Local Stereo Matching with Improved Matching Cost and Disparity Refinement

Jianbo Jiao, Ronggang Wang, Wenmin Wang, Shengfu Dong, Zhenyu Wang, and Wen Gao
Peking University Shenzhen Graduate School

A local stereo matching method that uses combined cost and a novel secondary disparity refinement mechanism ranked fifth out of the 153 methods submitted for the Middlebury benchmark.

One of the most active areas of computer vision research is *stereo matching,* which is the process of computing a disparity map given a pair of stereo images. Various stereo matching approaches exist,[1] with most algorithms categorized as either global or local. Typically, global methods achieve more accurate disparity maps with higher computational complexity, while local methods are more efficient. Recently, however, local methods based on adaptive support weight[2] have achieved results comparable to those of global methods that use graph cuts[3] or belief propagation.[4] Such local methods measure the similarity between a center pixel and its neighbor pixels by means of adaptive support weight. A high weight indicates that they are likely to be on the same object and thus have similar disparities. However, these methods involve high computational complexity, which is related to the window size used for aggregation.

In this article, we propose two strategies to further improve the performance of local stereo matching. (See "Related Work in Local Stereo Matching" sidebar for earlier work.) First, we propose a new cost measure by merging truncated absolute difference of color, gradients, and a modified color census transform to improve the initial matching performance. Second, after the traditional disparity refinement, we propose a secondary refinement approach called Remaining Artifacts Detection and Refinement (RADAR). RADAR can amend most of the remaining outliers following traditional postprocessing, which offers a remarkable improvement. For cost aggregation, we use the symmetric guided filter.[5,6] Our experimental results on the Middlebury benchmark[7] show our method's effectiveness; it is one of the best local stereo matching approaches, and it outperforms other cost-volume filtering-based methods on the Middlebury dataset (ranked fifth out of the 153 methods submitted). In addition, our method works well on both real-world sequences and some depth-based applications.

In this article, we extend our preliminary work[8] by using an adaptive judging-window to facilitate the RADAR strategy and present results from additional experiments to demonstrate our method's effectiveness. We also describe the RADAR mechanism in depth and propose a parallel implementation of our method on CPU. Finally, rather than confine it to a filtering-based method, we extend both the proposed cost measure and RADAR to a more common stereo matching framework.

Proposed Method

Figure 1 shows an overview of the process pipeline and the RADAR scheme. First, a cost volume is formulated using our proposed combined cost. Next, a symmetric guided filter is used for cost-volume filtering. Finally, we propose a RADAR-aided refinement scheme to further improve the disparity map's accuracy.

Combined Matching Cost and Initial Refinement

This phase of the process consists of four steps: modified color census transform, combined matching cost, symmetric guided filter aggregation, and initial postprocessing.

Modified Color Census Transform. Motivated by color census transform,[9] we propose a *modified color census transform* (MCCT). Because RGB color space is sensitive to radiometric changes, the image is first converted to

1070-986X/14/$31.00 © 2014 IEEE

Published by the IEEE Computer Society

Related Work in Local Stereo Matching

Several existing approaches aim to improve local stereo matching performance. Christoph Rhemann and his colleagues proposed a new approach for cost aggregation by smoothing cost volume, and its complexity is independent of the matching window size.[1] Several other cost-volume filtering-based methods have also been developed recently and achieved good performance. In one project,[2] a hardware-efficient bilateral filter was proposed for fast aggregation. In another,[3] the domain transform was imported so that cost aggregation was performed using 1D filters. Qingxiong Yang introduced a recursive bilateral filter for aggregation.[4] Leonardo De-Maeztu and his colleagues presented an $O(1)$ method[5] based on a symmetric filter.

For matching cost computation, researchers often use the absolute difference (AD) to calculate the difference between corresponding pixels' intensity to measure the likelihood. Cost measures such as sum of AD (SAD) and sum of squared difference (SSD) are also widely used. Gradient-based measures and nonparametric transforms such as rank and census, however, provide a better description of image structure.[6] Rhemann and his colleagues combined the AD with gradient and obtained impressive results,[1] while Xing Mei and his colleagues presented a new cost measure by combining AD and census to reduce the errors caused by individual measures.[6]

Although stereo matching performance has been improved, the final results often still contain some obvious artifacts. To date, researchers have spent considerable effort on cost aggregation improvement but have paid far less attention to disparity refinement and cost measurement.

References

1. C. Rhemann et al., "Fast Cost-Volume Filtering for Visual Correspondence and Beyond," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2011, pp. 3017–3024.
2. Q. Yang, "Hardware-Efficient Bilateral Filtering for Stereo Matching," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 36, no. 5, 2013, pp. 1–8.
3. C. Pham, and J.W. Jeon, "Domain Transformation-Based Efficient Cost Aggregation for Local Stereo Matching," *IEEE Trans. Circuits Systems Video Technology*, vol. 23, no. 7, 2013, pp. 1119–1130.
4. Q. Yang, "Recursive Bilateral Filtering," *Proc. Euro. Conf. Computer Vision*, 2012, pp. 399–413.
5. L. De-Maeztu et al., "Linear Stereo Matching," *Proc. IEEE Int'l Conf. Computer Vision*, 2011, pp. 1708–1715.
6. X. Mei et al., "On Building an Accurate Stereo Matching System on Graphics Hardware," *Proc. IEEE Int'l Conf. Computer Vision Workshops*, 2011, pp. 467–474.

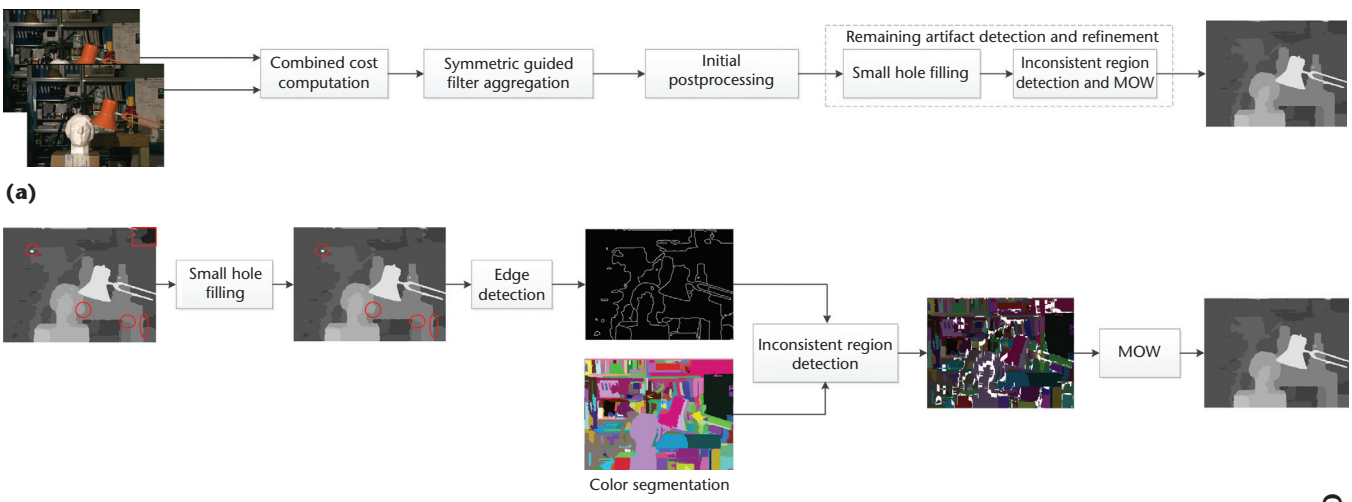


Figure 1. Overview of the proposed method. (a) The whole pipeline. (b) Workflow of RADAR.

