EURASIP Journal on Image and Video Processing
a SpringerOpen Journal

RESEARCH Open Access

Real-time stereo vision system using adaptive weight cost aggregation approach

Jingting Ding[1], Jilin Liu[1], Wenhui Zhou[2], Haibin Yu[3], Yanchang Wang[1] and Xiaojin Gong[1*]

Abstract

Many vision applications require high-accuracy dense disparity maps in real time. Due to the complexity of the matching process, most real-time stereo applications rely on local algorithms in the disparity computation. These local algorithms generally suffer from matching ambiguities as it is difficult to find appropriate support for each pixel. Recent research shows that algorithms using adaptive cost aggregation approach greatly improve the quality of disparity map. Unfortunately, although these improvements are excellent, they are obtained at the expense of high computational. This article presents a hardware implementation for speeding up these methods. With hardware friendly approximation, we demonstrate the feasibility of implementing this expensive computational task on hardware to achieve real-time performance. The entire stereo vision system, includes rectification, stereo matching, and disparity refinement, is realized using a single field programmable gate array. The highly parallelized pipeline structure makes system be capable to achieve 51 frames per second for 640 × 480 stereo images. Finally, the success of accuracy improvement is demonstrated on the Middlebury dataset, as well as tests on real scene.

Keywords: stereo vision, real-time, adaptive support weight, field programmable gate array (FPGA)

1 Introduction

Stereo vision has traditionally been and continues to be one of the most extensively investigated topics in computer vision. Since stereo can provide depth information, it has potential uses in many visual domains such as autonomous navigation, 3D reconstruction, object recognition, and surveillance systems. Especially, it is probably the most widely used for robot navigation in which accurate 3D information is crucial for the reliability of navigation. Compared with other range sensors such as laser scanner or time-of-flight, stereo vision is a technology that can deliver the sufficient description of the surrounding environment. Moreover, it is purely passive technology and thus offers low cost and high reliability solutions for many applications.

Stereo matching algorithms are computationally intensive for finding reliable matches and extracting dense map. As a result, sparse feature matching methods for stereo correspondence were widely used at first, due to their efficiency. A wide variety of approaches were proposed and greatly improved the accuracy of the stereo matching results over the last decade. However, real-time dense stereo is still difficult to be achieved with general purpose processors. For real-time requirements of most applications, the specific algorithms were often implemented using dedicated hardware, like digital signal processors (DSPs), graphics processing units (GPUs), application specific integrated circuits (ASICs), and field programmable gate arrays (FPGAs). In the last few years, the GPUs have become more and more popular. Using GPUs for stereo acceleration can directly be a solution for PC-oriented applications. However, the high power consumption limits their applications. FPGAs have already shown their high performance capacity for image processing tasks especially for embedded systems. An FPGA consists of an array of programmable logic blocks and represents an efficient solution to do parallel processing. Comparing with ASICs, FPGAs are re-programmable and have a relatively short design cycle. This makes the FPGAs offer great flexibility in manipulating the algorithm.

A lot of work has been carried out on hardware implementation of stereo algorithms. However, they differ considerably in their basic design principles.

* Correspondence: gongxj@zju.edu.cng
[1]Department of Information Science and Electronic Engineering, Zhejiang University, Yuquan Campus, Hangzhou, 310027, PR China
Full list of author information is available at the end of the article

Integrating specific algorithm in an embedded system is a delicate task, as it faces the factors of limited resources and different scales. At present, few high-performance implementations of stereo vision algorithms exist. The key challenge in realizing a reliable embedded real-time stereo vision system is keeping the balance of execution time and the quality of the matching results.

To solve these issues, a real-time stereo vision system is designed in this article to produce depth information both fast and accurate. All the modules are completely implemented inside a single chip of state-of-art FPGA. Our study is motivated by the following observation that: with carefully selecting the support region, the adaptive support weight (AW) approach [1] leads to a dramatic improvement in performance of the matching quality. Unfortunately, this advantage does not come for free. Since for each pixel many weight factors have to be computed, it is obtained at the expense of high computational requirements. This is very crucial, since it cancels out the biggest advantage over other complicated methods, the fast computation time. Being aware of hardware features, we propose a stereo system on FPGA with adaptive support weight approach. Currently, the only solution incorporating AW algorithm was proposed by Chang et al. [2] recently. Their proposed architecture was only evaluated by standard ASIC cell library. According to the implementation of a system, improvements are still needed in regard to the limited resources consumption, frame rate and integration of pre- and post-processing. Our contribution goes one step further, the entire stereo vision process, includes rectification, stereo matching, and post-processing, is designed using a single FPGA. The AW algorithm consists of multiplication and division operations which are difficult to implement on FPGA. As a consequence, original algorithm is modified in an essential way to eliminate the computational bottleneck and makes it hardware friendly. This design concept is realized as highly parallelized pipeline structure with good resource utilization. The implemented system can generate disparity images of 640 × 480 resolution at a frame-rate of 51 fps.

There are two main contributions in this article. First, the design of a complete stereo vision system is a combination of process including rectification, stereo matching, and post-processing. We keep the high data parallel structure in algorithm design, such that the architecture can efficiently exploit the capabilities of an FPGA with pipeline and process parallelism. Second, the AW approach is modified by introducing hardware-friendly approximation. The total design concept aims at balancing the functional performance, the computational burden, and the usage of the device resources. To the best of the authors' knowledge, it is the first solution to implement AW algorithm in a complete FPGA-based system.

The remainder of this article is organized as follows: Section 2 introduces background of stereo vision and related works on real-time implementations, and Section 3 describes the adaptive support weights cost aggregation method with corresponding hardware-friendly approximation. Then, we present our system design as well as detailed description about the implementation in Section 4. Section 5 evaluates the experimental results. In the end, Section 6 presents final conclusions.

2 Background and related works

A stereo matching algorithm tries to solve the correspondence problem for projected scene points and results in disparity map. Scharstein and Szeliski [3] have provided an excellent survey of stereo algorithms and a taxonomy was given based on matching cost, aggregation, and optimization. In general, two main categories can be distinguished: local algorithms and global algorithms. The local algorithms estimate the disparity value at a given point only based on the intensity values within a support region around the point. Global algorithms, such as dynamic programming [4,5], graph cuts [6], and belief propagation [7], make explicit smoothness assumptions of the disparity map to improve estimated results. Typically, these algorithms solve for the disparity map by minimizing a pre-defined global cost function. The smoothness constraint improves the stereo quality in textureless areas and pixel-based photo-consistency term reduces the blurring effect in object border regions. Recently, another kind of method called semi-global matching (SGM) was proposed [8]. It performs an optimization through multiple paths across the entire image to approximate the global optimization.

Most top-ranked dense stereo algorithms rely on global optimization methods. However, the algorithms usually did not depend on the specific global method alone. As indicated by Li and Zuker [9], the smoothness priors used implicitly encourage the frontal parallel plane which will poorly capture the real scene by improperly splitting a slanted or curved surface. As a result, the most successful variants related class of "segment-based" methods [10,11], and the optimization is usually complex and extremely computation intensive. To include processing time in the evaluation, only few algorithms declared capable of near real-time. And the original algorithm is constructed via a iterative and sequential procedure which is difficult to utilize hardware parallelism. In order to achieve real-time, recent advances exploit the parallel computational power within GPUs [12,13]. With some necessary modifications, some global methods also implemented in the very large scale integration circuit [14,15] at the expense of considerable high consumption of logic source, memory and bandwidth.

Take account of the balance between the resource consumption and performance, we still focus on the local algorithms. Local algorithms can be efficient but they are sensitive to local ambiguities and noise. Early research mostly studied the work of different similarity measure or used the combination of different pre and post-processing methods to improve the matching quality. Here we do it another way: our research is concern with the improvement on the essence of the algorithm. More specifically, we improve the way to aggregate support during correlation. Local algorithms rely on support windows. The major challenge in local algorithms is to find a well-suited size for the typically square support window. Large support windows give sufficient intensity variation to reduce ambiguities, but result blurred object borders and lose of detail. Small support windows reduce the problem, but increase the influence of local ambiguities, which leads to a decrease of correct matches. To handle these areas, the variable window [16] algorithm was proposed at first. Later, Bobick et al. [17] and Fusiello et al. [18] proposed shiftable-window approaches which consider multiple windows located at different positions and select the one with smallest cost. Hirschmüller et al. [19] proposed another extension which picks several sub-windows from multiple windows configuration.

The methods mentioned above are still restricted to limited sizes and shapes. Understanding that, several adaptive-window approaches [20,21] have been proposed, which could model non-rectangular support windows. Instead of finding an optimal support window with arbitrary shape and size, Yoon et al. [1] proposed the adaptive support weight algorithm that adjusts the support-weight of each pixel in a given support window. Tombari et al. [22] and Gong et al. [23] have evaluated many cost aggregation methods in a Winner-Takes-All framework, considering both matching accuracy and execution speed. In their evaluation results, adaptive support weight (AW) approach is the leading method in terms of accuracy. It is even comparable to many global optimization based algorithms. However, the aggregation process becomes computationally expensive. As reported in [1], it took about one minute to produce a small depth map.

