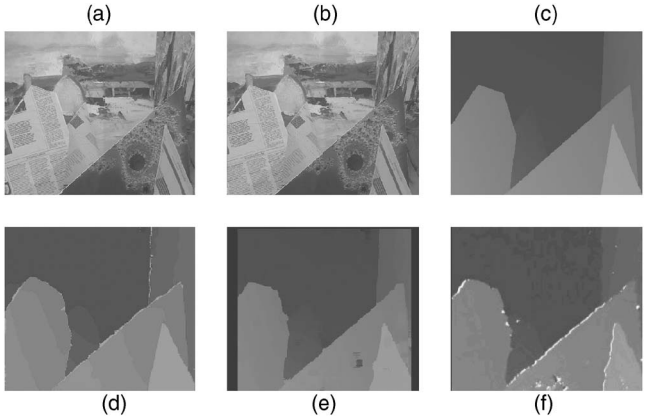


Fig. 13 Barn1 image pair: (a) left image; (b) right image; (c) ground truth; disparity maps using (d) SEL; (e) SMW; and (f) SMP algorithms.

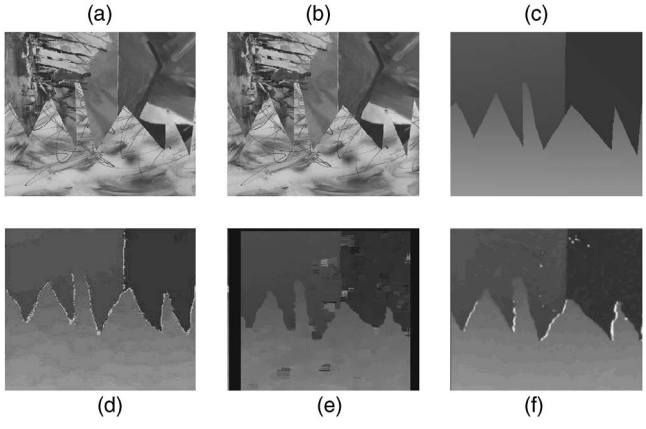


Fig. 11 Saw image pair: (a) left image; (b) right image; (c) ground truth; disparity maps using (d) SEL, (e) SMW, and (f) SMP algorithms.

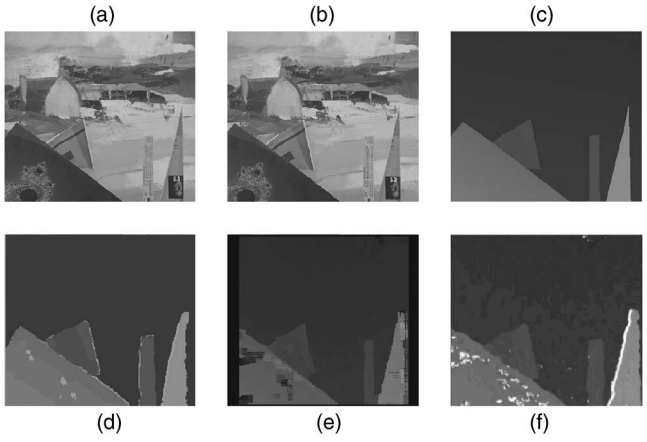


Fig. 14 Barn2 image pair: (a) left image; (b) right image; (c) ground truth, disparity maps using (d) SEL; (e) SMW; and (f) SMP algorithms.

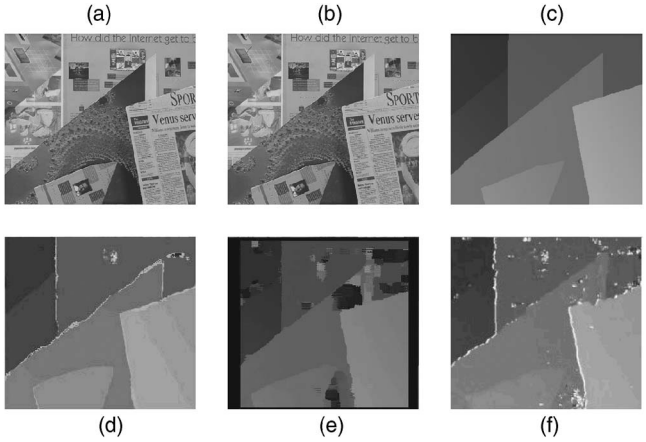


Fig. 15 Poster image pair: (a) left image; (b) right image; (c) ground truth, disparity maps using (d) SEL; (e) SMW; and (f) SMP algorithms.