Gaussian color model space.[10] Next, the difference between two pixels p and q is measured by the Euclidean distance, and the mean value of all these distances in the window centered at p is denoted by $D_m(p)$. The MCCT is formulated as follows:

$$\mathrm{MCCT}(p) = \bigotimes_{q \in N(p)} \xi(D_m(p), D_G(p,d))$$

$$\xi(a,b) = \begin{cases} 1, & b < a \\ 0, & \text{otherwise} \end{cases},$$

where operator $\otimes$ denotes a bit-wise catenation and $N(p)$ represents the neighbor pixel set of p.

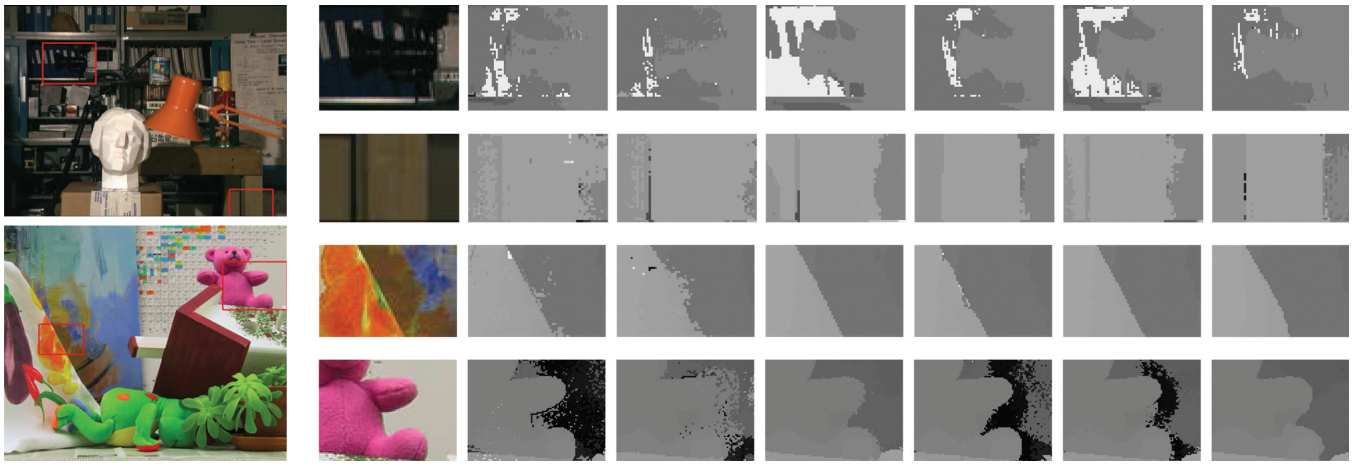


Figure 2. Cost measure comparison. (a) The source images; the red boxes show the target areas. The detailed rows of images show, from top to bottom, the repetitive region, the dark region, the edge, and the textureless region. The columns show (b) a close-up of the target areas; the results of (c) absolute difference (AD), (d) gradient, (e) census, (f) AD and gradient, and (g) AD and census; and (h) the results of the proposed combined cost.

The Hamming distance is used to calculate the difference between the two bit-strings generated by MCCT:

$$h(p,d) = \text{Hamming}(\text{MCCT}_L(p), \text{MCCT}_R(p-d)),$$

where d denotes the disparity of two corresponding pixels in the left and right images. Finally, a robust exponential function is used to normalize the cost:

$$C_{\text{MCCT}}(p,d) = 1 - \exp\left(\frac{h(p,d)}{\lambda_{\text{MCCT}}}\right),$$

where λ_{MCCT} is a normalizing parameter of 55. Our new MCCT has a better representation of the image structure than the traditional color census.[9] Our preliminary work[8] compares MCCT and traditional color census.

Combined Matching Cost. In addition to MCCT, we add two other cost components of truncated absolute differences of color and gradients, which are calculated respectively as follows:

$$C_{ADc}(p,d) = \min\left(\frac{1}{3}\sum_{i-R,G,B}\left\|I_i^L(p) - I_i^R(p-d)\right\|, \lambda_{ADc}\right)$$

$$C_{GDx}(p,d) = \min(\|\nabla_x I_L(p) - \nabla_x I_R(p-d)\|, \lambda_{GD}),$$

where λ_{ADc} and λ_{GD} are the truncated values[5] and ∇_x is the derivative in x direction. The gradient in y direction is also used, denoted as C_{GDy}. The final combined matching cost is formulated by merging the four cost components mentioned earlier:

$$C(x,y,d) = \alpha \cdot C_{\text{MCCT}} + \beta \cdot C_{ADc} + \gamma \cdot C_{GDy} + (1-\alpha-\beta-\gamma) \cdot C_{GDx},$$

where α, β, and γ are weights for different cost components, adjusting the four components' contributions to the total cost. Figure 2 compares the proposed combined cost and individual costs as well as other combined costs. All of the disparity maps are initial stereo matching results without any post-processing; the cost aggregation strategy is the same for all tests.

Symmetric Guided Filter Aggregation. The combined cost for each pixel at each disparity level is stored in a cost volume. To preserve both edges in reference and target images, we use the symmetric guided filter for the cost aggregation.[5,6] Once we aggregate the cost volume using the symmetric guided filter, we employ the "winner-takes-all" strategy for disparity selection—that is, we select the disparity label with the lowest cost. We then generate the initial disparity map.

Initial Postprocessing. The initial disparity map contains many outliers. To find the inconsistent pixels in the left and right images, we use the left-right consistency check (LRC). We

label a pixel p as an outlier if it violates the following constraint:

$$|d_L(p) - d_R(p - d_L(p))| < 1$$

where d_L and d_R are the disparities of the corresponding pixels in left and right images, respectively.

Once the outliers are detected, we use a cross-region based voting technique[11] to correct them. The voting operation runs iteratively to increase its robustness. Figure 3a shows the cross region of a pixel p. (Details about the voting method are available elsewhere.[11])

Following the cross-region voting, we use the nearest reliable pixel in the scan line to update the remaining outliers labeled by LRC. To remove the streak-like artifacts, we use a weighted median filter with bilateral filter weights.[5]

Remaining Artifacts Detection and Refinement

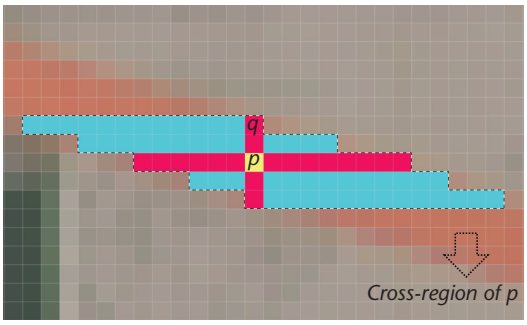
Some error regions remain after the initial post-processing. If these artifacts exist in both left and right images, they cannot be detected by LRC alone. Thus, we propose the secondary RADAR refinement scheme; Figure 1b shows its pipeline.