2.1 Related studies

Due to the computational complexity of stereo algorithm, a number of attempts have been made to realize real-time performance. The works in [24,25] tried to implement real-time stereo matching on general purpose process, however the limited computing power restricts the frame rates. The DSPs have more computation power than general purpose processors. An early system was introduced by Kanade et al. [26], a hybrid system based on a custom hardware with C40 DSP array. Later, Konolige [27] introduced the SRI stereo vision system performing rectification and area correlation. It is one of the most famous DSP solutions. However, the DSPs still have the limitation of their sequential operation. Another powerful solution is the use of GPUs. The GPU is a massively parallel computing device. Using GPUs for stereo matching was first investigated by Yang and Pollefeys [28]. They used the sum-of-square difference (SSD) dissimilarity measures for windows of different sizes. With the advantage of compatibility and flexibility, the GPUs solutions could implement complex global optimizing algorithms [29-31]. But, GPUs are generally too expensive in power consumption for embedded applications.

Regarding the hardware, FPGAs remains the most popular choice because of its inherent parallelism architecture and high computational power. The original study was begun by Woodfill and Von Herzen [32], the PARTS reconfigurable computer which consists of 4 × 4 mesh connected FPGAs was used to implement stereo matching. Keeping this research line, a number of real-time stereo systems were presented in the literature in recent years.

For the Sum of Absolute Differences (SAD) algorithm, Miyajima and Maruyama [33] proposed a stereo vision system which connects the personal computers using PCI cards. The system can process images with a size of 640 × 480 at a frame rate of 20 fps. The processing time is closely related to the window size, as it was mentioned, the performance becomes worse as the window size gets larger. Perri et al. [34] proposed an stereo matching circuit processing 512 × 512 images using a disparity range of 255 pixels. This study shows the advantage in terms of large disparity range, but the 5 × 5 of window size is not considered as sufficiently enough for correlation. MingXiang and Yunde [35] proposed a stereo vision system on programmable chip. It performs 320 × 240 pixels dense disparity mapping in 32 disparity levels, achieving video rate. Recently, Calderon et al. [36] presented a solution with two step processing algorithm. The hardware accelerator works within five pipeline stages could achieve 142 fps for CIF format image, at a frequency of 174.5 MHz.

Another popular algorithm for hardware implementation is census-based matching method. Woodfill et al. [37] described an ASIC design called DeepSea which enables the processing of 512 × 480 at a disparity range of 52 pixels, a block size of 7 × 7, and a high frame rate of 200 fps. The technique for implementing a flexible block size, disparity range and frame rate was proposed in [38]. The impact of using different similarity measures as SAD, rank, and census transform was also presented. Another census-based approach has been introduced by Murphy et al. [39]. Here, Xilinx Spartan-3 FPGA was used and a frame rate of 150 fps could be achieved for 320 × 240 pixel images.

The phase-based computational model provided another alternative to correlation methods [40,41]. Diaz et al. [42] developed a phase-based stereo vision design which generates about 20 stereo disparities using 1280 × 960 pixel images. As the number of phase correlation units directly related to the disparity range, the resource limitation on the FPGA limits the range of disparity. In order to handle this problem, Masrani and MacLean [43] took the advantage of the temporal information to extend disparity range without large resource consumption.

In recent years, Ambrosch and Kubinger [44] proposed a stereo matching implementation that extends the Census Transform to gradient image and prepared to offer as an IP core for embedded real-time system. Another recent implementations was proposed by Jin et al. [45], who designed a stereo matching system based on a Xilinx Virtex-4 FPGA, processing 640 × 480 images with block size 15 × 15 and a disparity range of 64 pixels. Although all these implementations mentioned above exhibit good real-time behavior, these two studies also present a complete discussion of the accuracy of the algorithm on the Middlebury stereo datasets.

3 Adaptive support weight approach

Local algorithms reduce ambiguity by aggregating matching costs over a correlation window. The correlation window also refers to local support region implicitly implies that the depth is equal for all pixels inside. And this intrinsic assumption will lead to numerous errors especially at the region of depth discontinuities. When doing cost aggregation, the support from a neighboring pixel is valid only if such pixel has same disparity. The way to select appropriate support is a key factor of the correlation method. For this purpose, adaptive support weight (AW) algorithm was proposed to perform aggregation on the appropriate support. The idea of adaptive support weight approach originated from a edge-preserving image smoothing technique called bilateral filtering [46]. It combines gray levels or colors based on both their geometric closeness and their photometric similarity, and prefers near values to distant values in both domain and range. This idea was extended to the cost aggregation in stereo matching. The similarity and proximity are two main visual cues which can help us to pre-estimate the support weight of every pixel. The mechanism relies on the assumption that neighboring pixels with similar color and closer range to the central pixel p are likely from the same object. This is the main idea behind adaptive support weight generation. Based on these gestalt principles, the support weight of a pixel can be written as

$$w(p,q) = f_c(\Delta c_{p,q}) \cdot f_g(\Delta g_{p,q}) \tag{1}$$

where $\Delta c_{p,\ q}$ represents the color difference and $\Delta g_{p,q}$ represents the spatial distance between the pixel p and q. The f_c and f_g represent the two weighting functions, respectively, assign weight value by color similarity and geometric proximity.

Typically, the weights are assigned by Gaussian function. Assuming the p is the pixel under consideration and q is a certain point in its neighbor $N(p)$ (support region centered in p). The function $f()$ implements this by

$$f(p,q) = \exp\left(-\frac{\Delta_{pq}}{\gamma}\right) \tag{2}$$

with choice of gaussian variance value γ tunes up the strength of the resulting weights.

Once the support weights are know, they are exploited to aggregate pixel dissimilarities within the support window. In AW approach, the final matching cost between two corresponding pixels is measured by normalized weighted costs. The original algorithm takes into account both target and reference image to calculate support weights. The matching cost between pixel p_t and p_r, $C(p_{t,}, p_r)$, can be expressed as

$$C(p_{t,}, p_r) = \frac{\sum_{q_t \in N(p_t),\, q_r \in N(p_r)} w\left(p_t, q_t\right) w\left(p_r, q_r\right) e\left(q_t, q_r\right)}{\sum_{q_t \in N(p_t),\, q_r \in N(p_r)} w\left(p_t, q_t\right) w\left(p_r, q_r\right)} \tag{3}$$

The function $e(q_t, q_r)$ computes the pixel dissimilarity between a pixel p_r of the reference image and a pixel p_t of the target image. Here, we chosen truncated absolute differences (TAD) to implement $f()$, the values exceeding *const* are truncated.

$$e\left(q_t, q_r\right) = \min\{|I_t\left(q_t\right) - I_r\left(q_r\right))\,|, const\} \tag{4}$$

This mechanism reduces the influence of occluded pixels.

3.1 Hardware-friendly approximation

In spite of the dramatic improvement of accuracy brought by adaptive support weight approach, it pays the cost of high computational complexity which makes it not only time-consuming but also resource-intensive. While designing with FPGAs is faster than designing ASICs, it suffers from the problem of fixed resources. Thus, approximations must be introduced to provide trade-off between best overall performance and resources consumption.

As it is well known that multiplication and division operations on FPGAs are computationally expensive in terms of the number of logic elements required. Although the advanced FPGAs typically support tens of digital signal processing elements to form wide variety of math functions, it is far below the required amount in many cases. Especially if the operation is being replicated many times to take the advantage of parallel mechanism. Therefore, careful design is needed to avoid these operations. The consumption of on-chip memory is another fact to be considered. In implementation of stereo vision, a fully paralleled design needs to estimate the matching cost of all the disparity candidates simultaneously. Suppose we use $n \times n$ support region size with disparity range of d, Table 1 illustrates the number of arithmetic operations required to calculate cost aggregation value for one pixel. Obviously, directly implementation of original algorithm is not realistic considering the resource constraint. As a result, we have modified the original AW algorithm to make it more hardware friendly.

We use the gray-scale image instead of CIELab color space originally adopted. This is not only due to benefit of two-thirds reduction in temporary memory, but also the compatibility with most existing stereo cameras. In addition to this, three simplifications are proposed to reduce the computational complexity. First, in order to get rid of division operation, the weight generation only refers to the target image. Hence, we can eliminate the normalization operation as different disparity hypotheses of every pixel share the same sum of weights. As a result, the aggregated costs were simplified as,

$$C(p_t, p_r) = \sum_{q_t \in N(p_t)} w(p_t, q_t)\, e(q_t, q_r) \tag{5}$$

The second simplification is to approximate the cost aggregation by means of dimensional separation. A one-dimensional cost aggregation is applied to the vertical direction at first and the intermediate result is aggregated again in horizontal direction. This two-pass approximation brings significant savings in computation. The complexity per disparity of the separable implementation is just $O(2n)$ compared to $O(n^2)$ for a whole support region aggregation, where n is the size of support window. Based on above simplifications, the aggregated costs in our approach can mathematically be represented as follows:

$$\begin{aligned} C_{\text{vet}}(p_t, p_r) &= \sum_{q_t \in N_{\text{vet}}(p_t),\, q_r \in N_{\text{vet}}(p_t)} w(p_t, q_t)\, e(q_t, q_r) \\ C(p_t, p_r) &= \sum_{q_t \in N_{\text{hor}}(p_t),\, q_r \in N_{\text{hor}}(p_t)} w(p_t, q_t)\, C_{\text{vet}}(q_t, q_r) \end{aligned} \tag{6}$$

where N_{vet} is a notion of neighborhood in vertical direction and N_{hor} represents neighborhood in horizontal direction, respectively.