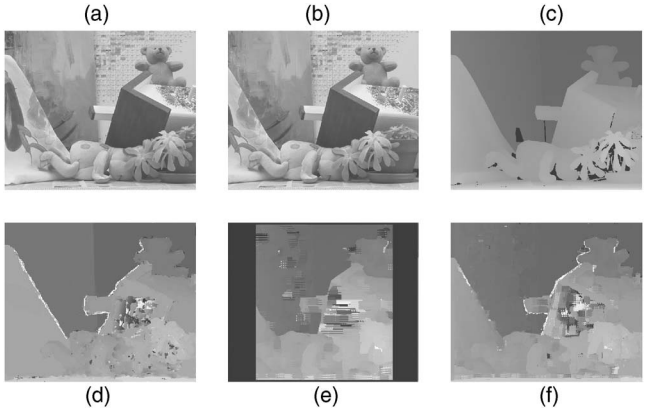


Fig. 16 Teddy image pair: (a) left image; (b) right image; (c) ground truth; disparity maps using (d) SEL; (e) SMW; and (f) SMP algorithms.

pletely missed. The SMP algorithm recovers these objects, too, but at the expense of the border localization problem in addition to the loss of details in some regions.

The experimental results for seven more stereo image pairs are shown in Figs. 9–15: Map, Bull, Saw-tooth, Venus, Barn1, Barn2, and Poster. These images consist of simple objects including paintings and posters placed at different depths. The images have large disparity ranges, resulting in large occlusions. With the exception of the Map image pair, which has a high textured pattern in the middle, they are mostly low in texture. The performance of the three algorithms for these images is quite similar except for the Venus and Poster image pairs, where the SMW algorithm produces poor results. In the Saw and Barn1 images, the border localization problem is quite prominent as compared to the other images for the SMP algorithm. In this case for the window size 9×9, the SMW algorithm performs better than the SEL and SMP algorithms. Due to the large disparity ranges in these images, many points are occluded, especially near the edges. These points constitute the bulk of the unmatched points and are represented in white in the SEL and SMP disparity maps. In the SMW disparity maps, such points are assigned to the disparity of the deeper plane, as done in Ref. 30, and hence are not represented in white.

Considering that the above images are composed of mainly planar objects, two more image pairs were tested: Teddy and Cones. These images are more realistic and offer a more challenging test of the matching algorithm. Figure 16 provides the outcome of the three algorithms for the Teddy image pair, while Figure 17 provides the same for the Cones image pair. From the outcome of the Teddy image pair, we can see that all the algorithms yielded errors during the matching process. There are gross errors on the slanting roof of the house as well as the doll lying on the floor. Due to the large disparity range of this image, the SMW algorithm does not process columns equivalent to the disparity range of this image on both sides, leading to the large black regions on the side. On the contrary, all three algorithms performed well on the Cones image pair. The algorithms recover fine objects like the straws in the cup and the tips of the cones. The SEL algorithm exhibits a better visual outcome than the other two algorithms.

In addition, a quantitative measure is applied to prove the superiority of the SEL algorithm over the SMW and SMP algorithms. This was achieved by using the percentage of correct matches. To do this quantitative assessment, we used the measure defined in Ref. 1 to compute the percentage of correct matches, denoted by CP. To calculate CP, all the pixels in an image were considered so as to provide an absolute measure of accuracy and illustrate the capability of the algorithm to assign correct disparities to pixels. Matches are said to be correct if the absolute difference between the obtained disparity and ground truth is less than or equal to one. Therefore, for pixels near occluding boundaries (shown in white) and unmatched pixels, this difference will be always greater than one. As a result, the percentage of unmatched pixels would be simply 100 minus CP. The following equation was thus used to compute CP between a disparity map D and a ground truth map D_T for an image of size MN:

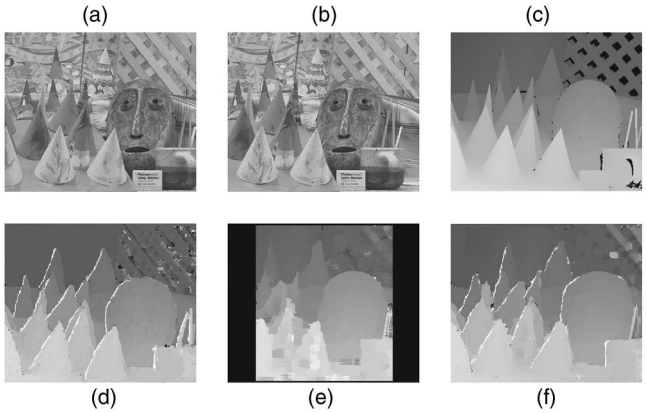


Fig. 17 Cones image pair: (a) left image; (b) right image; (c) ground truth; disparity maps using (d) SEL; (e) SMW; and (f) SMP algorithms.

Table 1 Percentage of correct matches for SEL, SMW, and SMP algorithms.

	SEL		SMW			SMP		
Image	NSSD	SAD	7×7	9×9	11×11	7×7	9×9	11×11
Tsukuba	92.18	92.12	91.04	91.30	91.23	89.26	89.71	89.43
Map	94.67	94.60	94.42	94.52	94.60	90.32	90.56	90.67
Bull	99.51	99.48	98.47	99.23	99.27	98.92	99.30	99.38
Saw	97.58	97.51	94.08	95.24	95.46	97.21	97.95	98.06
Venus	97.45	97.41	90.24	91.63	92.84	97.31	97.79	97.80
Barn1	98.87	98.86	95.61	95.73	95.80	97.91	98.18	98.23
Barn2	97.79	97.75	96.49	97.03	97.17	96.34	96.72	96.74
Poster	98.46	98.43	92.47	92.84	93.23	97.40	97.84	97.96
Teddy	83.59	83.38	76.73	77.81	78.45	80.86	81.63	82.08
Cones	92.22	92.18	82.24	83.48	85.21	87.57	89.03	89.77

$$\mathrm{CP} = \frac{1}{MN}\left(\sum_{y=0}^{M-1}\sum_{x=0}^{N-1}\delta d(x,y)\right) * 100, \tag{10}$$

where

$$\delta d(x,y) = 1 \quad if\ |D(x,y) - D_T(x,y)| \leqslant 1,$$
$$= 0 \qquad\qquad \text{otherwise}.$$

Table 1 lists the computed percentages corresponding to the three algorithms using different window sizes. In addition, the root mean square (RMS) error measure was computed. This measure reflects the overall error in a disparity map. Equation (11) was used to compute the RMS error between a disparity map D and a ground truth map D_T for an image of size MN:

$$\mathrm{RMS} = \sqrt{\frac{1}{MN}\sum_{y=0}^{M-1}\sum_{x=0}^{N-1}|D(x,y) - D_T(x,y)|^2}. \tag{11}$$

Table 2 lists the RMS values corresponding to the three algorithms using different window sizes. To evaluate the above metrics, we considered all the pixels in the images. However, the ground truth for the Tsukuba image pair has

Table 2 RMS errors for SEL, SMW, and SMP algorithms.

	SEL		SMW			SMP		
Image	NSSD	SAD	7×7	9×9	11×11	7×7	9×9	11×11
Tsukuba	2.89	2.96	3.21	3.02	3.12	4.68	4.41	4.56
Map	2.31	2.33	2.43	2.39	2.38	3.49	3.42	3.37
Bull	0.48	0.51	0.67	0.53	0.51	0.58	0.51	0.49
Saw	0.68	0.72	1.27	1.17	1.12	0.79	0.66	0.62
Venus	0.86	0.90	1.63	1.49	1.33	0.93	0.84	0.81
Barn1	0.65	0.68	1.15	1.13	1.11	0.78	0.72	0.67
Barn2	0.87	0.92	1.47	1.41	1.40	0.97	0. 91	0.89
Poster	0.72	0.74	1.46	0.95	0.90	0.89	0.80	0.75
Teddy	2.59	2.57	4.12	4.07	4.11	2.71	2.67	2.59
Cones	1.79	1.84	4.36	4.19	4.12	2.89	2.75	2.62

Table 3 Processing times (speedup factors) for SEL, SMW, and SMP with respect to NSSD SEL without recursion.