Small Hole Filling. According to our observation, remaining artifacts mainly consist of small holes and outliers around the object boundary (see Figure 4). *Small holes* are dark regions with disparities much smaller than their neighbors. We can detect them by comparing their disparities with those of their neighbors. After detecting a hole pixel, we use the most appropriate disparity in its neighborhood (both horizontal and vertical) to update it. Typically, the hole pixel p is updated by the pixel with a smaller disparity (background pixel), but if the pixel on the smaller disparity side of p is also invalid (hole pixel), it should be updated by the pixel on the other side:

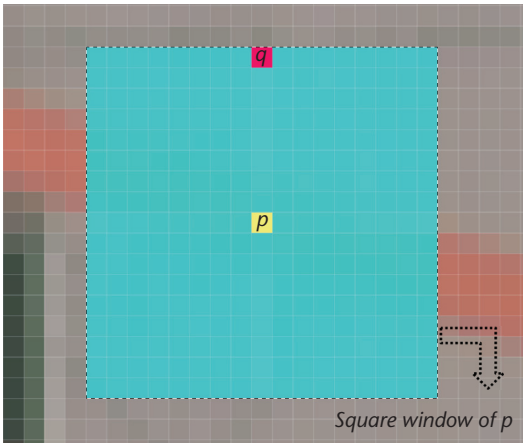
$$d_p^* = \begin{cases} \max\left\{d_p', d_p''\right\} & d_p' \cdot d_p'' \le d_{\text{thres}}^2 \\ \min\left\{d_p', d_p''\right\} & d_p' \cdot d_p'' > d_{\text{thres}}^2 \end{cases},$$

$$d_{\text{thres}} = \rho \cdot d_{\max}$$

where d_p' and d_p'' are the nearest (taking one direction as an example) pixels' disparities larger than d_{thres}, and $d_{\max}$ is the maximum disparity, while ρ is an empirical penalty of 1/7. The updated disparity is denoted as d_p^*.



(a)



(b)

Figure 3. Support region of pixel* p.** ***(a) Cross-region. (b) Traditional square window.

Inconsistent Region Detection. As Figure 4 shows, the other type of artifacts is outliers around the object boundary. We call these artifacts *inconsistent regions,* which consist of convex regions (also called "fattening" regions[1]) and concave regions. The inconsistent region is detected by checking whether the edges of disparity map coincide with the boundaries of objects in the scene. We use the Canny edge detector[12] to extract the disparity edges. We use color segmentation based on the mean-shift approach[13] to detect the objects' boundaries. First, however, a contrast enhancement operation (a histogram equalization on the luminance part of the color image) is performed, and then the image is converted to CIELab space.

This preprocessing improves segmentation accuracy, especially on dark regions. If a disparity map edge does not exactly coincide with the scene's object boundary, we label it a *mismatched edge.* Convex regions lie on the foreground side of the mismatched edge, while concave regions lie on the background side.

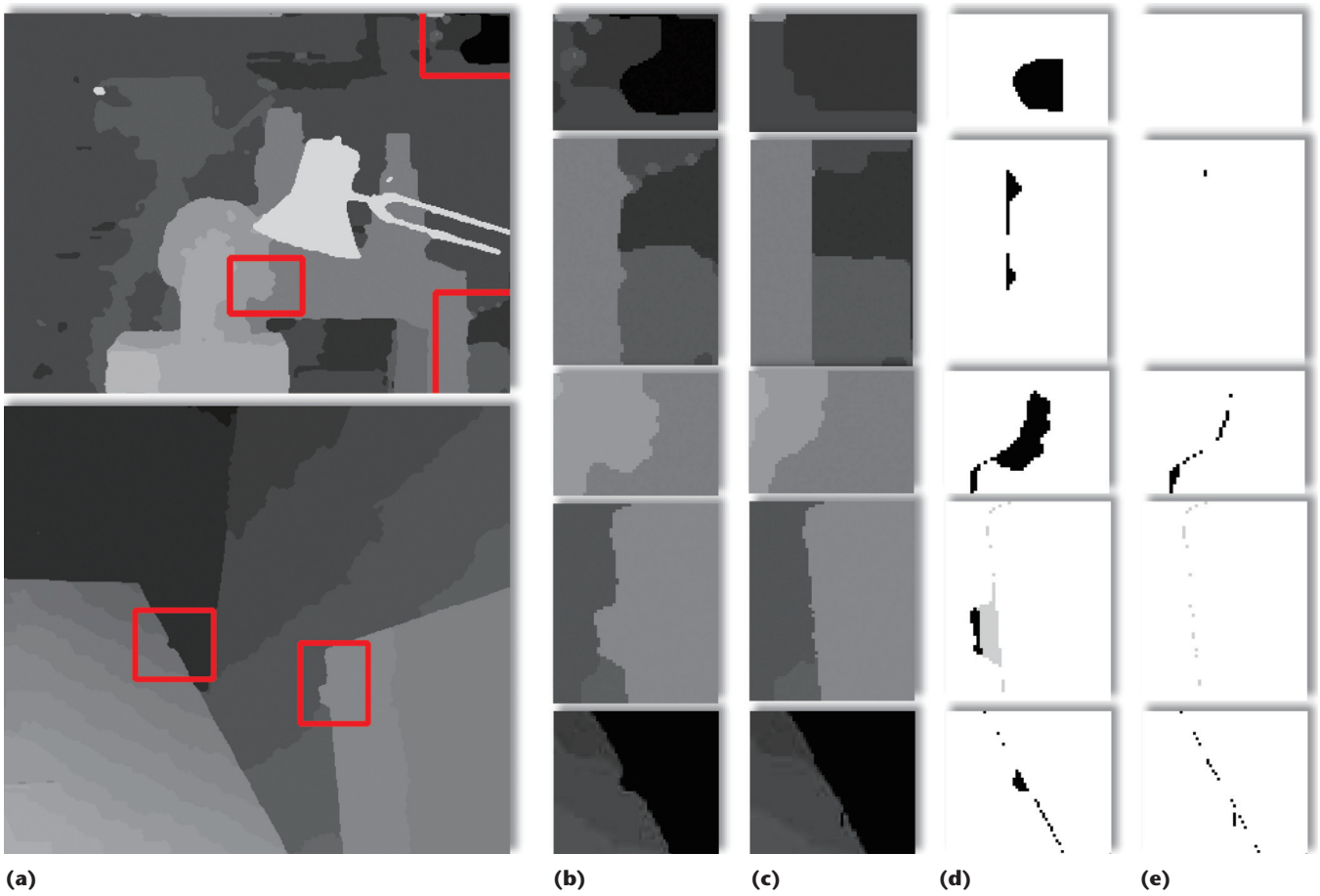


Figure 4. Remaining artifacts before and after RADAR. (a) Disparity maps before RADAR. In columns b–e, the first row represents the small hole, or dark regions with disparities much smaller than their neighbors; the second through fourth rows represent convex regions; and the last row represents the concave region. (b) A close-up of the target regions. (c) Results after RADAR. (d) Error maps of the target regions in (b). (e) Error maps of the regions after using RADAR.

