Real-Time and Accurate Stereo: A Scalable Approach with Bitwise Fast Voting on CUDA

Ke Zhang, *Student Member, IEEE,* Jiangbo Lu, *Member, IEEE,* Qiong Yang, *Member, IEEE,* Gauthier Lafruit, *Member, IEEE,* Rudy Lauwereins, *Senior Member, IEEE,* and Luc Van Gool, *Member, IEEE*

***Abstract*—This paper proposes a real-time design for accurate stereo matching on compute unified device architecture (CUDA). We present a leading local algorithm and then accelerate it by parallel computing. High matching accuracy is achieved by cost aggregation over shape-adaptive support regions and disparity refinement using reliable initial estimates. A novel sample-and-restore scheme is proposed to make the algorithm scalable, capable of attaining several times speedup at the expense of minor accuracy degradation. The refinement and the restoration are jointly realized by a local voting method. To accelerate the voting on CUDA, a graphics processing unit (GPU)-oriented bitwise fast voting method is proposed, faster than the traditional histogram-based approach with two orders of magnitude. The whole algorithm is parallelized on CUDA at a fine granularity, efficiently exploiting the computing resources of GPUs. Our design is among the fastest stereo matching methods on GPUs. Evaluated in the Middlebury stereo benchmark, the proposed design produces the most accurate results among the real-time methods. The advantages of speed, accuracy, and desirable scalability advocate our design for practical applications such as robotics systems and multiview teleconferencing.**

***Index Terms*—Algorithm-architecture co-exploration, bitwise fast voting, parallel computing, real-time stereo matching, scalability.**

I. Introduction

STEREO matching is an important computer vision technique to extract depth information from stereo images. In the past decades, many stereo matching algorithms have been proposed and dramatically advanced this field in different aspects, e.g., matching accuracy, execution speed, and robustness. These algorithms can be categorized in two classes, i.e., global methods and local methods. Global methods formulate the objective of stereo matching as an energy function and solve it using different optimization methods such as belief propagation (BP) [1] and graph cuts [2]. Local methods improve matching accuracy by appropriately aggregating support from neighboring pixels [3]–[5]. Scharstein and Szeliski [6] provide a substantial survey and many stereo matching algorithms were evaluated using their method in the benchmark [7].

Despite the advances of algorithms for higher matching accuracy, stereo matching with both fast speed and accurate results is still challenging due to its huge computational workload. On the contrary, real-time performance is crucial to the applications such as robot navigation and multiview teleconferencing. Therefore, many research efforts tried to tackle this challenge. Recently, as a successful example of computer vision on graphics processing units (GPUs) [8], stereo reconstruction on (GPUs) drew much attention and obtained desirable speedup leveraging parallel computing [9]– [13]. Utilizing the horsepower of massive parallel processors, GPUs are effective in accelerating stereo algorithms by exploiting their potential parallelism.

In terms of GPU evolution, GPUs have migrated from a strict pipelined task-parallel architecture to a unified data-parallel programmable unit [14]. As a modern GPU architecture, compute unified device architecture (CUDA) [15] is supported by Nvidia GPUs for general-purpose parallel computing. It is successfully utilized in many fields, ranging from image processing to scientific computing. Compared to traditional GPUs, CUDA provides more flexibility in managing the GPU resources while maintaining huge computational power. Thanks to its increased generality and easier programmability, CUDA can be utilized for general-purpose processing on GPUs (GPGPU) with general mapping principles rather than ad hoc implementations [16].

In this paper, we present a stereo method on CUDA with several important advantages favored by practical applications, i.e., *high matching accuracy*, *real-time performance*, and *fine scalability*. Our method produces the most accurate disparity maps among the real-time methods in the Middlebury benchmark [7]. In addition, it is capable of preserving the object boundaries with high quality, which is desirable in many applications [17]. This paper extends our preliminary conference publication [18] and the major contributions are summarized as follows.

1) We propose a novel method to make the stereo matching algorithm scalable in terms of tradeoff between

Manuscript received March 21, 2010; revised September 6, 2010; accepted January 10, 2011. Date of publication March 28, 2011; date of current version July 7, 2011. The work presented in this paper was carried out when J. Lu was a Ph.D. student at Katholieke Universiteit Leuven, Leuven, Belgium. This paper was recommended by Associate Editor H. Sun.

K. Zhang and R. Lauwereins are with IMEC, Katholieke Universiteit Leuven, and IBBT, Leuven B-3001, Belgium (e-mail: zhangke@imec.be; lauwerei@imec.be).

J. Lu is with Advanced Digital Sciences Center, Singapore Pte Ltd., 138632, Singapore (e-mail: jiangbo.lu@illinois-singapore.com).

Q. Yang and G. Lafruit are with NVision Group, IMEC, Leuven B-3001, Belgium (e-mail: yangqi@imec.be; gauthier.lafruit@imec.be).

L. V. Gool is with Katholieke Universiteit Leuven, Leuven B-3001, Belgium, and Eidgenössische Technische Hochschule, Zurich CH-8032, Switzerland (e-mail: luc.vangool@esat.kuleuven.be).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2011.2133150

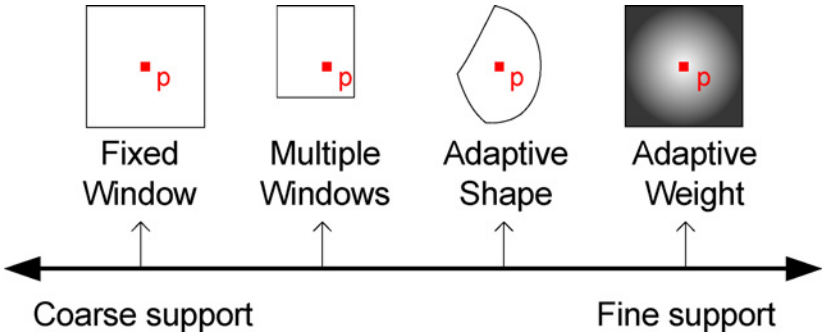


Fig. 1. Categories of local stereo matching algorithms based on the *fineness* of obtaining support from neighboring pixels for an anchor pixel. Four categories are listed from coarse to fine, i.e., fixed window methods, MWs methods, AS methods, and AW methods. One possible support region to the pixel p is presented for each category, respectively (brightness denotes the weight value for the AW methods).

execution speed and matching accuracy. The key idea is to produce a low-resolution initial disparity map and restore it to the original resolution by propagating reliable information across the neighborhood.

2) A GPU-oriented bitwise fast voting (BFV) method is proposed to accelerate the voting component of the algorithm. Running on GPUs, the BFV is two orders of magnitude faster than the traditional histogram-based method.
3) The whole algorithm is appropriately parallelized on CUDA. Efficient techniques of parallelizing algorithms on CUDA are exploited. Compared to the central processing unit (CPU) counterpart, we achieved dozens of times speedup without compromising the matching accuracy.

The rest of this paper is organized as follows. Section II introduces the related work. Section III presents our stereo matching algorithm. The BFV method is described in Section IV. In Section V, we parallelize the whole algorithm on CUDA. Section VI shows the experimental results and Section VII concludes this paper.

II. Related Work

A. Local Stereo Matching Algorithms

Many local algorithms have been proposed to improve the matching accuracy [3]–[5], [19]–[21]. In this paper, we categorize them according to their *fineness* in obtaining support from neighboring pixels for an anchor pixel. As shown in Fig. 1, the left side is the simplest fixed window (FW) or box filter method, in which a fixed square window is assigned to each anchor pixel as its support region. Though the method has the lowest complexity, it easily blurs object boundaries. A finer category is the multiple windows (MW) methods [3], [19]. MW selects one window or a combination of several windows from a number of candidates. The criterion is to select the support windows that produce a small matching cost. They attain much better results than the fixed window method. However, due to the shape of the support windows is limited in a small number of candidates, they are difficult to preserve delicate structures in disparity maps.

On the rightmost side, i.e., the category that has the finest support, is the adaptive weight (AW) method [4]. This method gives each pixel in the support region a supporting weight, which is adapted according to its intensity difference and spatial distance to the anchor pixel. The criterion of the adaptive method is that the pixels with similar intensity and small distance to the anchor pixel are more likely to have the same depth as the anchor pixel. The method is able to yield very accurate disparity maps, where the object boundaries are well preserved. However, it is quite computationally expensive [22] and requires a huge amount of memory [20].