Image	NSSD SEL (without recursion) sec	SEL		SMW 9×9	SMP 9×9
		NSSD	SAD		
Tsukuba	1066.3	21.8 (49)	15.9 (67)	11.3 (94)	7.6 (140)
Map	1164.4	36.4 (32)	20.4 (57)	14.5 (80)	10.2 (114)
Bull	1280.7	53.3 (24)	34.6 (37)	24.6 (52)	15.4 (83)
Saw	1265.2	48.6 (26)	31.6 (40)	23.8 (53)	14.7 (86)
Venus	1303.6	54. (24)	36.2 (36)	25.1 (52)	16.5 (79)
Barn1	1256.3	36.9 (34)	26.1 (48)	22.0 (57)	14.4 (87)
Barn2	1270.5	41.0 (31)	25.4 (50)	21.9 (58)	14.7 (86)
Poster	1283.1	53.4 (24)	37.7 (34)	25.1 (51)	16.4 (78)
Teddy	3214.3	178.5 (18)	133.9 (24)	100.4 (32)	65.6 (49)
Cones	3187.7	185.5 (17)	138.6 (23)	99.6 (32)	67.8 (47)

18 rows removed from the top and bottom and 18 columns removed from its sides. This amounts to 11,772, or 10.6%, of the total pixels that are not considered for the accuracy test. However, in our case we have considered all the pixels. Since the border pixels correspond to the background, we have extended the value of the background pixels by these 18 pixels in all directions and used it to compute the accuracy results. In other words, more pixels are used, which is the reason for the low value of the percentage of correct matches. Thus, the results in the table provide a true measure of the accuracy of the algorithms. From the above tables, it is evident that the developed algorithm outperforms the SMW and SMP algorithms with the exception of the two image pairs Saw and Venus for the window sizes 9×9 and 11×11, although the outcomes for these images are quite close. Furthermore, most importantly, it should be noticed that the outcome of the SEL algorithm is not dependent on the window size.