Third, we use shift operation instead of multiplication through the truncated approximation of the weight function. One essential step of hardware implementation is to convert the float-point to fixed-point, the weight value of exponential function is first scaled up and round to fixed word length in binary mode. As mentioned above, multiplication in FPGAs would be an expensive operation. The tens of multiplier elements are not able to meet the requirements for parallel computing (number of multipliers required is proportional to the product of support size and disparity range). To solve this problem, we truncate the fixed-point number that leaves only one non-zero most significant bit (MSB). With only one non-zero bit in weight, the multiplication between the weight and dissimilarity cost can be achieved simply by the shift operation. The final quantized coefficients depends on the word length N of the input data, the word length W of the coefficient and the parameter γ. Based on our experimental data, the weight coefficients are quantized in 9-bit word length which means the weight is quantized to the 9 quantization level. The comparison of color similarity weight function is shown in Figure 1 for N = 8 bit and γ_c = 14.0. The geometric proximity weight function is truncated and scaled in the same way, shown in Figure 2. The analysis of quantization level is given in detail in the following section.

3.2 Evaluation

In this section, the modified algorithm we present was evaluated. After those hardware-friendly approximation, the disparity maps we generated were not as good as those reported in the original article [1]. To find the good trade-off, the terms of matching quality and

Table 1 Number of operations required of AW cost aggregation for every pixel, when using $n \times n$ support region size and d disparity range

	Multiplication	Division	Addition	Shift
Original algorithm	$3(n^2 \times d)$	d	$2(n^2 - 1)$	0
Omit weights of reference image	$2(n^2 \times d)$	0	$n^2 - 1$	0
+Dimensional separation	$4(n \times d)$	0	$2(n- 1)$	0
+Truncation of weight value	0	0	$2(n- 1)$	$4(n \times d)$

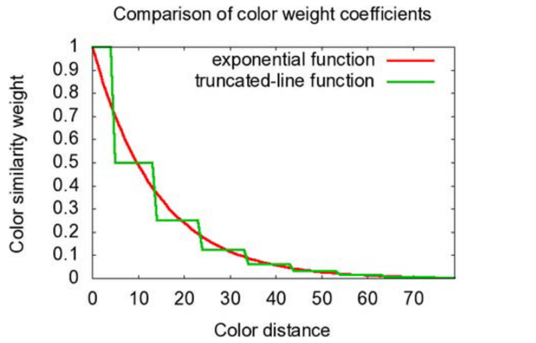

Figure 1 A comparison of weight function based on color similarity.

hardware efficiency were mainly considered. As reference for the matching quality, four stereo image sets from the Middlebury benchmark datasets are used. These are the Tsukuba, venus, Teddy, and Cones datasets. As evaluation criterion for the matching quality, we use the average error rate which is average percent of bad pixels of all four benchmark datasets.

First, we evaluate the influence of using gray-scale intensity instead of original CIELab color representation. As mentioned above, the original adaptive support weight algorithm came from the idea of bilateral filters. The choice of CIELab color space in bilateral filters mainly because it strongly correlate with human color discrimination and has perceptually meaningful measure of color similarity [46]. Thus, in a sense, it is an open issue if the CIELab color space still has such benefits in stereo matching. Therefore, we evaluate two color spaces which are CIELab and YUV as well as gray-scale. Figure 3 shows the average error rate of matching results using different photometric information. It is observed that the accuracy difference between using CIELab and YUV color space is insignificant. On the other hand, the gray-scale color reflects a lower accuracy as missing color information in photometric cue. Although using color space seems to be more accurate, the input of gray-scale images makes our system be suitable for standard HDR, infrared cameras and most existing stereo cameras.

Next, the influence of the separable aggregation strategy is evaluated in the same way. Original cost aggregation strategy is computationally expensive because the adaptive kernel has to be recomputed at every pixel. The separable implementation greatly reduces the redundant and only needs a fraction of computation. Effectiveness of this simplification was verified in bilateral filter [47] as it is not only fast but it also approximates the true bilateral filter reasonably well. Figure 4, comparing the accuracy curves of different aggregation strategies, shows that separable aggregation closely follows the original implementation in most cases. This is mainly due to the fact that the separable aggregation operates along the sampling axes, so the proposed method approximates the original aggregation strategy well for image patches whose dominant orientation aligns with the sampling grid.

Lastly, we evaluate the proposed scale-and-truncate method. Figure 5 shows the matching accuracy for varying number of the quantization level. It is observed that the accuracy improves with the number of the quantization level, but improvement gets very marginal after the quantization level exceeds 9. This is the main reason for choosing word length of coefficients.

After applying the proposed approximation, we modified the original algorithm in a hardware-friendly way and made it possible to exploit data parallelism at implementation below. Figure 6 shows two weight masks for different pixels in the Tsukuba image. In most cases, our approximation mask is very similar to the original mask. In all masks, pixels assigned high weights close to the center in terms of both photometric and geometric distance. It results disparity maps of similar quality.

To illustrate the effects of the support region size, Figure 7 plots the overall error rate of disparity maps

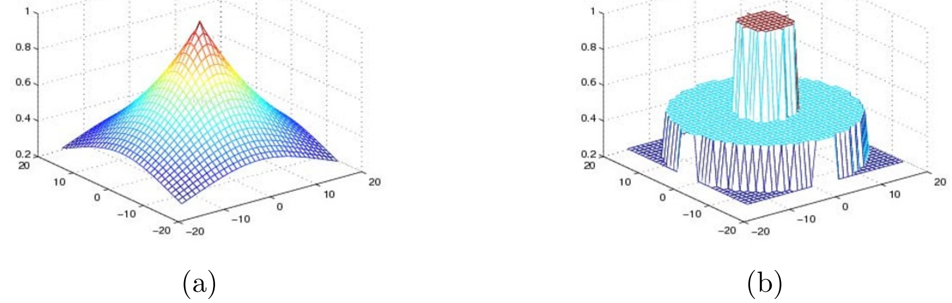

Figure 2 A comparison of weight function based on geometric proximity. **(a)** Geometric proximity weight assigned by Gaussian function. **(b)** Scaled and truncated weight function.

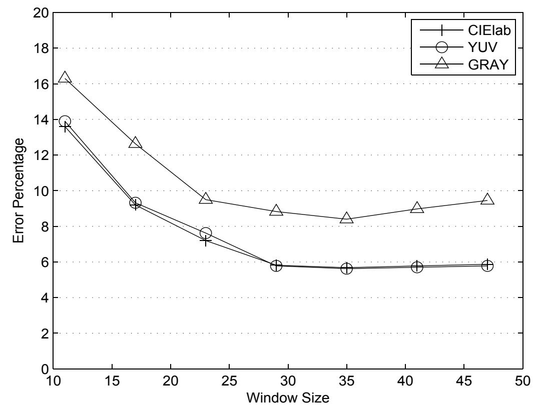

Figure 3 The disparity error rate of using different color spaces.

generated by our modified AW algorithm. The following can be observed that, in general, the error rate decrease as the regions get larger. This is reasonable as small support regions cannot be effectively distinguished. More importantly, the matching quality seems stable after increasing the support window size over a threshold. It means the parameter of region size is less sensitive for changes in real scenarios. This is mainly because the outliers in the larger support region may be excluded in the cost aggregation or may have very small weights. The necessary of large support region again confirms the requirement of algorithm approximation. In the following sections, we will show that our experimental results still have good performance in accuracy.

4 System design

In this section, we first describe the overview of our system briefly. Then, we discuss the implementation of each processing module in details.

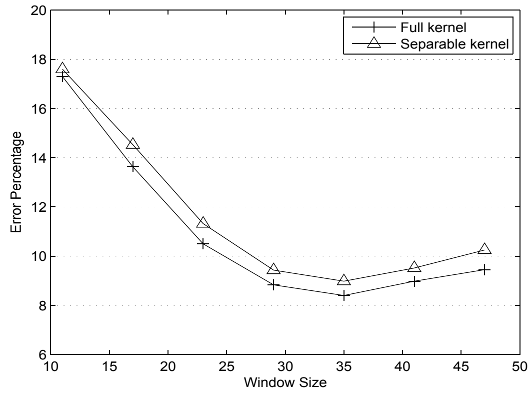

Figure 4 Performance comparisons between using different aggregation strategies.