Another category that lies between the MWs method and the AW method is the adaptive shape (AS) methods [5], [20], [21], where shape-adaptive support regions are constructed by approximating the local image structures. Unlike the MWs methods, the shape of support regions in AS is more flexible and is not restricted to be rectangles. By constructing arbitrarily shaped support regions, Zhang *et al.* [5] achieved very high matching accuracy, at a quality level similar to the AW method. Its support regions are constructed in a separable way and cost aggregation is accelerated by an orthogonal integral images technique. These advantages make the AS method [5] about two orders of magnitude faster than AW [4].

B. Real-Time and Accurate Stereo Matching on GPUs

Recently, several methods on GPUs have reached real-time performance while maintaining matching quality [23]–[25]. Yang *et al.* [23] implemented a global method based on BP. Though it achieved high matching accuracy, huge memory consumption could impede its prevalence when matching high-resolution images with a large disparity range. Wang *et al.* [24] presented an approach using dynamic programming (DP). By aggregating matching cost in the direction orthogonal to scanlines, it reduces the *streaking* artifacts associated with the traditional DP. However, the quality at depth-discontinuity regions is still not satisfactory. Using a cross-based local approach [5], Lu *et al.* [25] obtained fast speed on GPUs. Excluding the refinement stage of [5], this method [25] degrades the quality in occluded regions and large homogeneous regions. Whereas the aforementioned methods were implemented using traditional GPGPU, accurate stereo methods on CUDA are rarely reported so far. One recent design is [26], which adopts a semi-global algorithm. Since the algorithm is intrinsically serial

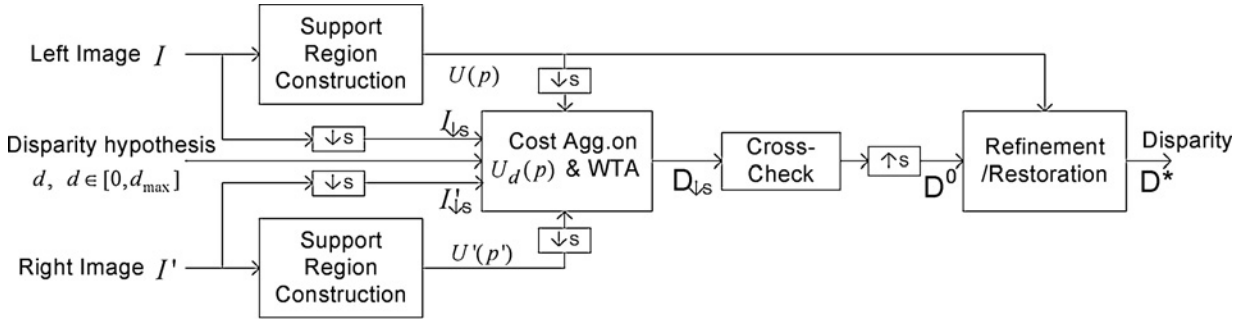


Fig. 2. Framework of the proposed algorithm. Scalability is realized by computing a low resolution initial disparity map and then restoring the disparity map to the original resolution.

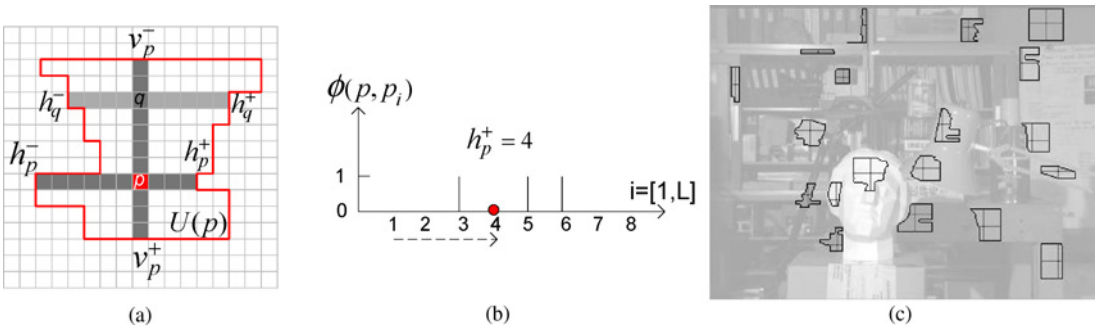


Fig. 3. (a) Support region $U(p)$ for the anchor pixel p is constructed using a cross-based approach. (b) Arms lengths are decided using a thresholding method with h_p^+ as an example. (c) Constructed support regions at sample pixels.

[26], the speedup obtained by parallel computing is small (4× speedup was attained compared to a CPU implementation).

III. Scalable Stereo Matching Algorithm

In this section, we present our stereo matching algorithm, which extends the concept in our earlier letter [5]. Compared with [5], our algorithm in this paper has several improvements. First, the support regions are constructed in a more computationally efficient way. Second, in the refinement stage, the reliability of initial disparity estimates is taken into account, improving the matching accuracy. Third, as an important extension, we propose a sample-and-restore (SAR) scheme to provide desirable scalability in terms of tradeoff between execution speed and matching accuracy.

The framework of our whole algorithm is summarized in Fig. 2. First, the supporting crosses are constructed for each pixel in both left image and right image. Then, raw matching costs are aggregated over support regions and an initial disparity map $D_{\downarrow S}$ is computed using the winner-takes-all (WTA) method. The refinement and the restoration are jointly realized with a local voting method, producing a high-quality disparity map with the same resolution as the input stereo images. Finally, simple post-process, i.e., a 3×3 median filter and a border extrapolation procedure, is applied to the disparity map. Next, we present the details of the algorithm.

A. Cost Aggregation Over Local Adaptive Support Regions

Given a pair of rectified stereo images, support regions $U(p)$ and $U'(p')$ are constructed for each pixel p in the left image I and p' in the right image I', respectively. $U(p)$ and $U'(p')$ are built from upright crosses. As shown in Fig. 3(a), a pixelwise cross is defined by a quadruple $\{h_p^-, h_p^+, v_p^-, v_p^+\}$, i.e., the left, right, up, and bottom arm length. The support region $U(p)$ is dynamically constructed by merging the horizontal arms of q, which lie in the vertical arms of p. Besides attaining high matching accuracy, the *cross-based* method of support region construction has several important benefits in improving computational efficiency. First, the support regions are compactly represented with quadruples, which amounts to memory efficiency. Second, aggregation over the support regions is separable and can be decomposed into two 1-D adding procedures [5]. Third, the computation among neighboring pixels can be reused, e.g., the sum over the horizontal arms of q can be used by multiple pixels which take q as a supporting pixel in the vertical direction. As a result, cost aggregation over a spatially-varying support region can still be efficiently completed with two 1-D aggregation.

To decide the arm lengths $\{h_p^-, h_p^+, v_p^-, v_p^+\}$, we use a thresholding method based on the intensity difference between the anchor pixel and its neighbors in four directions. The decision of h_p^+ is illustrated as an example in Fig. 3(b). The intensity difference between p and its right neighbors p_i (with a distance of i pixels to p) is evaluated as $\phi(p, p_i)$ as follows:

$$\phi(p, p_i) = \begin{cases} 0, & \max\limits_{c \in \{\mathrm{R,G,B}\}} (|I_c(p) - I_c(p_i)|) \le \tau \\ 1, & \text{otherwise} \end{cases} \tag{1}$$

where τ is a thresholding value and i increases from 1 to a maximum value L. The smallest i satisfying

$\phi(p, p_{i+1})\&\phi(p, p_{i+2}) = 1$ is chosen as h_p^+. Compared to filtering the input images using a median filter [5], the method suppresses isolated image noise in a computationally efficient way. Constructed support regions at sample pixels are illustrated in Fig. 3(c), where they desirably approximate local image structures and are suitable for cost aggregation.

In the cost aggregation stage at each disparity d, the raw matching cost is first computed using truncated sum of absolute differences as follows:

$$e_d(s) = \min\{\sum_{c\in\{\mathrm{R,G,B}\}} |I_c(s) - I_c(s')|, T\} \quad (2)$$

where $s' = (x_s - d, y_s)$ is the matching pixel of s in the right image. Then, the matching cost between $p = (x_p, y_p)$ in I and $p' = (x_p - d, y_p)$ in I' is the sum of the raw matching cost over the overlapping area $U_d(p)$ between support regions $U(p)$ and $U'(p')$ [5].