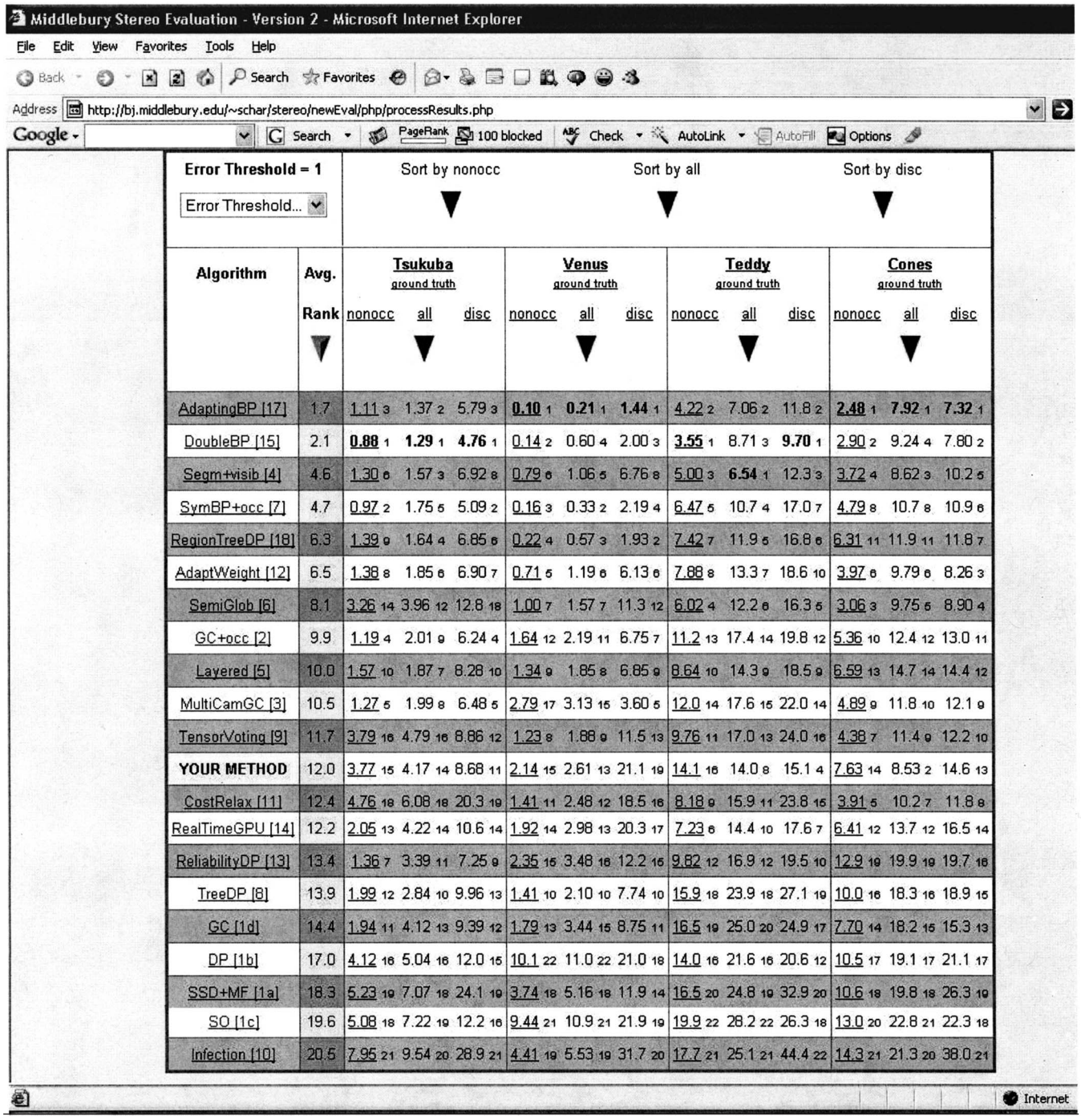
Table 3 lists the running times of the three algorithms on a Pentium 4, 1 GHz PC. It is important to know that all the implementations are in Matlab, which is the reason for the higher run-time speeds as compared to some other algorithms. For example, the implementations by Refs. 12, 13, and 22, make use of optimized C and multimedia extensions (MMX) available in most Pentium processors. From the table, it can be seen that the SAD-SEL reduces the processing times considerably as compared to the processing times of the NSSD-SEL algorithm. Furthermore, the SAD-SEL algorithm yields processing times comparable to those of the SMW and SMP algorithms, which is attributed to the three-level recursive computational scheme.

Finally, in an effort to compare the developed algorithm with other local algorithms, the results of the Tsukuba, Venus, Teddy, and Cones image pairs were submitted to Middlebury College for evaluation. The only local algorithm reported as part of this evaluation is the SSD-MF algorithm.[1] Table 4 lists the percentage of error in the three regions identified in Ref. 1. As observed from Table 4, the SEL algorithm outperformed the SSD-MF algorithm. In addition to the above comparison, the SEL algorithm was compared to the algorithms on the Middlebury Stereo Vision Research page. Table 5 provides the results obtained from this page. As seen from this table, the SEL algorithm was listed as the 12th-best performing algorithm and the 2nd-best performing *local* or correlation-based algorithm behind that in Ref. 31. However, in terms of computational complexity, the SEL algorithm is more efficient. All the other algorithms that fared better were *global* algorithms, and none was as computationally efficient.

Table 4 Local algorithm outcome from the Middlebury College stereo page evaluation; N (non-occlusion), A (all), and D (discontinuity).

	Tsukuba			Venus			Teddy			Cones		
Algorithm	N	A	D	N	A	D	N	A	D	N	A	D
SEL	3.77	4.17	8.68	2.14	2.61	21.1	14.1	14.0	15.1	7.63	8.53	14.6
SSD-MF	5.23	7.07	24.1	3.74	5.16	11.9	16.5	24.8	32.9	10.6	19.8	26.3

Table 5 Comparison with the Middlebury College algorithms: SEL is seen as the best-performing local algorithm; the other better-performing algorithms are global algorithms.