Figure 5 shows how we determine which side is the inconsistent region.

In Figure 5a, the inconsistent region is labeled using a judging-window (the black grid). Region A (circled by the blue dotted line) represents the foreground object, while the rest (region B) is background. The red line is the detected mismatched edge. At each (red) point on the mismatched edge, a judging-window is formed and divided into two areas: the foreground area (larger disparity d_1) and the background area (smaller disparity d_2). Thus, the inconsistent region is labeled on the smaller side of the mismatched edge, such as the d_1 side on the left image and the d_2 side on the right. The inconsistent (convex or concave) region in Figure 5a is the region encircled by the blue and red lines. However, when the area of d_1 is equal to the area of d_2 (see the left image in Figure 5b), the judgment of the inconsistent side would get into a dilemma. Once it occurs, the judging-window size is increased until the size of the two areas differs (see the right image in Figure 5b).

Modified OccWeight. To correct the inconsistent regions detected above, we modify the OccWeight.[14] The traditional OccWeight (MOW) method corrects a pixel's disparity by choosing the most likely one in a fixed square window around it. However, a squared window (Figure 3b) is not robust. Hence, we replace the square window with a cross window (Figure 3a). The cross window is consistent with image structure and provides a much more accurate support region. We also adopt disparity inheritance.[14] In a cross window of pixel p, the weight

of its neighboring pixel (q) is calculated as follows:

$$w(p,q) = \begin{cases} \exp\left(-\frac{\Delta c_{pq}}{\phi_c} - \frac{\Delta s_{pq}}{\phi_s}\right), & \text{if } q \notin R_I \\ 0 & \text{otherwise} \end{cases},$$

where $\Delta_{c_{pq}}$ and $\Delta_{s_{pq}}$ are the color distance and spatial distance between p and q, and ϕ_c and ϕ_s are parameters used to normalize the color and spatial distances, respectively.[14] R_I is the set of inconsistent regions. The updated disparity is calculated as

$$d^*(p) = \arg\max_{d \in D} \left(\sum_{q \in AW_p} w(p,q) \times m(q,d) \right)$$

$$m(p,q) = \begin{cases} 1, & \text{if } d(q) = d \\ 0, & \text{otherwise} \end{cases},$$

where D is the set of disparity candidates and AW_p is the set of pixels in the cross window of p. By employing the MOW, the outliers at inconsistent regions are corrected (the last image in Figure 1b). Finally, we smooth the disparity map with a median filter to remove the remaining noises. Figure 6 shows the RADAR algorithm's pseudo code.

To evaluate the effectiveness of our proposed RADAR-aided refinement pipeline, we tested it on four Middlebury dataset images (Tsukuba, Venus, Teddy, and Cones)[7] and used the evaluation measures of the nonoccluded regions (Nonocc), all regions (All), and regions near discontinuities (Disc). For each measure, we calculated the average value of the four images in the dataset.

Figure 7a shows the effectiveness of each step in the proposed refinement pipeline. CRV and WMF represent the cross-region voting and weighted median filter, respectively. As the figure shows, the error decreases after each step of our pipeline. We also compared the RADAR-aided disparity refinement with the fattening region refinement[15] (MDC) and OccWeight[14] (see Figure 7b). For MDC, we used its fattening detection method,[15] whereas for OccWeight, we used our region-detection method. Furthermore, we evaluated the RADAR-only (RADAR-o) item without the initial post-processing. All these methods are based on the same initial disparity map—that is, on our proposed combined cost and aggregation. As Figure 7b shows, our refinement pipeline performs the best among all these methods.

To validate RADAR's universality, we compared it under the platform of other local methods (see Figures 7c and 7d). We chose BlockMatch and DTAggr-P[16] for the comparison, replacing their disparity refinement steps with the methods to be compared. Our RADAR scheme still offers the best performance under these local methods.

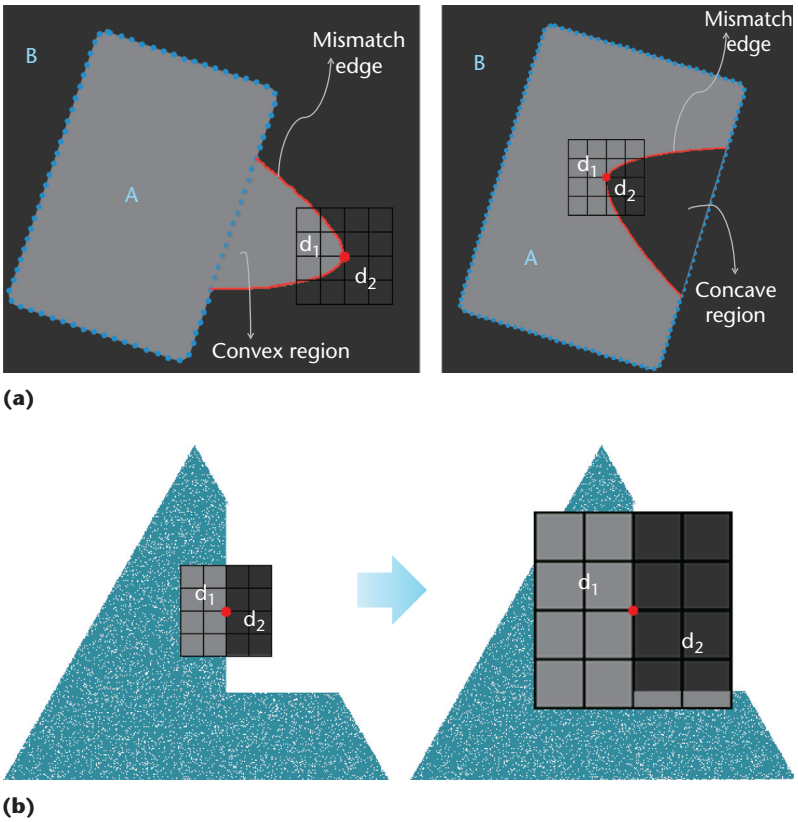


Figure 5. Inconsistent region detection. (a) Decisions on the convex and concave regions. (b) Adaptive judging windows.

Experimental Results

We performed an experimental evaluation of our proposed method using two test sets: the Middlebury dataset[7] and some typical real-world sequences. We also show applications in 3D reconstruction and virtual view synthesis. We chose the experiment's parameters empirically and kept them constant as $\{\alpha, \beta, \gamma\} = \{0.011, 0.15, 0.1\}$. We obtained these parameters based on 35 images with ground truth disparity maps on Middlebury datasets according to an optimization process aiming at obtaining the minimum average error percentage.