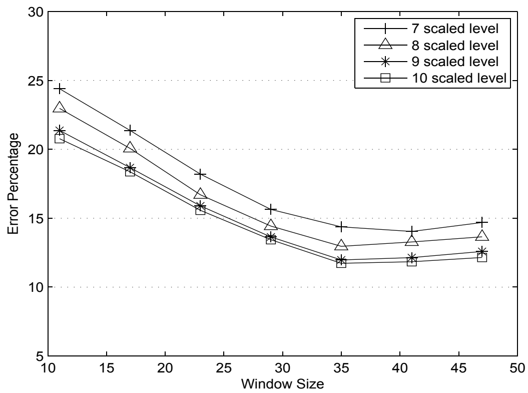

Figure 5 The error rates of disparity maps generated by different quantization levels.

4.1 System overview

The system is implemented on our new-designed FPGA-center platform (see Figure 8). The processing core of the system is Virtex-5 XC5VLX155T with 24,320 slices. The system interfaces two stereo cameras through high-speed LVDS links which can deliver the full 1.28 Gbps payload in each direction. The platform contains 128 MB of Nand flash memory used to maintain the rectification data. After system startup, the control module loads those data into Dram memory in order to maximize data throughput. The FPGA communicates with a host PC through one gigabit Ethernet. This port is used for disparity output and receiving instructions from high-level control module.

4.2 Hardware implementation

To achieve real-time performance, we designed a dedicated hardware architecture implemented in an FPGA. The architecture of the stereo-vision system is illustrated in Figure 9 which is mainly divided into three modules as follows: rectification, stereo matching, and post-processing. With the aid of the presented design concept the stereo process based on AW algorithm can be realized as highly parallelized pipeline structure with good resource utilization. All of the processing modules and dataflow are controlled by a controller module.

4.2.1 Rectification module

Image rectification is an important component of stereo vision process. It makes epipolar lines aligned with coordinate axis and parallel [48]. The pixels corresponding to same 3D point will lie on the same horizontal scanline of the rectified image pair and differ only in horizontal displacement which is called *disparity* in usual. It benefits real-time stereo matching by reducing the correspondence problem from the 2D search to just 1D search. The image rectification is kind of 2D projective transform, or *homography*, to each image. The

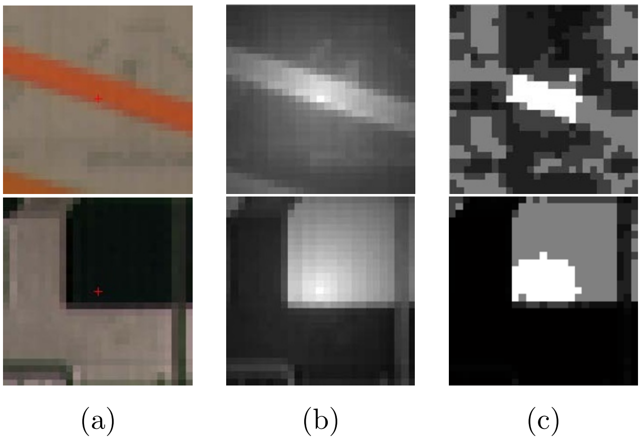

Figure 6 Support weight for selected windows of the Tsukuba image. The pixels marked by a plus sign are the pixels under consideration. The brighter intensity represents larger support weights. **(a)** Image crops. **(b)** Support weights computed by the original adaptive weight algorithm. **(c)** Our modified support weights.

homography H provides a planar mapping $m' \sim Hm$ between points $m = (u, v)$ in original image and points $m' = (u', v')$ in rectified images. So, the rectification can be implemented using a simple memory look-up tables that indicates the coordinates of source pixel for each rectified pixel. The camera calibration is done off-line to get the mapping relationship $(u', v') \leftrightarrow (u, v)$. A 2 × 2 pixel neighborhoods around the mapped pixel are used to perform bilinear interpolation and compute the pixel values at the fractional mapping coordinate.

$$I'\left(u', v'\right) = \sum_{i=1}^{4} a_i \cdot I\left(u_i, v_i\right) \tag{7}$$

I' and I represent rectified and original images, respectively. For each rectified pixel, we need to round the reverse mapping coordinate (u, v) to nearest-neighbor integers (u_i, v_i) and a_i are the corresponding weights used for interpolation. Figure 10 illustrates the storage format of rectification parameters. In order to synchronize the rectification for both left and right images, the parameters of these two images are mixed to form memory data. We reserve 20-bit width to represent top-left integer coordinate and all the four 10-bit weights a_i are encoded as rectification parameters for every pixel. The other neighboring coordinates are automatically calculated by offset.

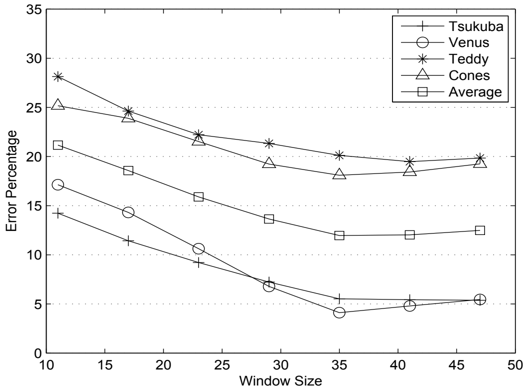

Figure 7 The error rate of disparity maps generated under different support region size.

Figure 11 shows the rectification procedure of left-image pixel. The dataflow is designed in a streaming fashion without any reliance on external memory. After initial latency, the first parameter data is read from the dram port and encoded to top-left address of reference image. In the next step, we can get the reference image data and its corresponding weight from second parameter data simultaneously. These two data are multiplied and then accumulated by DSP48E slices. During the process, the pixel coordinate of source image is incremental changed by offset. After four clock cycles the DSP48E outputs the rectified image data to the stereo matching module.

4.2.2 Stereo matching module

The stereo matching module is the core element of the whole system. It can be roughly divided into three stages: rectified image data input, adaptive support weighted cost aggregation and disparity search, seen in Figure 12. The first stage is the input stage that buffers the image data from rectification module. The buffers ensure that there are sufficient image data for the correlation support. Here, each image line is stored in a separate memory block. These memory blocks are serial-connected that the read-out data are written back to the next line buffers. The main advantage of this configuration is that it enables the data of the different lines to

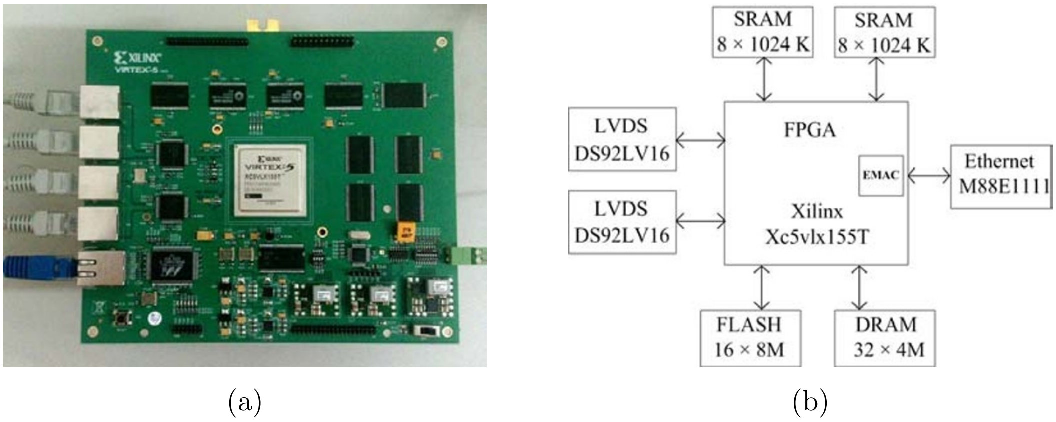

Figure 8 The new-designed FPGA-center platform. **(a)** Physical prototype of our FPGA-center platform. On the left above, two pairs of LVDS link for image input. At the bottom, gigabit Ethernet for communication with host PC. **(b)** Block diagram of the system components.

be read out in parallel. We use a 35 × 35 size of support region, thus there are 35 scan line buffers for both left image and right image. The output of different scan line buffers convert pixels into a column vector of the support window.

As we mentioned above, the corresponding pixels in rectified images only differ in horizontal displacement. Here we use the shift registers to make the correlation under different disparity candidates. The disparity search range is currently set to 0-59 pixels. The column vector of image pixels as a whole unit walks through the delay network one after another. Since the delay network consists of shift registers, it guarantees the correlated data under different disparity hypotheses be accessed in parallel. Multiple cost aggregation modules are used for calculating the final cost of every disparity candidate. This mechanism gives us the opportunity to estimate disparity synchronized with column data input.