Algorithm	Avg. Rank	Tsukuba ground truth nonocc	Tsukuba ground truth all	Tsukuba ground truth disc	Venus ground truth nonocc	Venus ground truth all	Venus ground truth disc	Teddy ground truth nonocc	Teddy ground truth all	Teddy ground truth disc	Cones ground truth nonocc	Cones ground truth all	Cones ground truth disc
AdaptingBP [17]	1.7	1.11 3	1.37 2	5.79 3	**0.10** 1	**0.21** 1	**1.44** 1	4.22 2	7.06 2	11.8 2	**2.48** 1	**7.92** 1	**7.32** 1
DoubleBP [15]	2.1	**0.88** 1	**1.29** 1	**4.76** 1	0.14 2	0.60 4	2.00 3	**3.55** 1	8.71 3	**9.70** 1	2.90 2	9.24 4	7.80 2
Segm+visib [4]	4.6	1.30 6	1.57 3	6.92 8	0.79 6	1.06 5	6.76 8	5.00 3	**6.54** 1	12.3 3	3.72 4	8.62 3	10.2 5
SymBP+occ [7]	4.7	0.97 2	1.75 5	5.09 2	0.16 3	0.33 2	2.19 4	6.47 5	10.7 4	17.0 7	4.79 8	10.7 8	10.9 6
RegionTreeDP [18]	6.3	1.39 9	1.64 4	6.85 6	0.22 4	0.57 3	1.93 2	7.42 7	11.9 5	16.8 6	6.31 11	11.9 11	11.8 7
AdaptWeight [12]	6.5	1.38 8	1.85 6	6.90 7	0.71 5	1.19 6	6.13 6	7.88 8	13.3 7	18.6 10	3.97 6	9.79 6	8.26 3
SemiGlob [6]	8.1	3.26 14	3.96 12	12.8 18	1.00 7	1.57 7	11.3 12	6.02 4	12.2 6	16.3 5	3.06 3	9.75 5	8.90 4
GC+occ [2]	9.9	1.19 4	2.01 9	6.24 4	1.64 12	2.19 11	6.75 7	11.2 13	17.4 14	19.8 12	5.36 10	12.4 12	13.0 11
Layered [5]	10.0	1.57 10	1.87 7	8.28 10	1.34 9	1.85 8	6.85 9	8.64 10	14.3 9	18.5 9	6.59 13	14.7 14	14.4 12
MultiCamGC [3]	10.5	1.27 5	1.99 8	6.48 5	2.79 17	3.13 15	3.60 5	12.0 14	17.6 15	22.0 14	4.89 9	11.8 10	12.1 9
TensorVoting [9]	11.7	3.79 16	4.79 16	8.86 12	1.23 8	1.88 9	11.5 13	9.76 11	17.0 13	24.0 16	4.38 7	11.4 9	12.2 10
YOUR METHOD	12.0	3.77 15	4.17 14	8.68 11	2.14 15	2.61 13	21.1 19	14.1 16	14.0 8	15.1 4	7.63 14	8.53 2	14.6 13
CostRelax [11]	12.4	4.76 18	6.08 18	20.3 19	1.41 11	2.48 12	18.5 16	8.18 9	15.9 11	23.8 15	3.91 5	10.2 7	11.8 8
RealTimeGPU [14]	12.2	2.05 13	4.22 14	10.6 14	1.92 14	2.98 13	20.3 17	7.23 6	14.4 10	17.6 7	6.41 12	13.7 12	16.5 14
ReliabilityDP [13]	13.4	1.36 7	3.39 11	7.25 9	2.35 15	3.48 16	12.2 15	9.82 12	16.9 12	19.5 10	12.9 19	19.9 19	19.7 16
TreeDP [8]	13.9	1.99 12	2.84 10	9.96 13	1.41 10	2.10 10	7.74 10	15.9 18	23.9 18	27.1 19	10.0 16	18.3 16	18.9 15
GC [1d]	14.4	1.94 11	4.12 13	9.39 12	1.79 13	3.44 15	8.75 11	16.5 19	25.0 20	24.9 17	7.70 14	18.2 15	15.3 13
DP [1b]	17.0	4.12 16	5.04 16	12.0 15	10.1 22	11.0 22	21.0 18	14.0 16	21.6 16	20.6 12	10.5 17	19.1 17	21.1 17
SSD+MF [1a]	18.3	5.23 19	7.07 18	24.1 19	3.74 18	5.16 18	11.9 14	16.5 20	24.8 19	32.9 20	10.6 18	19.8 18	26.3 19
SO [1c]	19.6	5.08 18	7.22 19	12.2 16	9.44 21	10.9 21	21.9 19	19.9 22	28.2 22	26.3 18	13.0 20	22.8 21	22.3 18
Infection [10]	20.5	7.95 21	9.54 20	28.9 21	4.41 19	5.53 19	31.7 20	17.7 21	25.1 21	44.4 22	14.3 21	21.3 20	38.0 21