Middlebury Dataset

Figure 8 shows our experimental results on four test images from the Middlebury online benchmark: Tsukuba, Venus, Teddy, and Cones. Our

Algorithm 1
Input: Disparity map D (disparity of p is d_p) and color image I
Output: Refined disparity map D^*

1: ----------"small hole" filling----------
2: $\boldsymbol{d}_{\text{thres}}$
3: d'_p, d''_p $\in$ {disparity of the nearest pixel to p}
4: **if** $d'_p.d''_p > d^2_{\text{thres}}$ **then**
5: $d^*_p \leftarrow \min\{d'_p, d''_p\}$
6: **else**
7: $d^*_p \leftarrow \max\{d'_p, d''_p\}$
8: **else if**
9: ----------inconsistent region detection (see Figure 5)----------
10: $E^* \leftarrow$ Canny edge detection on disparity map
11: $I^* \leftarrow$ contract enhancement on color image I
12: $S^* \leftarrow$ segmentation on I^* in CIELab space
13: **for** each edge e in E^*, each edge s in S^* **do**
14: **if** e through s **then**
15: mismatch edge $\leftarrow e$
16: **end if**
17: **end for**
18: **for** each point on mismatch edge do
19: **if** area d_1 < area d_2 **then**
20: $R_I \leftarrow$ area d_1
21: **else if** area d_1 > area d_2 **then**
22: $R_I \leftarrow$ area d_2
23: **else**
24: **while** area d_1 == area d_2 **do**
25: increase the size of judging window
26: **end while**
27: go to 19
28: **end if**
29: **end for**
30: ----------MOW----------
31: **for** each error pixel p in R_I **do**
32: $AW_p \leftarrow$ adaptive window of p
33: **for** $q \in AW_p$ **do**
34: $w(p, q) \leftarrow$ weight of p, q
35: **end for**
36: $d^*(p) = \arg\max_{d \in D} \left(\sum_{q \in AW_p} w(p,q) \times m(q,d) \right)$
37: **end for**

Figure 6. Algorithm 1. Remaining Artifacts Detection and Refinement (RADAR).

proposed method yielded competitive performance compared with the state-of-the-art methods and (at the time of this writing) ranked fifth out of 153 submitted methods. To the best of our knowledge, our method is the top cost-volume filtering-based local method.

We also compared our method with other filtering-based local methods and with ADCensus,[11] the top local method on Middlebury. Table 1 shows the results and the error percentages in different regions for the four images. We set the error threshold to the default 1.0. In addition, we chose a subpixel threshold of 0.75, and the rank is shown in Table 1's last column.

As the table shows, when the error threshold is 1.0, our method is the best cost-volume filtering-based approach and the second best local method (second only to ADCensus). However, when errors are evaluated at a subpixel level (0.75), our method performs the best. Subpixel evaluation means that the disparity can be a floating number, instead of being limited to an integer, which is useful in practical applications. However, it's worth noting that our method disregards subpixels, estimating all disparities at the integer level. On the subpixel level, our method's performance shows only a slight decline (from fifth to 13th in the rankings), which proves the method's robustness. Figure 9 shows experimental results on other images in the Middlebury dataset.

Real-World Dataset

For this experiment, we choose four real-world sequences as the test set:

- the BookArrival sequence from Fraunhofer Heinrich Hertz Institute (HHI) 3D video database;
- the Balloons sequence from Free-Viewpoint Television, MPEG-FTV;
- Cafe from the Gwangju Institute of Science and Technology (GIST); and
- the Newspaper sequences obtained from GIST.

For each test sequence, we randomly extracted a frame with its corresponding view as a test image pair. We used some of the competitive filtering-based methods mentioned earlier for comparison—specifically, RecursiveBF,[17] CostFilter,[5] DTAggr-P,[16] and Hardware-Efficient Bilateral Filtering (HEBF).[18] We used the original author's recommendations for the parameter settings of these methods. Figure 10 shows the visual results.

As the visual results show, our method has the best edge-preserving performance; see, for example, the lion in the BookArrival sequence

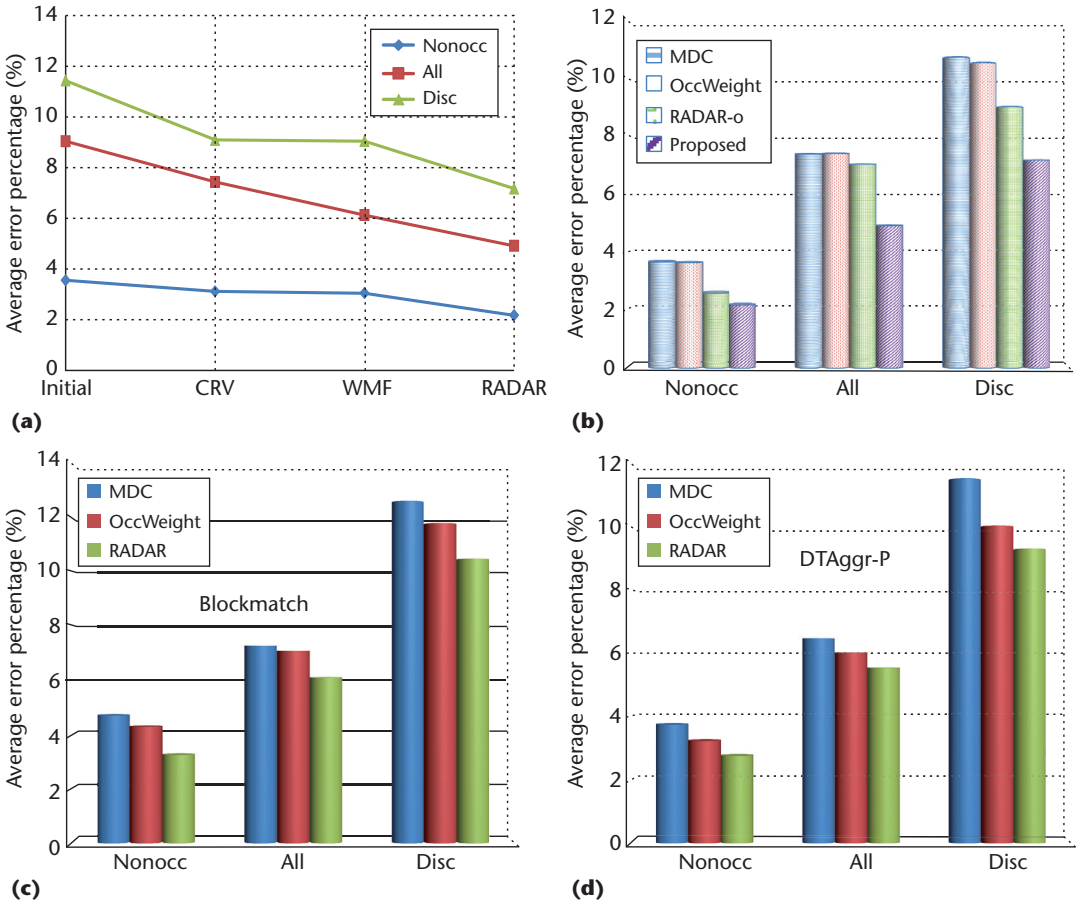


Figure 7. Testing the RADAR-aided refinement pipeline. (a) Improvements of each step in the refinement pipeline. (b) Comparison with other approaches. (c) Comparisons on BlockMatch. (d) Comparisons on DTAggr-P.

and the Balloons sequence's objects. Our disparity results also perform well on image borders—as in the coat on the left border of the BookArrival and Newspaper sequences—which is important in practical applications, such as virtual view synthesis and 3D reconstruction. These real-world sequence results again prove our proposed method's effectiveness.