The significant differences between AW and other aggregation methods is that the sliding window cannot be applied, as the adaptive weight has to be recomputed at every pixel. Therefore, cost aggregation has to be performed in an exhaustive manner and the computational requirements are directly depending on the size of matching window. This is particularly bad, as we found the AW approach usually require larger window to get good quality result. Thus, we had to reduce the computation's complexity and make it again suitable for real-time implementations. Figure 12 shows block diagrams of our aggregation strategy. Instead of directly aggregates cost of whole support size $n \times n$, we use a two pass approach that first pass aggregates cost along vertical direction, followed by a pass aggregating cost along horizontal direction. This reduces the arithmetic complexity form $O(n^2)$ to $O(2n)$. It is also worth observing that the weight term obtained only using the target

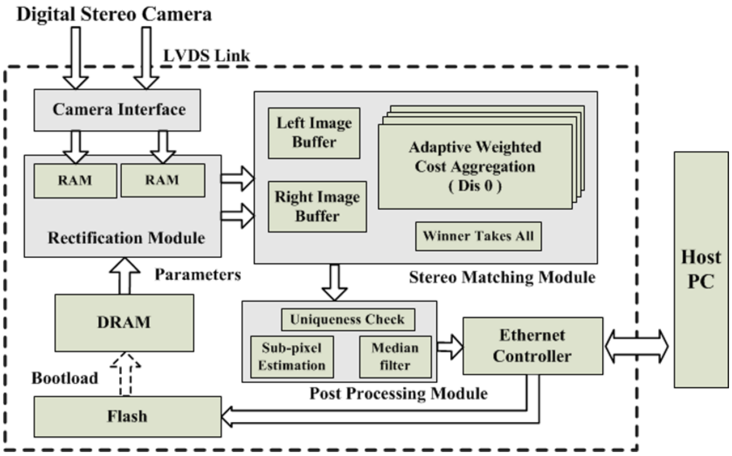

Figure 9 High-level hardware architecture of the proposed stereo vision system.

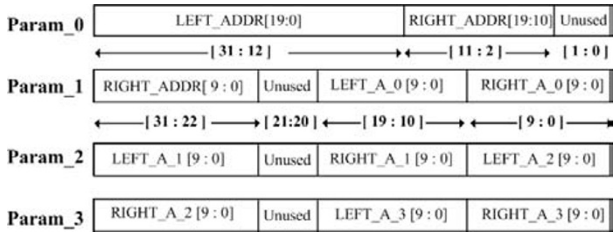

Figure 10 Storage format of the rectification parameters.

image in our weight generation strategy and the correlated blocks shares the same weights under the different disparity hypotheses. Consequently, we omit the weights accumulation operation as well as the normalization operation module. The weight of every pixel is a combination of the similarity and proximity measurements. Thus those two kinds of weights component have to be calculated in each aggregation pass.

Figure 13 shows the implementation of vertical aggregation. The COMP unit measures the dissimilarity between two pixels, which we use TAD as the cost function. The subtract unit (SUB) calculates the absolute intensity differences between the center and the neighbor pixel. The parallel input of the same column data allows the computation between the vertical center pixel and the remaining pixels at the same time. Instead of calculating the exponential function which is expensive in hardware, a look up table (LUTs) is used to store the pre-calculated function values corresponding to the difference. For an 8-bit gray scale image the LUTs contains maximum of 256 values. The intensity difference itself is directly taken as address of the LUTs to output the corresponding similarity weight coefficient. The coefficients are stored in the LUTs in the initialization phase. During read out of the weight coefficient, the related raw cost value is registered for synchronicity. The quantized coefficients are scaled into multiple levels that only one non-zero MSB is kept for every coefficients.

It simplifies the weighted action through using the shift operation instead of multiplication. The intensity difference of center pixel is shifted by the highest coefficient C_0. In the next stage, the outputs of the shift modules are provided to the subsequent geometric proximity weighting module. Here we exploit the symmetry of the weighting function of the geometric component. The top and bottom pixels in the column share the same geometric coefficient P_0, so those two costs are summed at first then shifted as a group. The cost of pixel in the center of the column is not belong to any pair and is directly done with shift. Once the costs are proximity weighted, the weighted values are summed up by the adder tree to one row value. The intensities of vertical center pixels are also transmitted out for calculating the color similarity in horizontal aggregation.

After the vertical aggregation stage, the support window is reduced to one row and then we move forward to the horizontal aggregation stage. And little memory overhead is required since the output image can be written direct onto the input's memory. Figure 14 illustrates how the horizontal module works. There is little difference between those two stages. Two shift register arrays are used to buffer the input data of different columns so

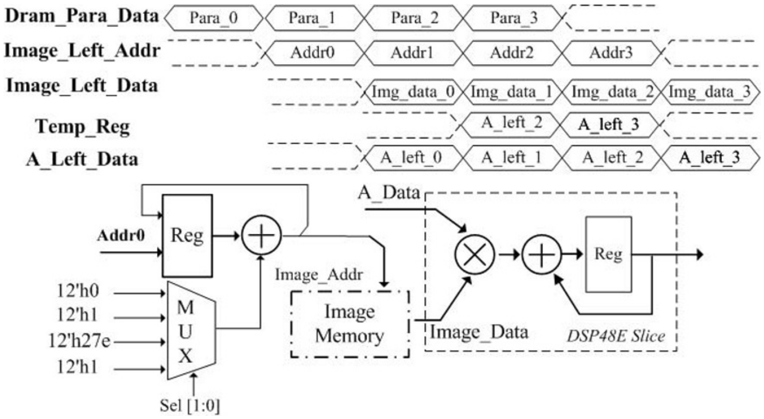

Figure 11 Flowchart of pixel rectification process.

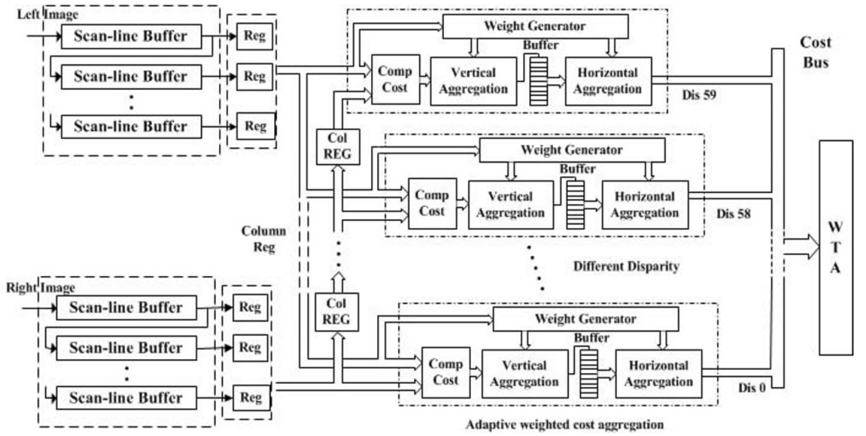

Figure 12 Hardware architecture of stereo matching module.

that the whole support window can be constructed. The remaining parts are similar to the processing described in the vertical aggregation module.

Once the horizontal aggregation finished, the costs of all possible matches are forwarded to the final stage. The disparity which produces the smallest matching cost $C_{(p)}$ is selected as the optical disparity d_p of each pixel p in the target view, which is so-called Winner-Takes-All (WTA) strategy:

$$d_p = \arg\min_{d \in D} C_d(p) \tag{8}$$

where D represents the set of all disparity hypotheses. The WTA algorithm is implemented in the parallel-cascaded form, which is shown in Figure 15.

4.2.3 Post processing module

The post processing module contains three sub-modules as follows: sub-pixel estimation, uniqueness check, and median filter. They can be divided into two categories: (a) the sub-pixel estimation improves the resolution of final results, (b) the other two sub-modules used to detect unreliable matches and improve the accuracy of the depth measurement.

Stereo matching module only reconstructs discrete layers of disparity. This stepwise disparity variation will result stair-step effects in reconstruction. In order to get the continuous depth estimation, the sub-pixel interpolation is used based on parabola fitting. Even if there exist more accurate techniques for the sub-pixel estimation [49], they are computationally too expensive for real-time stereo vision. The parabola fitting works over three points at discrete disparity candidate: d, d_-, and d_+, with their similarity measures. d is the selected disparity, d_- and d_+ are the closest disparity at one pixel deviation. Optimal disparity d^o is solved by finding the maximum of the quadratic function which we use the

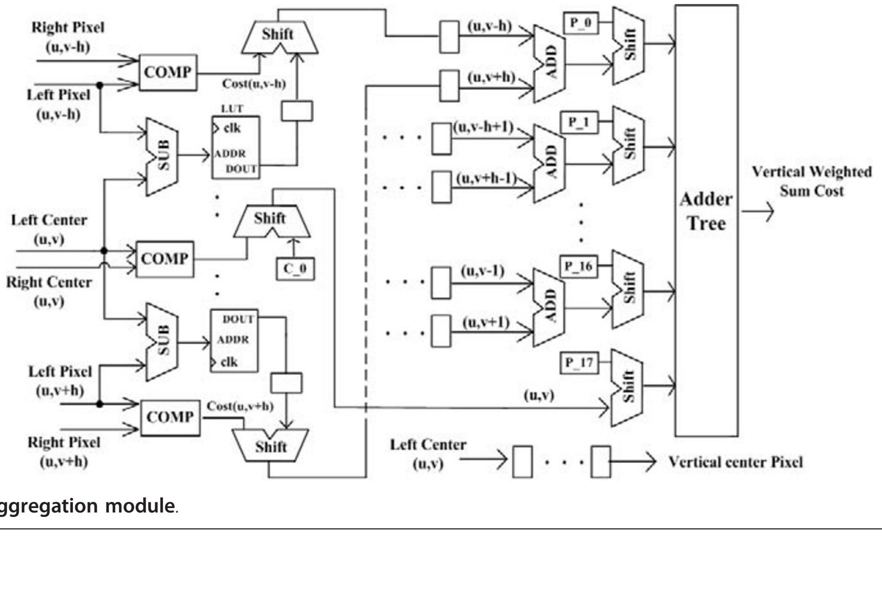

Figure 13 Vertical aggregation module.