B. Voting-Based Disparity Refinement Using Reliable Initial Estimates

After cost aggregation, initial disparity estimates are selected using a WTA method. As shown in Fig. 4(a), the errors of initial estimates mainly lie in homogeneous regions and occluded regions. In homogeneous regions, the aggregated matching costs at different disparities are usually similar and lack of discriminative power, therefore yielding noisy results. In the occluded regions, since there is no correspondence for the pixels in the other view, a local WTA framework is difficult to handle these pixels. To tackle these problems, a refinement scheme is adopted in [5], effectively improving the matching accuracy [see Fig. 4(b)]. It is based on a local voting method with a piecewise smoothness prior. In this paper, by considering the reliability of initial estimates, we further improve the voting method to be more robust to outliers and produce more accurate results.

The reliability of initial estimates is determined by a left-right consistency check, i.e., the disparity of p in the left image should have the similar absolute value as the disparity of the point $(x_p - d_p, y_p)$ in the right image. If the disparity estimate d_p passes the consistency check, it is labeled as *reliable* $\eta(d_p) = 1$, otherwise it is labeled as *unreliable* $\eta(d_p) = 0$. Fig. 4(c) shows the reliability of the WTA results for the *Teddy* image, where the unreliable estimates mainly lie in the homogeneous regions and the occluded regions as discussed before.

We refine all the initial estimates with a similar local voting method. Different from [5], voting in the proposed method is performed using only the reliable estimates. At each pixel p, we build a histogram $\varphi_p(\cdot)$ over the reliable disparities in its support region as follows:

$$\varphi_p(d) = \sum_{s\in U(p)} \delta(d_s - d)\eta(d_s) \quad (3)$$

where $\delta(\cdot)$ is an impulse function. The disparity d_p^* associated with the peak of the histogram is taken as the optimal disparity for p as follows:

$$d_p^* = \arg\max \varphi_p(d), d \in [0, d_{max}]. \quad (4)$$

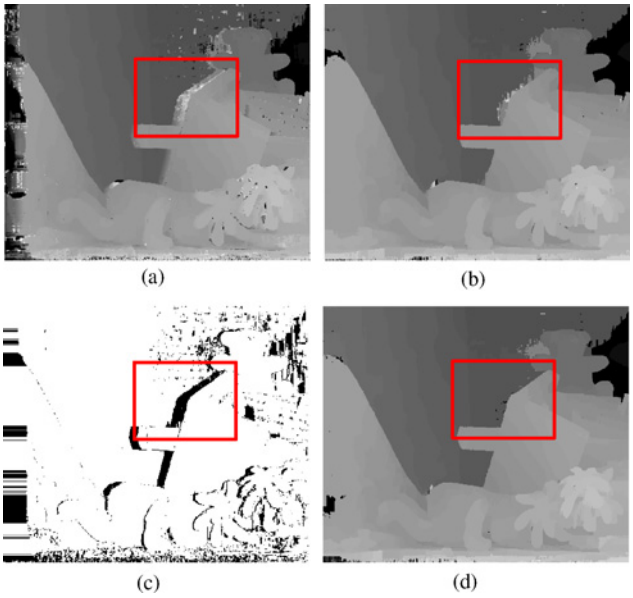


Fig. 4. (a) Disparity map selected by WTA. (b) Refined disparity map without considering the reliability of initial estimates [5]. (c) Unreliable initial estimates determined by the cross-check. (d) Refined disparity map by the proposed method.

As shown in Fig. 4(d), the proposed refinement improves the matching accuracy. First, it acts as a regularization in local support regions and the regularization effectively removes the spurious errors. Second, by propagating the disparity information of their neighbors, the refinement infers the disparity of occluded areas and often attains accurate results for the occluded pixels. Compared with [5] [Fig. 4(b)], the proposed method can produce more accurate results especially in occluded regions, e.g., the regions delineated by the rectangle in Fig. 4.

C. Scalable Stereo Matching Through a SAR Scheme

The key idea of achieving the quality-complexity scalability is to utilize the local voting stage as an upsampling or restoration method, more than as a refinement stage. In this section, we refer to it as a restoration stage instead of a refinement stage. The restoration stage is preformed on the initial disparity estimates and efficiently propagates information among local neighbors. We illustrate its efficiency with an example. In the example of Fig. 5(b), d_s is set to 0 for the points $\{s|(x_s \mod 2 = 1)||(y_s \mod 2 = 1)\}$ and they are labeled as unreliable. This means 75% initial estimates are discarded due to image downsampling. With the restoration, we can still reconstruct the disparity map with high quality [see Fig. 5(c)].

This encouraging result motivates us to design a computationally efficient method by computing WTA results in downsampled images and then utilize the proposed restoration stage to restore the disparity map. The method is referred as SAR and explained as follows. Given a pair of stereo images with a resolution of $W \times H$ and a sampling factor S, we compute an initial disparity map $D_{\downarrow S}$ with a resolution of $W/S \times H/S$. While the support regions are constructed on the original images I and I', the cost aggregation is performed

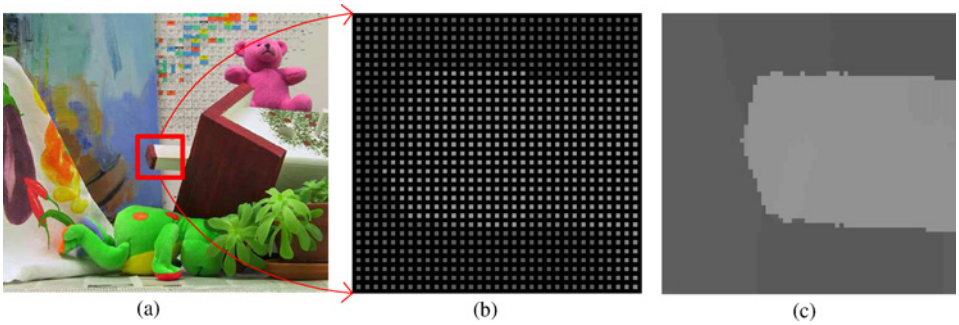


Fig. 5. Restoration from a low resolution disparity map with a voting method. (a) Left image of *Teddy*. (b) Expanded map D^0 from a low resolution ($W/2 \times H/2$) initial disparity map by stuffing 0s. (c) Restored disparity map using the proposed voting method.

on downsampled images $I_{\downarrow S}$ and $I'_{\downarrow S}$. The resolution of $I_{\downarrow S}$ is the same as $D_{\downarrow S}$. To calculate the $D_{\downarrow S}$, $I'_{\downarrow S}$ should contain all the possible correspondences of $I_{\downarrow S}$. $I'_{\downarrow S}$ has a resolution of $W \times H/S$, which is only downsampled in the vertical direction.

Given the $I_{\downarrow S}(x, y)$ and the $I'_{\downarrow S}(x, y)$, the raw matching cost is calculated using (2). The arm length of support crosses is also divided by S to form the support region $U_d(p)$. Based on the aggregated matching cost as introduced in (III-A), $D_{\downarrow S}$ is computed using the WTA method. The low resolution disparity map $D_{\downarrow S}$ is expanded to the original resolution D^0, with the invalid points stuffed with 0s and labeled as unreliable estimates shown in Fig. 5(a). Then, using the local voting method in (III-B), we assign the derived disparity value in (4) to each pixel and the disparity map D^0 is restored to D^* shown in Fig. 5(c). The high quality of the restoration is mainly due to three reasons. First, thanks to our cross-based construction method, most pixels in a local support region have similar disparity values and hence the disparity information can be shared among them. Second, the WTA produces a modest number of reliable initial estimates. Third, the proposed voting method reliably propagates the disparity information across pixels from the same support region.

In terms of computational efficiency, the cost aggregation stage is accelerated by a factor of S^2. The speedup of the whole algorithm can be computed using the Amdahl's law. Assuming the time of cost aggregation contributes a portion γ to the total execution time, the speedup of the whole algorithm is $\frac{1}{(1-\gamma)+\gamma/S^2}$. As an example, a speedup factor of 5 is achieved when $\gamma = 90\%$ and $S = 3$. The SAR scheme provides desirable scalability by adjusting the value of S.