3D Reconstruction and View Synthesis

To verify our proposed method's effectiveness, we generated 3D reconstruction and virtual view synthesis results based on the disparity map that our method calculated. Figure 11a shows the 3D views of Tsukuba and Teddy reconstructed by the disparity maps generated by our method, CostFilter, and ADCensus. For the red rectangle regions, our method performs better than the other two methods.

Another application is depth-based virtual view synthesis. Here, we used MPEG's popular view synthesis reference software (VSRS) reference software to synthesize a virtual view at the middle of the two reference views. We chose the Balloons and BookArrival sequences for the test. Figure 11b shows results based on the disparity maps generated by Block Matching, CostFilter, DTAggr-P, and our proposed method. As the figure shows, many artifacts in the virtual view are synthesized by BlockMatch, CostFilter, and DTAggr-P, while there is no artifact in the synthesized view by our method, which validates the effectiveness of our proposed stereo matching method.

Computational Complexity

Our proposed stereo matching method's computational complexity is determined by four steps:

1. combined cost computation,
2. guided filter aggregation,
3. initial refinement, and
4. RADAR.

Let N be the number of image pixels and D be the number of disparity levels. The

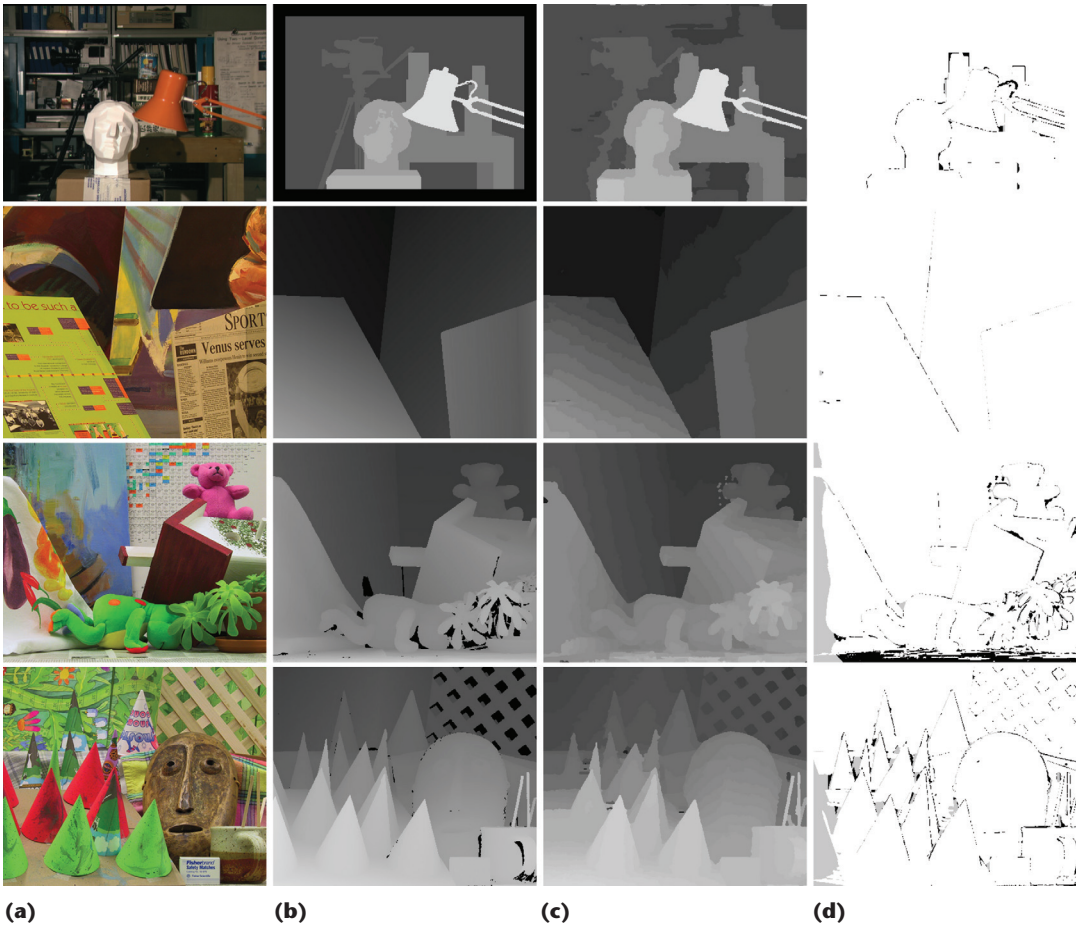


Figure 8. Experimental results on main Middlebury dataset images. The Middlebury images (from top to bottom) Tsukuba, Venus, Teddy, and Cones: (a) color images, (b) ground truth, (c) the results from our method, and (d) error maps set at a 1.0 threshold.

Table 1. Quantitative evaluation of the proposed method compared with other local methods on Middlebury.

		Tsukuba			Venus			Teddy			Cones			
Algorithm	Rank	nonocc	all	disc	nonocc	all	disc	nonocc	all	disc	nonocc	all	disc	Rank
ADCensus	2	1.07	1.48	5.73	0.09	0.25	1.15	4.10	6.22	10.9	2.42	7.25	6.95	28
Proposed	5	1.15	1.42	6.23	0.15	0.27	1.89	5.39	10.6	14.7	2.01	7.37	5.88	13
HEBF	31	1.10	1.38	5.74	0.22	0.33	2.41	6.54	11.8	15.2	2.78	9.28	8.10	24
DTAggr-P	39	1.75	2.10	7.09	0.24	0.45	2.59	5.70	11.5	13.9	2.49	7.82	7.30	33
CostFilter	40	1.51	1.85	7.61	0.20	0.39	2.42	6.16	11.8	16.0	2.71	8.24	7.66	19
P-LinearS	53	1.10	1.67	5.92	0.53	0.89	5.71	6.69	12.0	15.9	2.60	8.44	6.71	53
RecursiveBF	68	1.85	2.51	7.45	0.35	0.88	3.01	6.28	12.1	14.3	2.80	8.91	7.79	36

complexity of the combined cost computation is $O(ND)$, the cost aggregation is $O(1)$,[6] given the integral image and the complexity of computing integral image is $O(N)$. The complexity of initial disparity refinement is also $O(N)$. In RADAR, small hole filling and the MOW can be done in $O(N)$ and $O(\varepsilon)$, where ε is the number of error pixels. For the Canny edge detector, the complexity is $O(N \log N)$. The color segmentation in RADAR can be implemented in $O(N \log N)$ or even $O(N)$ with the help of a *k*-dimensional (k-d) tree or integral image. Therefore, our proposed method's overall complexity is $O(N + ND + N \log N + \varepsilon)$. Because D and ε are small constants for practical applications, the complexity can be approximately equal to $O(N \log N)$. In addition, most parts of the method can be implemented in parallel. Thus, our method is applicable for high-resolution images or video applications.