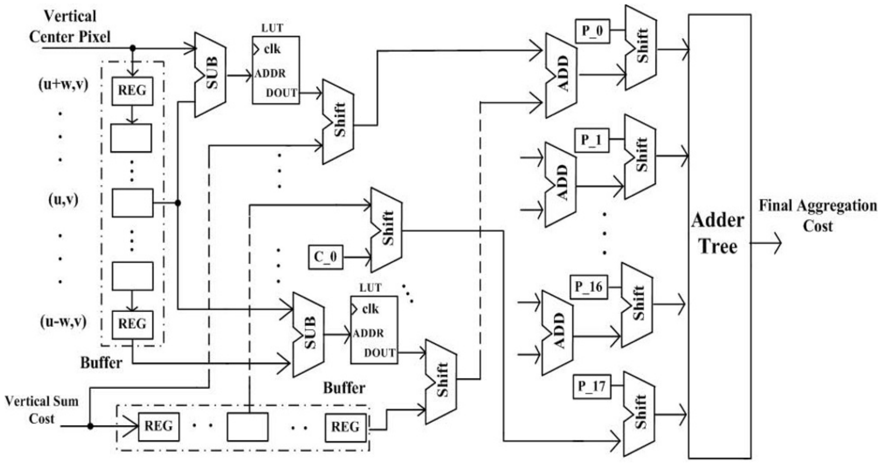

Figure 14 Horizontal aggregation module.

following relation, where $\rho(d)$ is the similarity value at disparity d.

$$d^o = d + \frac{\rho(d_-) - \rho(d_+)}{2\rho(d_-) - 4\rho(d) + 2\rho(d_+)} \tag{9}$$

The parabola fitting in Equation 9 contains a division, which is resource intensive when implemented in hardware. However, fractional part is usually limited to a value with a dedicated resolution of a few bits. As suggested by Ambrosch and Kubinger [44], the sub-pixel refinement can be computed by simply calculating the enumerator as well as the denominator, and comparing how much smaller the enumerator is compared to the denominator. Since the number of bits in the result is limited, it requires only a limited number of comparisons. Our approach uses 4 bits data widths to represent the fractional part which requires only six shift operations, seven additions/subtractions and four comparisons. Figure 16 shows the implementation of the sub-pixel estimation module. After eight cycles of clock latency, a calculated sub-pixel value is obtained and synchronized with the pixel clock. The sub-pixel refinement adds additional accuracy to the disparity result and extends the disparity resolution to 10 bits.

While doing the sub-pixel refinement, the system uses uniqueness check on the disparity map for the removal of occluded or mismatched areas. As far as local matching algorithms are concerned, the mutual consistency check is widely adopted to detect unreliable matches. For simple cost aggregation approaches, the mutual consistency check can be accomplished without doing the correlations more than once because of its symmetry properties. Unfortunately, this symmetry no longer holds in adaptive support weights aggregation as the weights vary between different templates. As a result, we used a uniqueness check method proposed by Di Stefano et al. [50]. Below, we give a brief description of this

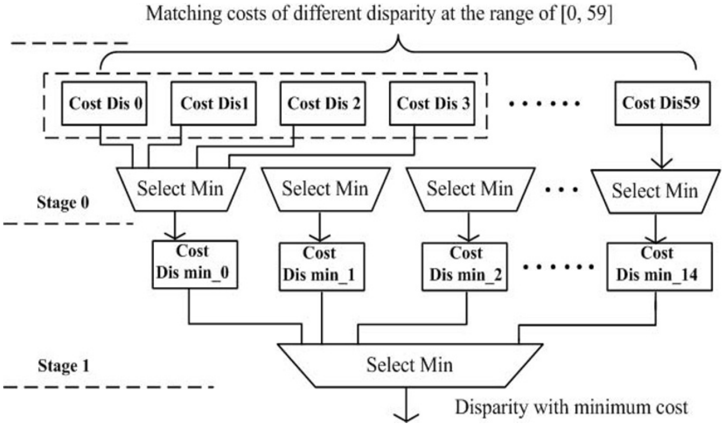

Figure 15 Architecture of search for the correct disparity.

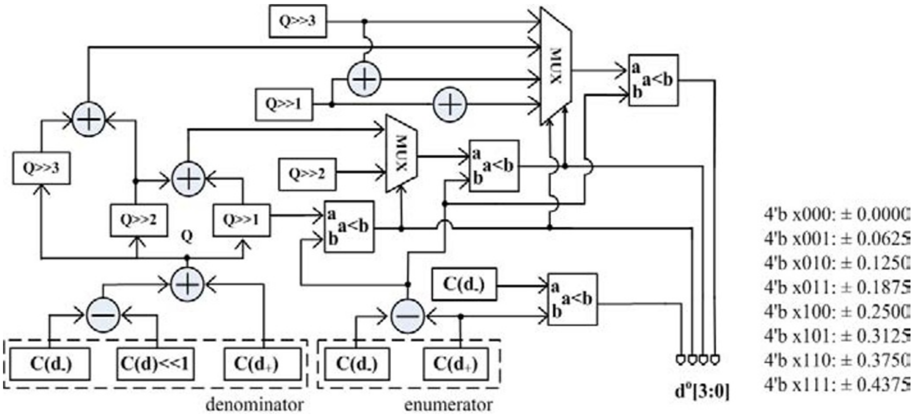

Figure 16 Realization of the sub-pixel estimation using parabola fitting method.

method. Assume that the left image is chosen as reference and the disparity candidates range form [0, ..., $d_{\max}$]. $L(x, y)$ is one point of left image, the algorithm searches for the best candidate by minimizing matching cost *C*. Suppose now the best match found for $L(x - \alpha + d_{\max}, y)$ is $R(x, y)$, with matching cost $C(x - \alpha + d_{\max}, x, y)$. And another point of the left image $L(x - \beta + d_{\max}, y)$ has previously been matched with $R(x, y)$ with cost $C(x - \beta + d_{\max}, x, y)$. Based on the uniqueness constraint, we conclude that at least one of the two matches is incorrect and only the match with minimum cost is retained. Thus, if the point $L(x - \alpha + d_{\max}, y)$ has less matching cost than $L(x - \beta +d_{\max}, y)$, uniqueness check will reject the previous match and accept the new one. This implies that the proposed approach allows for recovering from previous errors as long as better matches are found during the search. The procedure is shown in Figure 17. Each new arriving left point $L(x, y)$ belongs to the interval [$R(x - d_{\max}, y)$, ..., $R(x, y)$] of the potential matching points in the right image. During the Implementation stage, we only need to set up $d_{\max}$ registers to keep track of the best match and corresponding matching cost for right image points in the range of interval. The match newly created for $R(x, y)$ is compared with previous match, and the one be replaced will be labeled “incorrect”.

Finally, the median filtering is applied to the disparity data. The median operation applies a 3 × 3 filter on the disparity image. It can enhance the disparity results by cleaning up spurious isolated matches which almost are false ones. Median filtering is also implemented based on the window processing architecture and similar scan line buffers are used to construct filter window as stereo matching module. For such small kernel, the element in the middle position can be identified through pipeline sorting and swapping operations which only consumes limited system resources [51].

5 Experimental results and discussion

The application of our hardware-based stereo vision system is mainly focus on robot navigation. This task requires the system continuing providing accurate information about the environment. Thus, for the analysis of our system we evaluate the performance of the implementation both in accuracy and running speed.

The proposed real-time stereo vision system is designed and successfully implemented on a Virtex-5 LX155T FPGA. This device offers 24,320 slices and up to 7,632 kbit block RAMs. The implemented system interfaces two stereo camera and receives an image size of 640 × 480 pixels. In current architecture, we set 60 pixels as a maximal disparity measure with 35 × 35

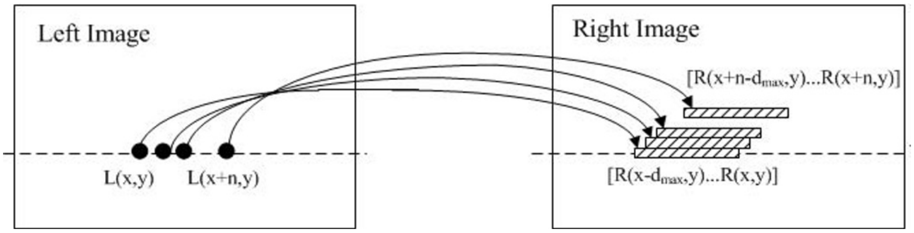

Figure 17 Matching procedure for the successive points along the scanline.

pixels windows size used in cost aggregation. As the sub-pixel estimation offers 4-bit additional resolution, it can produce a dense disparity map of size 640 × 480 pixels with 10-bit sub-pixel accuracy disparity result. With these parameters, our proposed architecture is synthesized by Xilinx ISE 10.1 development tool. Table 2 shows the resources required in order to implement our stereo vision algorithms. The number of logic blocks (slices) is given, along with necessary on-chip memory (block RAM).