IV. BFV on GPUs

To speedup the proposed algorithm on GPUs, one key challenge is to accelerate the refinement stage presented in Section III-B, where histograms need to be built at every pixel as in (3). Let N_p be the sample set of local support pixels $\{q\}$ for the anchor pixel p, and f_p^* be the optimal value to be voted from the set. The traditional voting needs to first build a histogram as follows:

$$\mathcal{H}_p(i) = \sum_{q \in N_p} \delta(f_q - i) \tag{5}$$

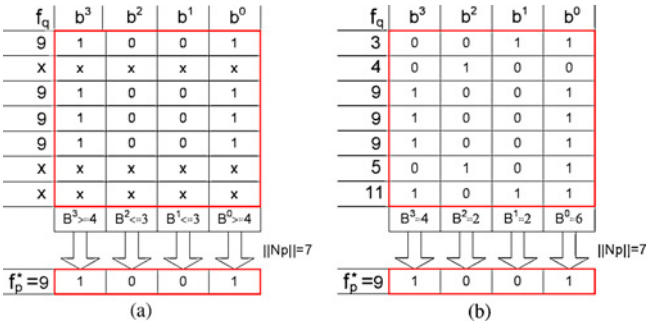


Fig. 6. In the BFV, each bit of the optimal value f_p^* is derived independent of the others. (a) Voting result is fully determined by the dominant value. (b) BFV could obtain the correct result, when f_p^* does not dominate the sample set.

and then select a maximum as follows:

$$f_p^* = \arg\max_i \mathcal{H}_p(i). \tag{6}$$

In terms of fast implementations, since the proposed refinement scheme in this paper needs to vote over non-rectangular support regions, some advanced techniques such as *integral histogram* [27] for a CPU-based implementation are inapplicable or hardly efficient. If this step is implemented on a CPU separately, it will severely affect the overall speed of the whole design besides occupying the CPU resources. When implemented on GPUs, histogram calculation is difficult to accelerate because of the serial and random memory updates [28]. As an instance of histogram extraction on GPUs (over rectangular regions), Shams and Kennedy [29] achieved 3–30 times speedup with thorough optimization. To speedup the voting, we propose a GPU-oriented BFV method, which is capable of accelerating the refinement stage with two orders of magnitude. The key idea is presented as follows.

Essentially speaking, in the proposed voting scheme, we are only concerned with the maximum but not the other values. If the votes of the maximum *dominate* the statistical set, i.e., the maximum is bigger than the sum of the values at the other bins, we can directly derive the maximum without building a histogram. Equivalently, this assumes $\mathcal{H}_p(f_p^*) > 0.5 \times \sum \mathcal{H}_p(i)$. Since most pixels belonging to the same local support region are likely to have the same disparity value, the assumption is often reasonable. Our method directly derives every bit of f_p^*

TABLE I
EXECUTION TIME OF THE HISTOGRAM-BASED VOTING (HIST.) ON A CPU AND THE PROPOSED BFV ON A GPU

No. of inputs (image resolution)	450×375
Sample set size ($\|N_p\|$)	32×32
No. of bins (n)	256
Histogram on a CPU (Core Duo 2.66 GHz)	500 ms
BFV on a GPU (GeForce8800 GTX)	3.6 ms

from N_p. Based on the assumption, the lth bit $b^l(f_p^*)$ of f_p^* is uniquely determined by the lth bit of the samples in N_p. Specifically, we sum the lth bit $b^l(f_q)$, which could be "1" or "0," of all the samples in N_p and the sum is denoted as $B^l(N_p)$ as follows:

$$B^l(N_p) = \sum_{q \in N_p} b^l(f_q). \tag{7}$$

If $B^l(N_p)$ is bigger than $0.5 \times \|N_p\|$, the lth bit of f_p^* is set to "1." Otherwise, the lth bit of f_p^* is "0" as follows:

$$b^l(f_p^*) = \begin{cases} 1, B^l(N_p) > 0.5 \times \|N_p\| \\ 0, \text{otherwise.} \end{cases} \tag{8}$$

In Fig. 6(a), an example is used to illustrate the algorithm. "$\times$" represents a random value of "1," or "0," indicating that the voting result is fully determined by the samples with the value of f_p^*, regardless of the other samples. In the example, there are four "9" in seven samples ($\|N_p\| = 7$), so "9" dominates the sample set. At each bit l, the relation between $B^l(N_p)$ and 0.5×7 is fully determined by $b^l(9)$, therefore f_p^* is derived as "9" with (8).

The assumption of $\mathcal{H}_p(f_p^*) > 0.5 \times \sum \mathcal{H}_p(i)$ guarantees that the proposed BFV approach obtains the same results as the traditional histogram-based method. However, even when the assumption is violated, the BFV still has a good chance to get the correct results. This is because f_q could have the same bitwise value as f_p^* at some bit, even if f_q is different from f_p^*. For example, if we replace the first "9" in the f_q column of Fig. 6(a) with a random value "$\times$" (the assumption is violated), the result will still be correct if one of the four "$\times$"s at bit l is the same as $b^l(f_p^*)$, with $l = 0, 1, 2, 3$, respectively. A more concrete example is given in Fig. 6(b). The effectiveness of the BFV method in improving matching accuracy has been confirmed by the experimental results in Section VI.

In terms of implementation on GPUs, the proposed BFV has several important advantages compared to the histogram-based voting. First of all, it reduces the memory consumption from n to $log_2(n)$, where n is the number of histogram bins. This advantage is essential to effective computing on CUDA. Since CUDA has limited registers for each thread (around ten to activate the maximum number of threads) and shared memory for each multiprocessor (e.g., 16 kB in GeForce8800 GTX), a large memory consumption would greatly reduce the number of active threads [16]. Second, our method allows parallel computing among bits because each bit is derived independently. Different from the serial and random memory updates in the histogram-based method, the memory access in the BFV is data-independent. Third, similar to the cost aggregation in Section III-A, the voting for a 2-D region in (7) is separable and can be completed by two 1-D adding procedures.

Thanks to these advantages, the proposed method runs quite fast on GPUs. The performance of the BFV is shown in Table I, compared with the histogram-based method. Running on a GeForce8800 GTX GPU, our BFV only needs 3.6 ms to compute the maximum values for 450×375 local histograms, with 32×32 sample set size and 256 bins for each local histogram. Compared to a histogram-based approach on an Intel Core Duo 2.66 GHz CPU, a speedup factor of 140 is achieved by the proposed method. The BFV significantly reduces the time penalty of the disparity refinement and restoration to be minor. For instance, this BFV step consumes only 4% of total execution time in matching the *Teddy* images [7].

V. STEREO MATCHING ON CUDA

A. CUDA Parallel Processing Architecture

CUDA is provided by Nvidia GPUs for general-purpose parallel computing. CUDA is implemented as a set of single instruction multiple threads processors. One *kernel* function is executed by multiple threads simultaneously. For example, in a GeForce8800 GTX GPU, a maximum number of 768 threads could be activated. With huge data-parallel computing power, CUDA is well-suited to speedup algorithms with high data parallelism.

With a unified programming model, CUDA provides more flexibility to utilize GPU resources. Its programming model mainly consists of a hierarchy of threads and memory [16]. Single threads are organized as thread blocks, facilitating data processing in a coordinated manner. A grid consists of multiple thread blocks with the same size. The thread blocks are executed independently and the transparent scalability of CUDA, i.e., the same code runs faster on advanced GPUs, is realized mainly by adapting the number of active thread blocks. Global memory is off-chip and accessible by all threads. Shared memory is on-chip memory and accessible by the threads from the same block. In addition, each thread has a small number of private registers. The number of active threads could be limited by either computing units or on-chip memory resources. Key to the performance on CUDA is to activate massive threads on a large number of cores and hide the memory latency with computation [16].

B. Parallelize Stereo Matching on CUDA

The framework of our design on CUDA is shown in Fig. 7. There are mainly five functional kernels, i.e., building support windows, cost aggregation in horizontal direction, cost aggregation in vertical direction and WTA, cross-check of left and right disparity maps, and refining the WTA results of the left disparity map, respectively. To reduce the data access to the off-chip global memory, the computation of pixelwise raw matching cost is merged into the kernel of horizontal

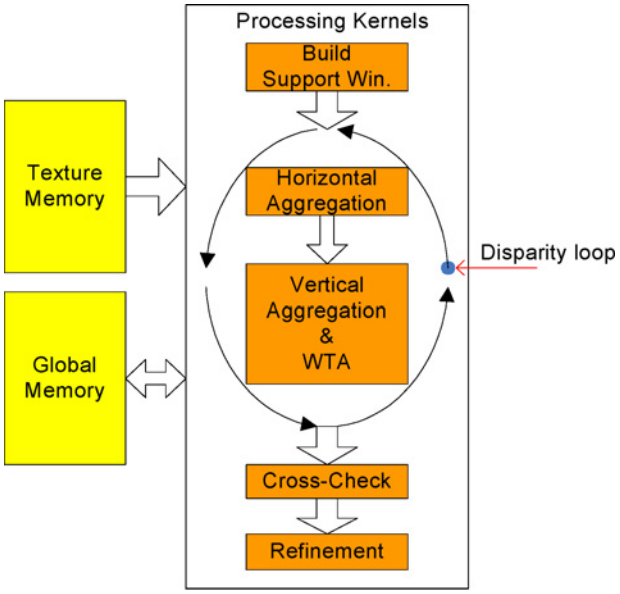


Fig. 7. Implementation framework of the proposed design on CUDA, with five sequential processing kernels.

cost aggregation. Since the input stereo images are frequently accessed and not modified during the whole algorithm, they are stored in the texture memory which has a hardware-implemented cache scheme. The image-level intermediate data are stored in the global memory.