6 Conclusions

This paper has presented a selective multiple-window algorithm to perform stereo matching. It is shown that by using a series of windows with increasing sizes, one can ensure that at least one window yields a reliable disparity estimate. For low textured images, this approach provides a *local–global* matching strategy with the large windows accounting for global matching and the small ones for local matching. A reliability test is introduced to select the most reliable estimate among multiple windows, in particular, increasing accuracy in low textured regions of an image. A three-stage recursive computation is also used to have a computationally efficient implementation. The recursion scheme makes the matching process independent of window size. The results obtained from a standard set of image pairs demonstrate that the developed algorithm provides higher-accuracy disparity maps as compared to two popular stereo matching local algorithms. The results from the

Middlebury College evaluation page also indicate that the introduced algorithm is the second-best performing correlation-based, multiwindow algorithm.

Acknowledgments

This work was partially supported by the Institute for Interactive Arts and Engineering at the University of Texas at Dallas.

References

1. D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Int. J. Comput. Vis.* **47**(1–3), 7–42 (2002).
2. V. Kolmogorov and R. Zabih, "Computing visual correspondence with occlusions using graph-cuts," *Proc. Int. Conf. Comp. Vis.*, pp. 508–515 (2001).
3. H. Ishikawa and D. Geiger, "Occlusions, discontinuities, and epipolar lines in stereo," *Proc. Eur. Conf. Comp. Vis.*, pp. 232–248 (1998).
4. Y. Ohta and T. Kanade, "Stereo by intra and inter-scan line search using dynamic programming," *IEEE Trans. Pattern Anal. Mach. Intell.* **7**(2), 139–154 (1985).
5. D. Geiger, B. Ladendorf, and A. Yuille, "Occlusions and binocular stereo," *Int. J. Comput. Vis.* **14**(3), 211–226 (1995).
6. O. Veksler, "Fast variable window for stereo correspondence using integral images," *Proc. Comp. Vis. Patt. Recog.* (2003).
7. Y. Boykov, O. Veksler, and R. Zabih, "A variable window approach to early vision," *IEEE Trans. Pattern Anal. Mach. Intell.* **20**(12), 1282–1294 (1998).
8. J. Sun, Y. Shum, and N. Zheng, "Stereo matching using belief propagation," *Proc. Eur. Conf. Comp. Vis.*, pp.510–524 (2002).
9. S. Roy and I. Cox, "A maximum-flow formulation of the *N*-camera stereo correspondence problem," *Proc. IEEE Int. Conf. on Computer Vision*, pp. 492–499 (1998).
10. A. Fusiello, V. Roberto, and E. Trucco, "Symmetric stereo with multiple windowing," *Int. J. Pattern Recognit. Artif. Intell.* **14**(8), 1053–1066 (2000).
11. J. Jeon, C. Kim, and Y. Sung Ho, "Sharp and dense disparity maps using multiple windows," *Lect. Notes Comp. Sci.*, Springer-Verlag, Berlin, 1057–1064 (2002).
12. L. Di. Stefano, M. Marchionni, and S. Mattoccia, "A fast area-based stereo matching algorithm," *Image Vis. Comput.* **22**(12), 983–1005 (2004).
13. K. Muhlmann, D. Maier, J. Hesser, and R. Manner, "Calculating dense disparity maps from color stereo images, an efficient implementation," *Int. J. Comput. Vis.* **47**(1–3), 79–88 (2002).
14. T. Kanade and M. Okutomi, "A stereo matching algorithm with an adaptive window: Theory and experiments," *IEEE Trans. Pattern Anal. Mach. Intell.* **16**(9), 920–932 (1994).
15. D. Marr and T. Poggio, "A computational theory of human stereo vision," *Proc. R. Soc. London, Ser. B* **204**, 301–328 (1979).
16. P. Fua, "Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities," *Proc. 12th Int. Joint Conf. on AI*, pp. 1292–1298 (1991).
17. M. Okutomi, Y. Katayama, and S. Oka, "A simple stereo algorithm to recover precise object boundaries and smooth surfaces," *Int. J. Comput. Vis.* **47**(1–3), 261–273 (2002).
18. M. Okutomi and T. Kanade, "A multiple baseline stereo," *IEEE Trans. Pattern Anal. Mach. Intell.* **15**(4), 353–363 (1993).
19. O. Faugeras, B. Hotz, M. Mathieu, T. Viville, Z. Zhang, P. Fua, E. Thron, L. Moll, and G. Berry, *Real-time correlation-based stereo: algorithm, implementation and applications*, INRIA Technical Report No. 2013 (1993).
20. T. Kanade, H. Kato, S. Kimura, A. Yoshida, and K. Oda, "A stereo machine for video-rate dense depth mapping and its new applications," *Proc. Comp. Vis. Patt. Recog.*, pp. 196–202 (1996).
21. S. Forstmann, J. Ohya, Y. Kanou, A. Schmitt, and S. Thuering, "Real-time stereo using dynamic programming," *Proc. Comp. Vis. Patt. Recog. Workshop on Real-time 3D Sensors and their use* (2004).
22. H. Hirschmüller, P. Innocent, and J. Garibaldi, "Real-time correlation-based stereo vision with reduced border errors," *Int. J. Comput. Vis.* **47**(1–3), 229–246 (2002).
23. L. Zitnick and T. Kanade, "A co-operative algorithm for stereo matching and occlusion detection," *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(7), 675–684 (2000).
24. M. Brown, D. Burschka, and G. Hager, "Advances in computational stereo," *IEEE Trans. Pattern Anal. Mach. Intell.* **25**(8), 993–1008 (2003).
25. S. Birchfield and C. Tomasi, "Depth discontinuities by pixel to pixel stereo," *Proc. IEEE Int. Conf. on Computer Vision*, pp. 1073–1080 (1998).
26. O. Faugeras, *Three-Dimensional Computer Vision: A Geometrical Viewpoint*, MIT Press, Cambridge, MA (1993).
27. A. Fusiello, E. Trucco, and A. Verri, "A compact algorithm for rectification of image pairs," *Mach. Vision Appl.* **12**(1), 16–22 (2000).
28. E. Trucco, V. Roberto, S. Tinonin, and M. Corbatto, "SSD disparity estimation for dynamic stereo," *Proc. Brit. Mach. Vis. Conf.*, pp. 342–352 (1996).
29. Middlebury College Database, http://www.middlebury.edu/stereo.
30. J. J. Little and W. E. Gillett, "Direct evidence for occlusions in stereo and motion," *Image Vis. Comput.* **8**(4), 328–340 (1990).
31. K. J. Yoon and I. S. Kweon, "Adaptive support-weight approach for correspondence search," *IEEE Trans. Pattern Anal. Mach. Intell.* **28**(4), 650–656 (2006).