The licensed Ethernet Media Access Controller (TEMAC) core is used to develop Ethernet communications. Two DSP48E slices are used to rectify left and right images, respectively. The results show that the logic resources consumption are dominated by the stereo matching module due to its high number of aggregations, while the rectification and post processing require slightly less logic. Since the correlation modules of different disparity hypothesis are processed in parallel, increasing the range of disparities will proportionally increases the necessary resources for stereo matching module. Applying block reusing techniques could optimize resource usage, but on the expense of processing speed. On the other hand, increasing image resolution has little effect on the resources consumption, since our architecture is based on the local pixel processing. The results also show that vast majority of the FPGA's block RAM is consumed by the scan line buffers in the stereo matching module.

Before to describe disparity image resulted of our system, we first analyze the real-time performance of hardware implementation. The distinguishing feature of our architecture is intensive use of parallelism and pipelining. As described above, each functional module is replicated and executed in parallel to achieve parallelism. For example, we place d numbers of the aggregation modules to get the correlation cost of different disparity candidate simultaneously. Moreover, the scan line buffer structure joint with two pass approach ensure that the matching module updates support window and calculates corresponding cost value in every clock cycle. In order to maximize allowed frequency, some complex operations like multiple addition and WTA are divided into a set of pipeline stages. Figure 18 illustrates the scheduling of disparity estimation. First, 35/2 - 1 = 17 rows rectified image data are needed to initialize scan line buffers. The rectification module generates rectified image data once every four clock cycles as it requires fetching four pixel data to do bilinear interpolation. The initialization signal (INIT) holds low until the end of buffer initialization. When the INIT goes high, it indicates that the following module should initialize its processing state. All the following stages in our design are synchronized with the rectified data generation rate. The components of the stereo matching module are timed by data enable signal (DAE) indicates that there is a new rectified data input. After pipeline latency, the disparity candidates with minimum cost are generated by WTA module and synchronized with DAE signal. In order to do the uniqueness check, the 60 most recent disparity data are collected in the FIFO memory. At the same time, the final disparity values are calculated through the sub-pixel estimation and median filter modules. The output data enable signal (DOE) indicates that there is final disparity data on the DATA bus. The clock frequency for the proposed system is 60 MHz. The highly parallelized pipeline structure enables one disparity data output of every four clock cycles. A pair of 640 × 480 images are processed at 15 Mpixels/s, which is equivalent to 51 fps. In the metric of points times disparity per second (PDS), this system achieves a performance of 940×10^6 PDS, which is suitable for demanding applications. We also compare the proposed system to other stereo vision methods reported. As shown in Table 3, nearly all the stereo vision systems use fixed support region during the correlation. Taking advantage of incremental calculation schemes during aggregation, the systems like can exploit fully the parallelism and achieve very high speed. While the PDS metric reflects the density and the speed of the system, the accuracy of the implemented algorithm is another factor to be considered. Although our method is slower than the state-of-art fixed window methods, an important feature of our system is its high-performance AW algorithm and integration due to our efficient modification.

Table 2 Device utilization summary

	Used	Available	Utilization (%)
Utilization summary			
Number of Slice Registers	35,020	97,280	36
Number of Slice LUTs	50,585	97,280	52
Number of DSP48Es	2	128	1
Number of TEMACs	1	2	50
Module level utilization			
Number of occupied Slices	16,697		69
rectification	201		1
stereo matching	14,835	24,320	61
post processing	1,169		5
Number of BRAM/FIFO	39		18
stereo matching	37	212	17
post processing	2		1

In order to evaluate the disparity quality of our approach, the Middlebury stereo evaluation is used. This evaluation platform provides a huge number of datasets consisting of the stereo image pair and the appropriate ground truth image. Each image is divided into different

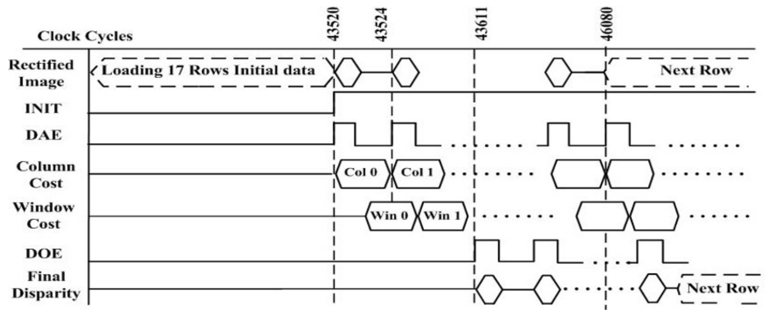

Figure 18 Time diagram for the disparity estimation.

regions, namely, non-occlusion regions and discontinuity regions. We measured the percentage of bad pixels on the four images of the dataset (Tsukuba, Venus, Teddy, and Cones). Since the system is built for real-time processing of an incoming image, the disparity results for evaluation were generated through functional simulation. Evaluation results are shown in Figure 19. Each row of Figure 19 presents the left image of the reference pair, the ground truth as well as the depth maps produced by original AW algorithm, and our hardware implementation for visual comparison. The disparity maps correspond to the left stereo images and are scaled by a certain factor.

In Table 3, we have listed a number of hardware systems based on FPGA. Unfortunately, they are mostly no adequate discussion on the quality of the output disparity map. In contrast to our proposed system, we included the other presented algorithms not only on FPGA, but also on CPU and GPU. The numerical accuracies of these disparity maps and those generated by other related algorithms are listed in Table 4. Compared with original algorithm, our approach performs reasonable good in area with enough details but not so well in weakly textured areas. The evaluation engine calculates the percentage of incorrect matching pixels by pixel-wise comparison with the ground truth over the whole image. As our work aimed to gain the accuracy results rather than overall error rate that contains the occluded regions, all occluded or mismatched areas removed by our uniqueness check are just labeled "incorrect" without the process of extrapolation. As a result, this would pull-down the evaluation performance of our system. It can be seen that the complicated algorithms implemented on GPU generally get better results.

Table 3 Real time performance of reported stereo vision systems based on FPGA

	Image size	Matching method	Disparity range	Rectify	fps (60 MHz)	PDS(10^6)
Ambrosch et al. [44]	750 × 400	SAD Square Window	60	No	60	1080
Miyajima et al. [33]	640 × 480	SAD Square Window	60	Hard-wired	20	368.6
Mingxiang et al. [35]	320 × 240	SAD Square Window	32	No	100	247.8
Calderon et al. [36]	320 × 240	BSAD Square Window	16	No	142.2	174.7
Perri et al. [34]	512 × 512	SAD Square Window	255	No	25.6	1711.27
Murphy et al. [39]	320 × 240	SAD Square Window	20	No	150	230.4
Jin et al. [45]	640 × 480	Census Square Window	64	Hard-wired	200	2949.1
Diaz et al. [42]	1280 × 960	Phase Correlation	15	Hard-wired	52	975
Darabiha et al. [40]	360 × 256	Phase Correlation	60	No	20	110.6
Chang et al. [2]	352 × 288	Census Adaptive Weights	64	No	26	168.7
Our proposed	640 × 480	TSAD Adaptive Weights	60	Hard-wired	51	940