In the cost aggregation stage, the disparity hypothesis $d_i \in [d_0, d_1, \ldots, d_{max}]$ is taken as the outermost loop. For both left and right images, we store the smallest matching cost among $[d_0, d_{i-1}]$ and the corresponding disparity value achieving the smallest cost at each pixel. At d_i, based on the aggregated matching cost, the stored values at p for the left image and at $p' = (x_p - d_i, y_p)$ for the right image are updated. At the end of the loop, two disparity maps are obtained and can be used for the cross-check. By this method, the storage to get the WTA results is reduced to four images and is independent of the disparity range.

In order to utilize the computing power of CUDA, the algorithm is first partitioned into fine-grained parallel subtasks. For each kernel, outputs are fully independent of each other, therefore parallelism granularity of all the kernels is pointwise. Each function kernel in Fig. 7 is parallelized according to its output data, respectively. We create a number of threads in each kernel, same as the resolution of the input images. Each thread performs the computation for one output, i.e., a thread with thread index (x, y) is corresponding to the output with image position (x, y). Considering that the maximum number of threads per block is 512 and that limited shared memory is allocated for each block, 16×16 thread blocks are chosen for our design.

C. Data Reuse Utilizing Shared Memory

Since access to off-chip memory is slow (200–300 cycles), a primary concern to optimize the performance on CUDA is managing the off-chip memory latency [16]. One intrinsic scheme of CUDA is to activate massive thread warps and hide the memory latency by performing zero-overhead scheduling among them. However, it cannot guarantee the peak performance for many algorithms which either have a small number of threads or are bound on memory bandwidth. An effective technique to alleviate off-chip memory access is data reuse utilizing the fast shared memory that is allocated to each thread block.

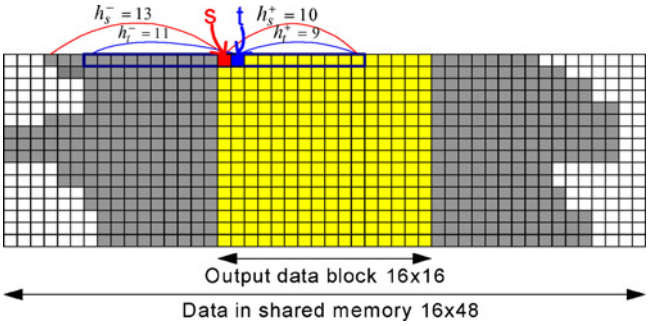


Fig. 8. Data reuse among different threads in one thread block by utilizing shared memory.

In our design, all the kernels except the cross-check have great potential of sharing data among threads in one block. We take the kernel of horizontal aggregation to illustrate our technique of data reuse. As shown in Fig. 8, each thread at s is to aggregate input data over a horizontal line segment, which is covered by the horizontal arms of h_s^- and h_s^+. Typically, there is a large overlap between the accessed data of thread s and that of its neighboring thread t as shown in Fig. 8. Therefore, we store the data in shared memory for reuse. For a thread block with 16×16 threads, an *apron* of data [30] is loaded as shown in the gray area of Fig. 8. The maximum amount of required data is $16 \times (16 + L + L)$ with L as the maximum arm length.

Considering both implementation efficiency and matching accuracy, we set L to 16 and store a 16×48 block of data in shared memory for each thread block. In terms of implementation efficiency, the first advantage is that data in off-chip memory can be loaded to shared memory in parallel, i.e., three 16×16 data blocks are loaded by the 16×16 thread block consecutively. In addition, when off-chip data is stored in global memory, all the accesses are coalesced to 16-word reading. The proposed technique is applied to other kernels similarly. For instance, in the kernel of vertical aggregation, a 48×16 block of data is stored in shared memory.

The technique of data reuse greatly reduces the off-chip memory accesses. To give a concise analysis, we can view the aggregation above as a 1-D image convolution. Assuming the average of all arm lengths is $\overline{L}$, $(2 \times \overline{L} + 1) \times W \times H$ pixels have to be loaded for one $W \times H$ image without using the technique. In contrast, the amount is reduced to $3 \times W \times H$ with the proposed scheme of data reuse. For example, the saving factor is 9 when $\overline{L} = 13$. Observe that, since the arm lengths of different pixels vary arbitrarily, direct accesses from the global memory are hard to be coalesced.

VI. Experimental Results and Discussions

We evaluate the matching accuracy and the execution speed of our method using the Middlebury stereo database [7]. The

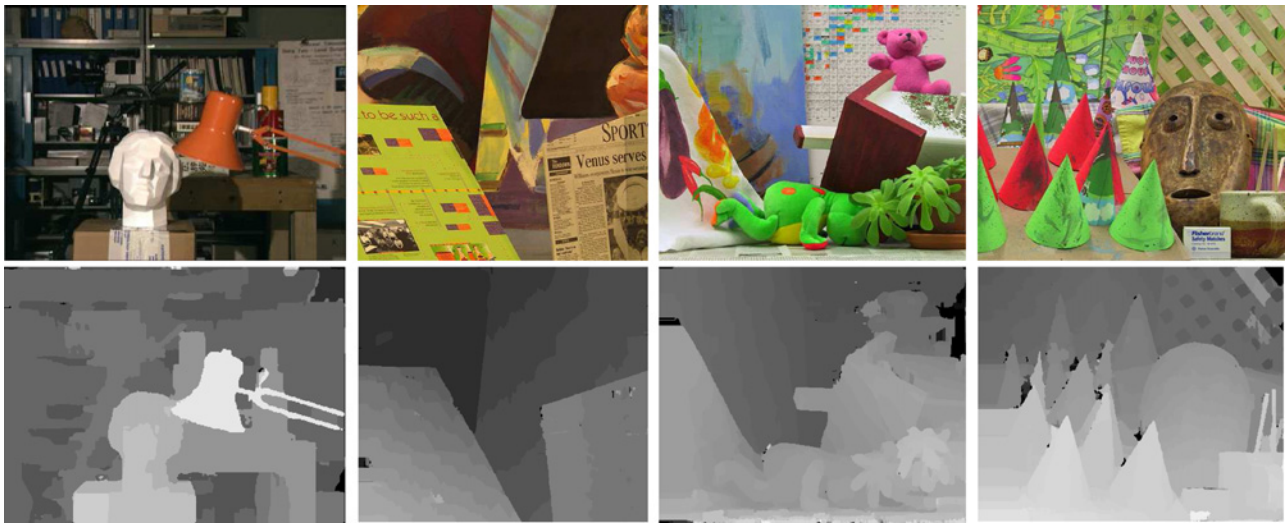


Fig. 9. Input images (top row) and produced disparity maps (bottom row) with the proposed method, from left to right, for the *Tsukuba*, *Venus*, *Teddy*, and *Cones* images.