Satyajit Anil Adhyapak received his MS degree from the University of Texas at Dallas in electrical engineering and his BS degree from the University of Mumbai, India, in electronics engineering. He is currently working at the Wireless Systems Development Unit of M/A-Com as a DSP Engineer. His research interests include DSP system design and its applications to communications and image processing.

Nasser Kehtarnavaz received his PhD degree in electrical and computer engineering from Rice University in 1987. He is a professor of electrical Engineering at the University of Texas at Dallas. His research interests include signal and image processing, pattern recognition, and real-time imaging. He has authored or co-authored 5 books and more than 130 journal and conference papers in these areas. He is currently serving as co-editor-in-chief of the *Journal of Real-Time Image Processing* and Chair of the Dallas chapter of the IEEE Signal Processing Society. Dr. Kehtarnavaz is a Fellow of SPIE, a senior member of IEEE, and a Registered Professional Engineer. More information on Dr. Kehtarnavaz's research activities are available at http://www.utdallas.edu/∼kehtar.

Mihai Nadin received his PhD degree in aesthetics from the University of Bucharest. He is currently director of the Institute for Research in Anticipatory Systems and the Ashbel Smith Professor in interactive arts, technology, and computer science at the University of Texas at Dallas. His research areas extend from the arts, aesthetics, philosophy, semiotics, to digital media, communications, mind, education, human–machine interaction and anticipation. He has published extensively in these areas.