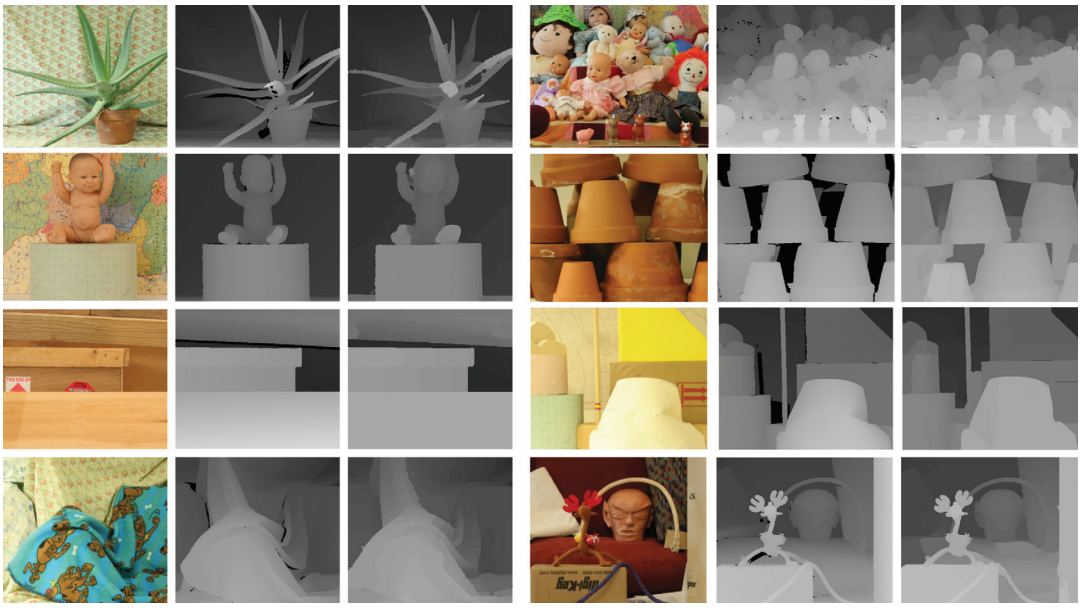


Figure 9. Experimental results on additional Middlebury dataset images. Results are shown for two sets of images for (from left to right) color images, ground truth, and the proposed method.

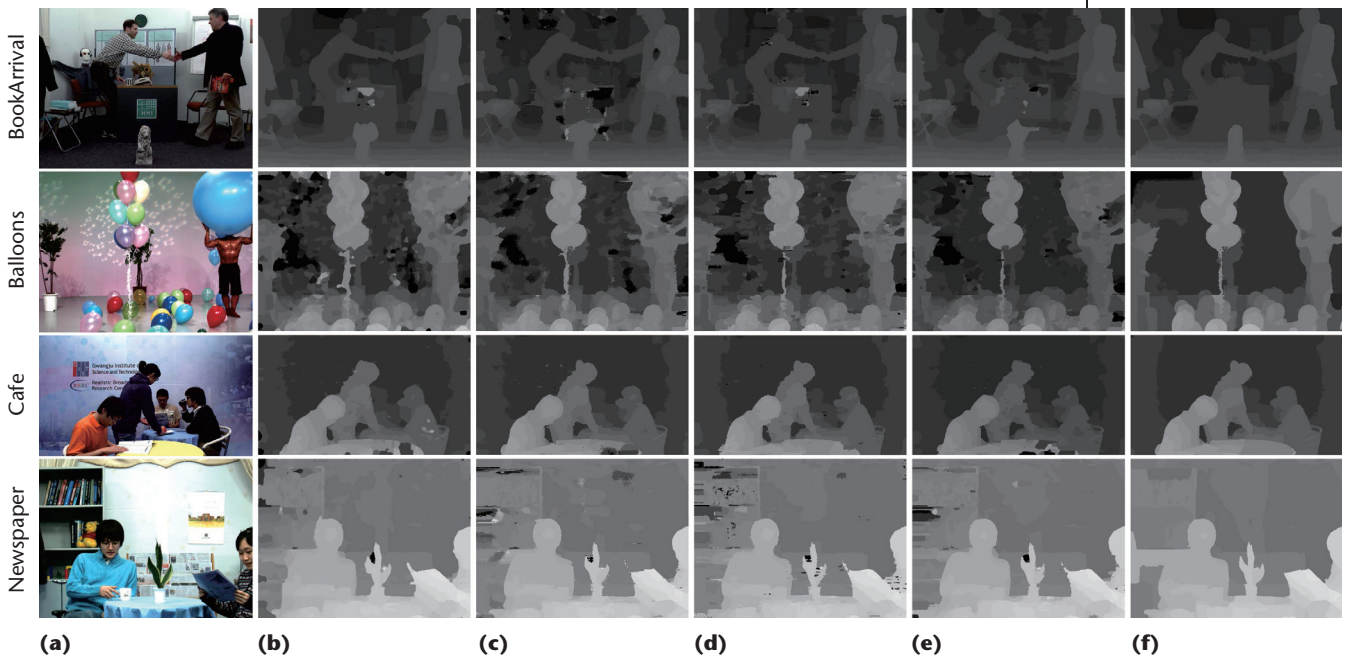
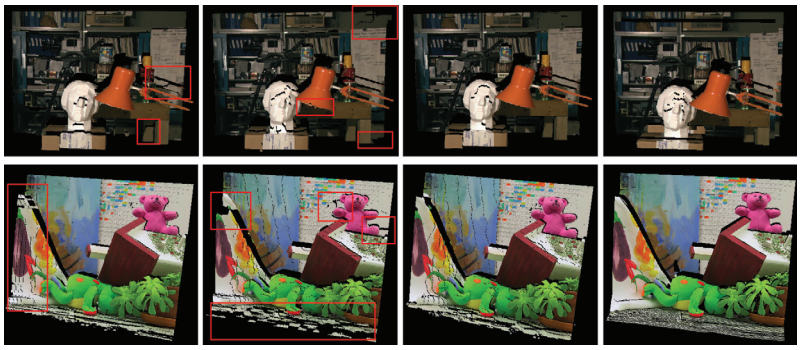
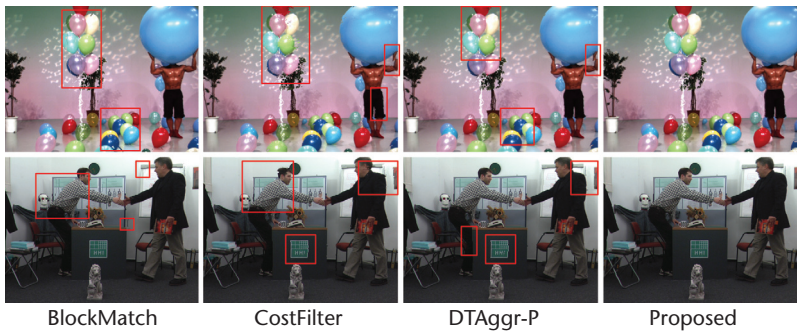


Figure 10. Experimental results of our method compared with competitive algorithms on real-world sequences. (a) Left frames. Results of the (b) RecursiveBF, (c) CostFilter, (d) DTAggr-P, and (e) HEBF methods. (f) The results of our proposed method.