TABLE II
QUANTITATIVE EVALUATION RESULTS FOR THE MIDDLEBURY STEREO DATABASE [7]

Algorithm	*Tsukuba*			*Venus*			*Teddy*			*Cones*			Ave.
	nocc.	all	disc.	nocc.	all	disc.	nocc.	all	disc.	nocc.	all	disc.	
VariableCross [5]	1.99	2.65	6.77	0.62	0.96	3.20	9.75	15.1	18.2	6.28	12.7	12.9	7.60
Our method	**1.64**	**2.13**	**6.43**	**0.57**	**0.90**	**2.90**	**9.70**	**14.8**	**19.3**	**7.00**	**12.5**	**13.6**	**7.63**
RealtimeBP [23]	1.49	3.40	7.87	0.77	1.90	9.00	8.72	13.2	17.2	4.61	11.6	12.4	7.69
Lu [25]	2.80	4.84	7.29	2.14	3.40	11.5	9.67	16.3	18.8	5.85	13.7	12.1	9.03
Our method ($\downarrow 2\times \downarrow 2$)	**2.05**	**2.59**	**6.87**	**1.76**	**2.30**	**7.91**	**12.0**	**17.2**	**22.7**	**8.17**	**13.9**	**15.5**	**9.41**
RealtimeGPU [24]	2.05	4.22	10.6	1.92	2.98	20.3	7.23	14.4	17.6	6.41	13.7	16.5	9.82

same parameters are used for all the tested images, i.e., the thresholding value $\tau = 20$, the maximum arm length $L = 16$, and the truncation value of raw matching cost $T = 60$. The program completely runs on a GeForce8800 GTX GPU, which supports native 32-bit floating point arithmetic operations. We first show the experimental results without using the SAR scheme and then illustrate the scalability of the proposed stereo method by applying the SAR scheme.

A. Evaluation of Stereo Matching Accuracy and Speed

The produced disparity maps by the proposed method are shown in Fig. 9, where high quality is obtained in most regions. Fig. 10 shows the disparity maps for the *Tsukuba* image, comparing some state-of-the-art methods with real-time performance. Compared to their results, our method preserves object borders more accurately [see Fig. 10(b)], which is usually favored by applications of robotics and multiview rendering [17]. Compared to [25] [see Fig. 10(e)], our design effectively improves the matching accuracy by the BFV-based refinement, producing more accurate results in both occluded regions and homogeneous regions. Table II reports the percentage of bad pixels, where a disparity error is bigger than one pixel. The error rate in non-occluded (nocc.) regions, all regions, and depth-discontinuity (disc.) regions are reported, respectively. The algorithms are listed in ascending order of the average percentage of bad pixels (Ave.). We observe that our method obtains the highest rank among the leading real-time methods.

Table III compares the speed of our design and the referenced work. Our design attains 15 f/s in matching images with a 475×375 resolution and 64 disparities. Compared to the recent stereo design on CUDA [26], the proposed design is more than twice faster. Compared to [24], our design completely runs on a GPU and frees the CPU. It is dozens of times faster than the previous CPU implementation [5] without compromising the matching accuracy. The large speedup mainly attributes to twofold contributions. First, the stereo algorithm has been optimized for parallel computing and its data parallelism is fully exploited. Second, the resources of CUDA are properly utilized according to their strengths and constraints. Whereas [25] was implemented with the traditional GPGPU approach, our design exploits the computing power of CUDA. The comparison also suggests that, compared to the traditional GPGPU, CUDA maintains the huge computational power while providing more generality and programmability. The increased programmability facilitates the implementation of effective algorithm components. The programming model with more generality also increases the portability of CUDA programs.

B. Scalability of the Proposed Method with the SAR Scheme

We show the scalability of the proposed method by applying the SAR scheme. As introduced in Section III-C, an initial

TABLE III
SPEED PERFORMANCE OF OUR DESIGN COMPARED WITH SEVERAL OTHER STEREO METHODS

	Image Size	Disparities	Speed (f/s)	MDE/s	Processors
Our Method	**450 × 375**	**64**	**15**	**162**	**GeForce 8800 GTX**
Gibson [26]	450 × 375	64	6	64.8	GeForce 8800 GTX
Our Method	**320 × 240**	**16**	**93**	**114.3**	**GeForce 8800 GTX**
RealtimeGPU [24]	320 × 240	16	43	52.8	ATI's Radeon XL1800 + 3GHz CPU
RealtimeBP [23]	320 × 240	16	17	20.9	GeForce 7900 GTX
Our Method	**384 × 288**	**16**	**64**	**113.2**	**GeForce 8800 GTX**
Lu [25]	384 × 288	16	17	30.0	GeForce 7900 GTX
Zhang [5]	384 × 288	16	1.1	1.9	3GHz CPU

MDE/s = million disparity evaluations per second.

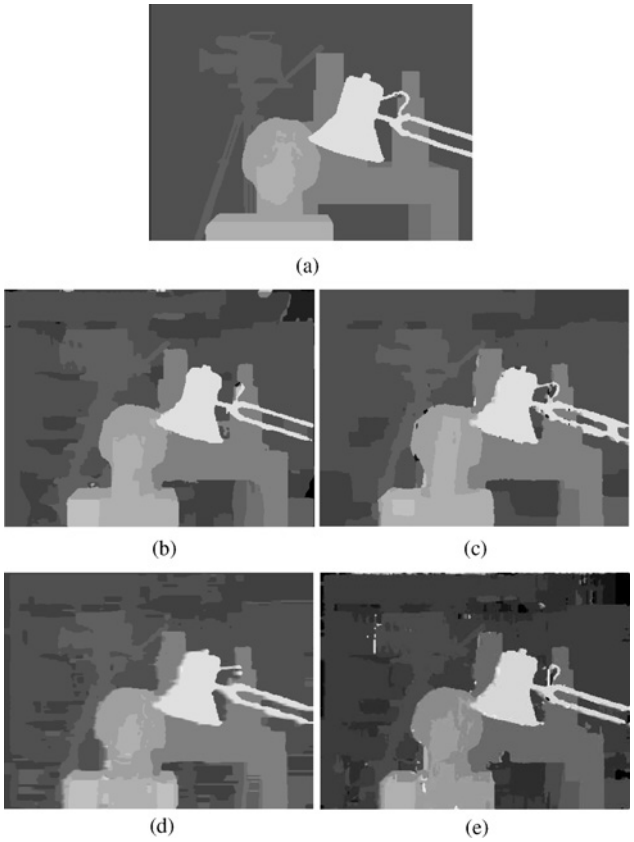


Fig. 10. (a) Ground truth and produced disparity maps by (b) our method, (c) RealtimeBP [23], (d) RealtimeGPU [24], and (e) Lu [25] for the *Tsukuba* image.

disparity map with a low resolution is first produced and then restored to the original resolution using the BFV. We use the sampling factor $\downarrow S_w \times \downarrow S_h$ to denote a SAR configuration in which the initial disparity map $D_{\downarrow}$ has a resolution of $W/S_w \times H/S_h$, i.e., downsampled by S_w in width and S_h in height, respectively. Execution speed and matching accuracy are tested for different sampling factors to depict the scalability of the proposed method. The experiments are performed on the *Cones* image, which has a resolution of 450 × 375 and a disparity search range of 60.

Fig. 12 shows the variation of speed and error rate, depending on the sampling factor. We can observe that the execution speed dramatically increases when large sampling factor are used, e.g., 1.8× speedup for $\downarrow 1\times \downarrow 2$ sampling and 2.8× speedup for $\downarrow 2\times \downarrow 2$ sampling. As analyzed in III-C, the speedup of the whole algorithm also depends on the portion γ which is taken by the cost aggregation stage in the total execution time. Using CUDA Visual Profiler, we profile the complexity of different algorithm components in Table IV. Because the matching cost is aggregated at multiple disparity values, the cost aggregation stage consumes most of the execution time, i.e., 84%. The total speedup is $\frac{1}{0.16+0.84/(S_w \times S_h)}$ according to the Amdahl's law, as confirmed by the result in Fig. 12. With a sampling factor of $\downarrow 3\times \downarrow 3$, the proposed method achieves the speed of 64 f/s and 648 MDE/s. It is faster than the state-of-the-art methods in Table III with one order of magnitude. The value of γ increases along with the disparity range because the range value only affects the cost aggregation stage. Therefore, with the SAR method, there is more speedup potential for high resolution input images with a larger disparity search range.

TABLE IV
EXECUTION TIME ON GPUS AND THE PORTION IN OVERALL TIME OF DIFFERENT ALGORITHM COMPONENTS, WITH IMAGE RESOLUTION OF 475 × 375 AND 60 DISPARITIES

Algorithm Component	Execution Time (ms)	Portion (%)
Build support windows	5.0	8
Cost aggregation and WTA	52.5	84
Cross-check and refinement	3.1	5
Median filter and Border handling	1.3	2

On the contrary, the degradation of the matching accuracy is quite moderate as shown in Fig. 12. Compared to the result of $\downarrow 1\times \downarrow 1$ (without sampling), the error rate of using $\downarrow 3\times \downarrow 3$ sampling increases less than 3%. The stereo matching results at four sampling factors are shown in Fig. 11. For the sample factor of $\downarrow 2\times \downarrow 2$, we also show the quantitative results of the produced disparity maps in Table II. We can observe that the high visual quality of the disparity map remains for all the sampling factors. Even using the large sampling factor $\downarrow 3\times \downarrow 3$, the delicate structures are well preserved (marked with a rectangle) and the quality is still state of the art in real-time methods. Considering both the matching accuracy and execution speed, our method reaches considerable speedup at the expense of minor accuracy degradation.

In summary, the experimental results have validated three major advantages of the proposed method, i.e., fast speed, high matching accuracy, and fine scalability. In the Middlebury

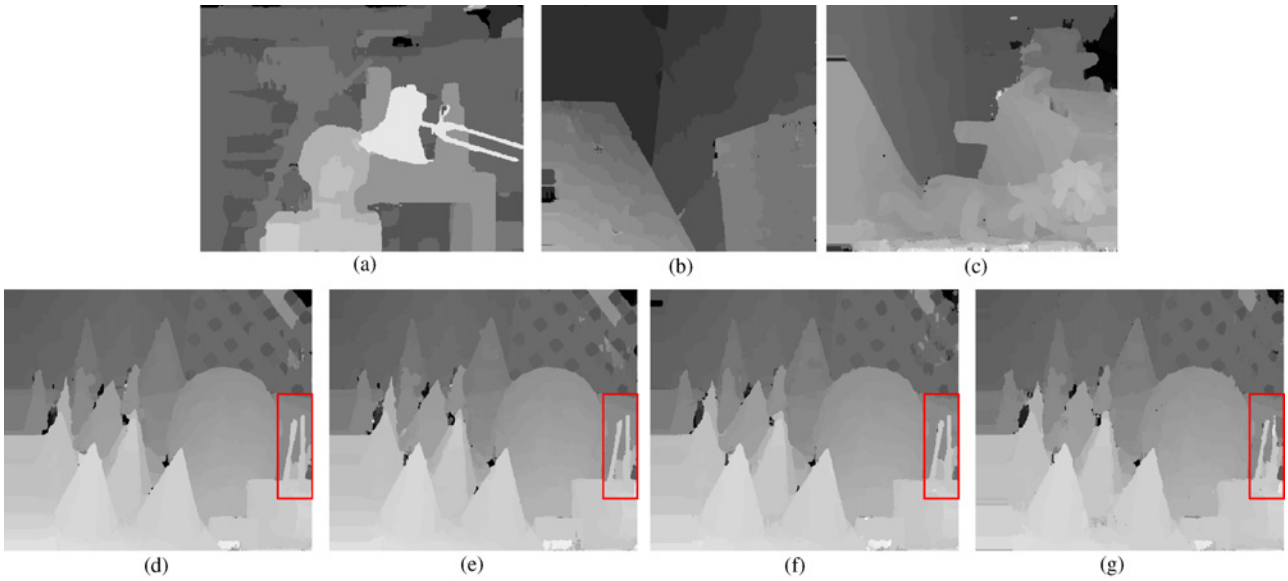


Fig. 11. Top row: produced disparity maps for (a) *Tsukuba* image, (b) *Venus* image, and (c) *Teddy* image at the constant sampling factor of $\downarrow 2\times \downarrow 2$. Bottom row: produced disparity maps for *Cones* image and the corresponding speed at different sampling factors, i.e., (d) $\downarrow 1\times \downarrow 2$, (e) $\downarrow 2\times \downarrow 2$, (f) $\downarrow 2\times \downarrow 3$, and (g) $\downarrow 3\times \downarrow 3$.

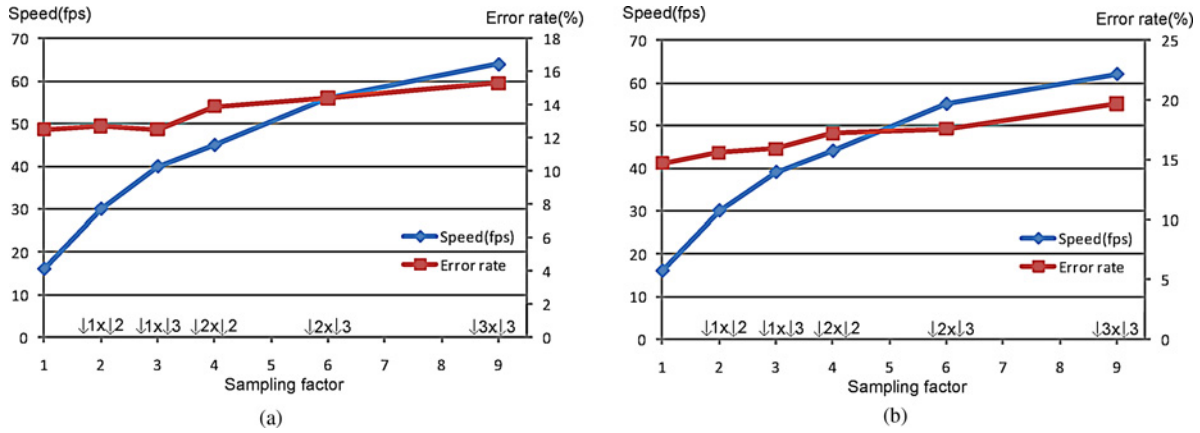


Fig. 12. Execution speed and matching accuracy of the proposed method varies with different sampling factors, where $\downarrow S_w \times \downarrow S_h$ denotes the left image is downsampled by S_w in width and by S_h in height for cost aggregation. (a) Results on the *Cones* image. (b) Results on the *Teddy* image.

stereo benchmark, our method obtains the highest rank in matching accuracy among the real-time methods. By changing the sampling factor, we can instantiate the method with desirable tradeoffs between matching accuracy and execution speed.

VII. Conclusion

This paper presented a real-time and accurate stereo method on GPUs. A SAR scheme was proposed, obtaining desirable scalability in terms of execution speed and matching accuracy. The high accuracy and the scalability attributed to a local voting method. A BFV method was proposed to accelerate the voting method on GPUs. The whole algorithm was parallelized on CUDA, attaining dozens of times speedup compared to the CPU counterpart. Experimental results showed the superiority of the proposed design in both matching accuracy and execution speed. With real-time performance and high accuracy, our design was suitable for practical applications. In addition, the achieved speedup showed the importance of co-exploring parallel algorithms and computing architectures in GPGPU. The design flow with desirable generality suggested that, compared to the traditional GPU architectures, CUDA provided more design flexibility while maintaining the huge computational power.

References

[1] J. Sun, N. Zheng, and H. Shum, "Stereo matching using belief propagation," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 25, no. 7, pp. 787–800, Jul. 2003.

[2] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, Nov. 2001.

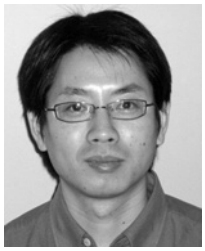
[3] O. Veksler, "Fast variable window for stereo correspondence using integral images," in *Proc. IEEE Comput. Vision Patt. Recog.*, vol. 1. Jun. 2003, pp. 556–561.

[4] K. J. Yoon and S. Kweon, "Adaptive support-weight approach for correspondence search," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 28, no. 4, pp. 650–656, Apr. 2006.