For the Middlebury dataset, the dimensions of images Tsukuba, Venus, Teddy, and Cones are 384 × 288, 434 × 383, 450 × 375, and 450 × 375 with the disparity level of 15, 19, 59, and 59, respectively. We applied our algorithm to each pair of stereo images to calculate the disparity results. In each case, the computational time was recorded. We implemented our experiment on a PC equipped with a 3.40 GHz Intel core i7 CPU and 4 Gbytes of memory. Our method's main time-consuming steps are aggregation and disparity refinement, which account for 60.61 and 24.49 percent of the total time, respectively. However, both of those steps—as well as cost computation—can be performed in parallel. With the help of OpenMP, we parallelized parts of our algorithm on the CPU, and the acceleration ratio was about 3.2 times. The



(a)



(b)

Figure 11. 3D reconstruction and view synthesis. (a) 3D reconstruction of Tsukuba and Teddy. (b) Virtual view synthesis on the Balloons and BookArrival sequences.

computational time for Tsukuba, Venus, Teddy, and Cones were 3.11, 5.70, 12.34, and 12.48 seconds, respectively. Because the algorithm has not been fully optimized and is implemented on a CPU instead of the GPU, we expect it to accelerate greatly in our future work.

Conclusion

We proposed a combined matching cost function and a secondary disparity refinement scheme to improve the local stereo matching performance. Experimental results demonstrated the effectiveness of our method. In the near future, we will make it more suitable for real-time applications by utilizing GPUs. In addition, we will endeavor to improve the performance of the subpixel level accuracy. Our experimental results also show that the disparity map generated by the proposed method can be used in some 3D applications. The proposed stereo matching method can be integrated into a virtual view synthesis system and used for the free-view 3D video generation or video coding scenarios. **MM**

Acknowledgment

This work was partially supported a National Science Foundation of China grant (61370115) and a Shenzhen Peacock Plan grant.

References

1. D. Scharstein, and R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms," *Int'l J. Computer Visualization*, vol. 47, nos. 1–3, 2002, pp. 7–42.
2. K.J. Yoon and I.S. Kweon, "Adaptive Support-Weight Approach for Correspondence Search," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, 2006, pp.650–656.
3. V. Kolmogorov and R. Zabih, "Computing Visual Correspondence with Occlusions Using Graph Cuts," *Proc. IEEE Int'l Conf. Computer Vision*, 2001, pp. 508–515.
4. J. Sun, N.-N. Zheng, and H.-Y. Shum, "Stereo Matching Using Belief Propagation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 7, 2003, pp. 787–800.
5. C. Rhemann et al., "Fast Cost-Volume Filtering for Visual Correspondence and Beyond," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2011, pp. 3017–3024.
6. L. De-Maeztu et al., "Linear Stereo Matching," *Proc. IEEE Int'l Conf. Computer Vision*, 2011, pp. 1708–1715.
7. D. Scharstein and R. Szeliski, "Middlebury Stereo Evaluation Online," version 3, Middlebury College Computer Vision; http://vision.middlebury.edu/stereo/eval.
8. J. Jiao et al., "Cost-Volume Filtering-Based Stereo Matching with Improved Matching Cost and Secondary Refinement," *Proc. IEEE Int'l Conf. Multimedia and Expo*, 2014; doi:10.1109/ICME.2014.6890201.
9. S.C. Pei and Y.Y. Wang, "Color Invariant Census Transform for Stereo Matching Algorithm," *Proc. IEEE Int'l Symp. Consumer Electronics*, 2013, pp. 209–210.
10. J. Geusebroek, R. Boomgaard, and A.W.M. Smeulders, "Color Invariance," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 12, 2001, pp. 1338–1350.
11. X. Mei et al., "On Building an Accurate Stereo Matching System on Graphics Hardware," *Proc. IEEE Int'l Conf. Computer Vision Workshops*, 2011, pp. 467–474.
12. J. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, 1986, pp. 679–698.

13. D. Comaniciu and P. Meer, "Mean Shift: A Robust Approach Toward Feature Space Analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, 2002, pp. 603–619.
14. W. Wang, and C. Zhang, "Local Disparity Refinement with Disparity Inheritance," *Proc. Symp. Photonics and Optoelectronics*, 2012, pp. 1–4.
15. Y.-C. Wang, C.-P. Tung, and P.-C. Chung, "Efficient Disparity Estimation Using Hierarchical Bilateral Disparity Structure Based Graph Cut Algorithm with a Foreground Boundary Refinement Mechanism," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 23, no. 5, 2013, pp. 784–801.
16. C. Pham and J.W. Jeon, "Domain Transformation-Based Efficient Cost Aggregation for Local Stereo Matching," *IEEE Trans. Circuits Systems Video Technology*, vol. 23, no. 7, 2013, pp. 1119–1130.
17. Q. Yang, "Recursive Bilateral Filtering," *Proc. Euro. Conf. Computer Vision*, 2012, pp. 399–413.
18. Q. Yang, "Hardware-Efficient Bilateral Filtering for Stereo Matching," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 36, no. 5, 2013, pp. 1–8.

Jianbo Jiao is a graduate student at Peking University Shenzhen Graduate School, China. His research interests include computer vision and stereo vision. Jiao has a BS in computer science and technology from the Southwest Jiaotong University, China. Contact him at jianbojiao@sz.pku.edu.cn.

Ronggang Wang is an associate professor at Peking University Shenzhen Graduate School. His research interests include video coding and processing, and he has contributed to several video coding standards, including ISO/IEC MPEG Internet Video Coding, IEEE 1857, and China AVS. Wang has a PhD in computer engineering from Chinese Academic of Sciences. Contact him at rgwang@pkusz.edu.cn.

Wenmin Wang is a professor in the School of Electronic and Computer Engineering at Peking University Shenzhen Graduate School. His research interests include multimedia analysis and retrieval; he is also interested in artificial intelligence and machine learning, and Web and 3D technology. Wang has a PhD in computer architecture from Harbin Institute of Technology, China. Contact him at wangwenmin@pkusz.edu.cn.

Shengfu Dong is an engineer at Peking University Shenzhen Graduate School. His research interests include video coding, focusing on optimization algorithms, and high-performance parallel computing. Dong has an MS in electronics and communication engineering from Harbin Institute of Technology, China. Contact him at dongsf@pkusz.edu.cn.

Zhenyu Wang is a PhD candidate in computer science at Peking University and an engineer in the Peking University Shenzhen Graduate School. His research interests include video coding and multimedia systems. Wang has an MS in computer science from Peking University. Contact him at wangzhenyu@pkusz.edu.cn.

Wen Gao is a professor of computer science at Peking University. His research interests include image processing, video coding and communication, pattern recognition, and multimedia information retrieval. Gao has a PhD in electronics engineering from the University of Tokyo. He is a fellow of IEEE and ACM and is a member of the Chinese Academy of Engineering. Contact him at wgao@pku.edu.cn.