[5] K. Zhang, J. Lu, and G. Lafruit, "Cross-based local stereo matching using orthogonal integral images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 7, pp. 1073–1079, Jul. 2009.
[6] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Int. J. Comput. Vision*, vol. 47, no. 1, pp. 7–42, May 2002.
[7] Middlebury Stereo Vision Page [Online]. Available: http://vision.middlebury.edu/stereo
[8] OpenVIDIA [Online]. Available: http://openvidia.sourceforge.net/index.php
[9] R. Yang, M. Pollefeys, and S. Li, "Improved real-time stereo on commodity graphics hardware," in *Proc. IEEE Comput. Vision Patt. Recog. Workshops*, Jun. 2004, p. 36.
[10] N. Cornelis and L. V. Gool, "Real-time connectivity constrained depth map computation using programmable graphics hardware," in *Proc. IEEE Comput. Vision Patt. Recog.*, vol. 1. Jun. 2005, pp. 1099–1104.
[11] M. Gong and R. Yang, "Image-gradient-guided real-time stereo on graphics hardware," in *Proc. Int. Conf. 3-D Digital Imag. Model.*, 2005, pp. 548–555.
[12] C. Zach, A. Klaus, and K. Karner, "Accurate dense stereo reconstruction using graphics hardware," in *Proc. Eurographics*, 2003, pp. 227–234.
[13] J. Lu, S. Rogmans, G. Lafruit, and F. Catthoor, "Stream-centric stereo matching and view synthesis: A high-speed approach on GPUs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 11, pp. 1598–1611, Nov. 2009.
[14] J. Owens, M. Houston, D. Luebke, S. Green, J. Stone, and J. Phillips, "GPU computing," *Proc. IEEE*, vol. 96, no. 5, pp. 879–899, May 2008.
[15] Nvidia CUDA [Online]. Available: http://developer.nvidia.com/object/cuda.html
[16] S. Ryoo, C. Rodrigues, S. Baghsorkhi, S. Stone, D. Kirk, and W. Hwu, "Optimization principles and application performance evaluation of a multithreaded GPU using CUDA," in *Proc. 13th ACM SIGPLAN Symp. Principles Practice Parallel Program.*, 2008, pp. 73–82.
[17] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," *ACM Trans. Graphics*, vol. 23, no. 3, pp. 600–608, 2004.
[18] K. Zhang, J. Lu, G. Lafruit, R. Lauwereins, and L. V. Gool, "Real-time accurate stereo with bitwise fast voting on CUDA," in *Proc. Int. Conf. Comput. Vision Workshops*, 2009, pp. 794–800.
[19] H. Hirschmuller, P. Innocent, and J. Garibaldi, "Real-time correlation-based stereo vision with reduced border errors," *Int. J. Comput. Vision*, vol. 47, nos. 1–3, pp. 229–246, 2002.
[20] J. Lu, G. Lafruit, and F. Catthoor, "Anisotropic local high-confidence voting for accurate stereo correspondence," *Proc. SPIE-IS&T Electron. Imag.*, vol. 6812, pp. 1–10, Jan. 2008.
[21] K. Zhang, J. Lu, and G. Lafruit, "Scalable stereo matching with locally adaptive polygon approximation," in *Proc. Int. Conf. Image Process.*, 2008, pp. 313–316.
[22] F. Tombari, S. Mattoccia, and L. Stefano, "Classification and evaluation of cost aggregation methods for stereo correspondence," in *Proc. IEEE Comput. Vision Patt. Recog.*, Jun. 2008, pp. 1–8.
[23] Q. Yang, L. Wang, R. Yang, S. Wang, M. Liao, and D. Nister, "Real-time global stereo matching using hierarchical belief propagation," in *Proc. British Mach. Vision Conf.*, 2006, pp. 989–998.
[24] L. Wang, M. Liao, M. Gong, R. Yang, and D. Nister, "High quality real-time stereo using adaptive cost aggregation and dynamic programming," in *Proc. Int. Symp. 3-D Data Process., Visualiz. Transmission*, 2006, pp. 798–805.
[25] J. Lu, K. Zhang, G. Lafruit, and F. Catthoor, "Real-time stereo matching: A cross-based local approach," in *Proc. Int. Conf. Acou., Speech, Signal Process.*, 2009, pp. 733–736.
[26] J. Gibson and O. Marques, "Stereo depth with a unified architecture GPU," in *Proc. IEEE Comput. Vision Patt. Recog. Workshops*, Jun. 2008, pp. 1–6.
[27] F. Porikli, "Integral histogram: A fast way to extract histograms in Cartesian spaces," in *Proc. IEEE Comput. Vision Patt. Recog.*, vol. 1. Jun. 2005, pp. 829–836.
[28] V. Podlozhnyuk, "Histogram calculation in CUDA," NVIDIA, Santa Clara, CA, Tech. Rep., 2007.
[29] R. Shams and R. A. Kennedy, "Efficient histogram algorithms for Nvidia CUDA compatible devices," in *Proc. Int. Conf. Signal Process. Commun. Syst.*, 2007, pp. 418–422.
[30] V. Podlozhnyuk, "Image convolution with CUDA," NVIDIA, Santa Clara, CA, Tech. Rep., 2007.

Ke Zhang (S'08) received the B.S. and M.S. degrees in electrical engineering from Zhejiang University, Hangzhou, China, in 2005 and 2007, respectively. He is currently pursuing the Ph.D. degree from the Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven, Belgium.

Since 2007, he has been a Researcher with IMEC, Leuven. His current research interests include multimedia processing systems, computer vision, and video coding.



Jiangbo Lu (M'09) received the B.S. and the M.S. degrees from Zhejiang University, Hangzhou, China, in 2000 and 2003, respectively, and the Ph.D. degree from Katholieke Universiteit Leuven, Leuven, Belgium, in 2009, all in electrical engineering.

From April 2003 to August 2004, he was with VIA-S3 Graphics, Shanghai, China, as a Graphics Processing Unit Architecture Design Engineer. In 2002 and 2005, he was a Visiting Researcher with Microsoft Research Asia, Beijing, China. Since October 2004, he has been a Ph.D. Researcher with the Multimedia Group, Interuniversity Microelectronics Center, Leuven, where he pioneered and coordinated research on real-time stereo matching and view synthesis. Since September 2009, he has been working as a Post-Doctoral Researcher with the Advanced Digital Sciences Center, Singapore, a joint research center between the University of Illinois at Urbana-Champaign (UIUC), Urbana, and the Agency for Science, Technology, and Research, Singapore. He also holds an adjunct appointment as a Principal Research Affiliate with the Coordinated Science Laboratory, UIUC. His current research interests include many aspects of interactive multimedia applications over networks, ranging from theory, algorithms to integrated systems.

Qiong Yang (M'03) received the Ph.D. degree from Tsinghua University, Beijing, China, in 2004.

From 2004 to 2007, she was an Associate Researcher with Microsoft Research Asia, Beijing. In September 2007, she joined VISICS Group, Katholieke Universiteit Leuven, Leuven, Belgium. Currently, she has been with IMEC, Leuven, as a Senior Research Scientist since January 2009. Her current research interests include pattern recognition and machine learning, feature detection/extraction and matching, visual tracking and detection, object segmentation and cutout, stereo and multiview vision, and multichannel fusion and visualization.

Gauthier Lafruit (M'99) received the Masters and Ph.D. degrees in electrical engineering from the Free University of Brussels, Brussels, Belgium, in 1989 and 1995, respectively.

From 1989 to 1994, he was with the Belgian National Foundation for Scientific Research as a Research Scientist, active in the area of wavelet image compression. In 1996, he joined the Multimedia Group with IMEC, Leuven, Belgium, as a Senior Scientist. Currently, he is the Principal Scientist of Vision Systems with the Department of Smart Systems and Energy Technology, IMEC. He has acquired image processing expertise in various applications, video coding, multicamera acquisition, multiview rendering, image analysis, and video stereoscopy, where he has applied the "Triple-A" philosophy in application-algorithm-architecture tradeoff studies.

Dr. Lafruit received the Barco Scientific Award on the initiative of Barco NV and the Belgian National Foundation for Scientific Research, for his contribution to efficient hardware implementations in wavelet image processing applications. He is currently an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY.

Rudy Lauwereins (SM'97) received the Ph.D. degree in electrical engineering in 1989.

He is currently the Vice President of IMEC, Leuven, Belgium, which performs world-leading research and delivers industry-relevant technology solutions through global partnerships in nanoelectronics, information and communications technology, healthcare, and energy. He is responsible for IMEC's Smart Systems Technology Office, covering energy efficient green radios, vision systems, (bio)medical and lifestyle electronics as well as wireless autonomous transducer systems and large area organic electronics. He is also a part-time Full Professor with the Katholieke Universiteit Leuven, where he teaches computer architectures in the Master of Science in Electrotechnical Engineering Program. Before joining IMEC in 2001, he held a tenure Professorship with the Faculty of Engineering at the Katholieke Universiteit Leuven since 1993. He has authored and co-authored more than 360 publications in international journals, books, and conference proceedings.

Luc Van Gool (M'85) received the Electromechanical Engineering degree from the Katholieke Universiteit Leuven, Leuven, Belgium, in 1981.

Currently, he is a Professor with the Katholieke Universiteit Leuven and the Eidgenössische Technische Hochschule, Zurich, Switzerland. He leads computer vision research at both places and he also teaches computer vision. He has authored over 200 papers in this field. His current research interests include 3-D reconstruction and modeling, object recognition, and tracking and gesture analysis. He is a co-founder of five spin-off companies.

Prof. Gool has received several Best Paper Awards. He has been a program committee member of several major computer vision conferences.