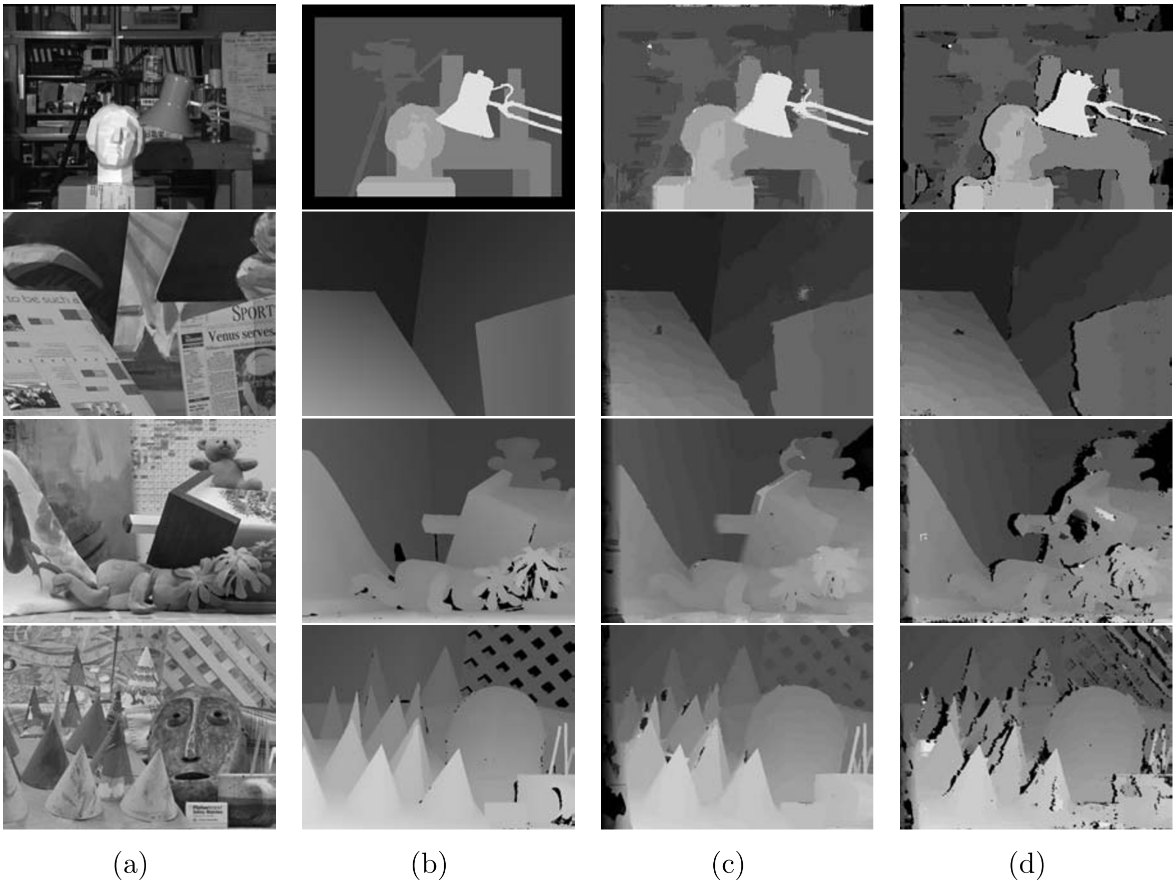

Figure 19 Evaluation result of the proposed system using Middlebury stereo-pairs. **(a)** left image. **(b)** Ground truth. **(c)** Disparity maps yielded by the original AW algorithm. **(d)** Our results.

Table 4 The accuracy of disparity maps

	Tsukuba			Venus			Teddy			Cones		
	Nonocc	All	Disc	Nonocc	All	Disc	Nonocc	All	Disc	Nonocc	All	Disc
Jin et al. [45] (FPGA)	9.97	11.56	20.29	3.59	5.27	36.82	12.5	21.5	30.57	7.34	17.58	21.01
Darabiha et al. [41] (FPGA)		19.59	37.62		10.51	31.52						
Ambrosch et al. [44] (FPGA)	12.1	13.4	28.2	4.06	4.86	25.9	12.3	19.7	31.3	6.91	14	19.2
Banz et al. [52] (FPGA)	6.8			4.1			13.3			4.1		
Veksler et al. [5] (CPU)	1.99	2.84	9.96	1.41	2.10	7.74	15.9	23.9	27.1	10.0	18.3	18.9
Grauer-Gray et al. [29] (GPU)	3.37	5.34	13.6	1.12	2.06	14.1	12.2	19.0	27.2	6.29	14.2	16.4
Gong et al. [30] (GPU)	1.36	3.39	7.25	2.35	3.48	12.2	9.82	16.9	19.5	12.9	19.9	19.7
Yang et al. [31] (GPU)	1.49	3.40	7.87	0.77	1.90	9.00	8.72	13.2	17.2	4.61	11.6	12.4
Adapt. Weights [1] (original)	1.38	1.85	6.90	0.71	1.19	6.13	7.88	13.3	18.6	3.97	9.79	8.26
Our proposed	3.73	5.65	10.3	1.59	3.46	10.4	13.5	20.6	20.9	10.8	18.2	19.0

However, the high power consumption makes them not suitable for embedded applications. The authors of [44,45] have presented the state-of-art systems based on FPGAs. Even though our simplified implementation does not perform as good as the original algorithm, it is already comparable with the state-of-art systems. Furthermore, it is important to mention that it is difficult to pick a good parameter for fix-size support region approaches as it is highly depends on the input datasets. As shown in [44], the optimum block size has to be selected before the final evaluation could be performed. The final optimum parameter is a compromise between four datasets as one may favor small window size while others favor large window size. The above problem is not as important for our algorithm. Actually, our algorithm shows a rather balanced accuracy over all four images. Our results are slightly worse in Teddy and Cones and much better in case of Tsukuba and Venus. This feature will make our system more robust to the scale changes of the actual scene.

Finally, it must be acknowledged that our algorithm is not among the top performance in the comparative results. The global algorithm generally performs better than correlation-based algorithm. However, the iterative, sequential operations of global algorithm still make it difficult to implement in median-scale FPGA. SGM is another interesting algorithm which performs an optimization through multiple paths across the entire image to approximate the global optimization. Recent advancements in FPGA technology have made it possible to implement SGM algorithm with necessary modifications [52,53]. Although it still needs to consume excessive memory to store the temporary cost of different aggregation path, it still have the potential to affect the trends in Hardware implementation due to its high accuracy results.

In addition to the evaluation presented above, our stereo system was used to test on real scenes. Figure 20 shows the disparity maps produced by our system. The red line in Figure 20c represents the Canny edge

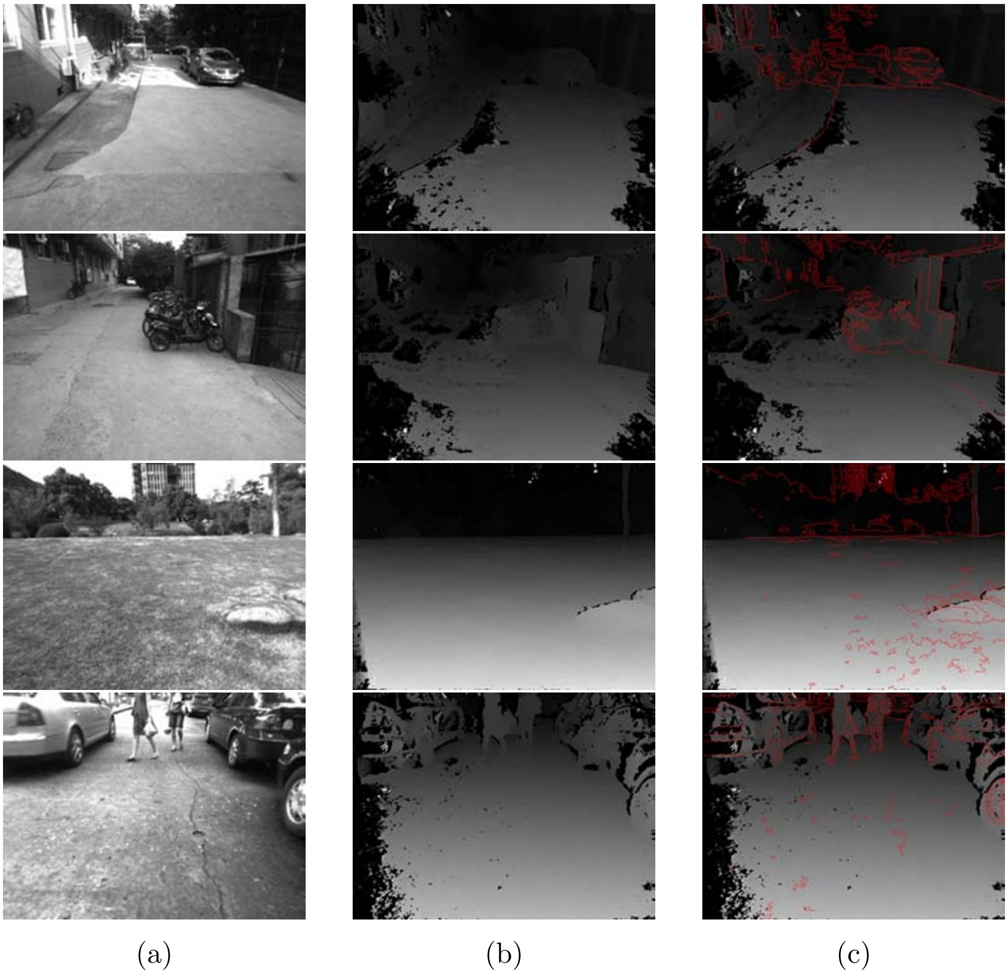

Figure 20 Experimental results on real scenes. **(a)** Left camera image. **(b)** Disparity map. **(c)** Overlay canny edge onto disparity map.

detection results of left image. The overlap view of Canny edge and disparity map shows the scene structure and object borders have been well detected.

6 Conclusion

In this article, we have proposed a high performance FPGA-based stereo vision system. Our system exploits a novel cost aggregation approach called adaptive support weights method. This approach has shown to be remarkably effective. It essentially similar to "segment-based" algorithm while avoids the difficult problem of image segmentation. However, this aggregation scheme is very expensive in terms of computation. As the weighting mask varies from pixel to pixel, it cannot be computed by means of incremental calculation schemes. Also it suffers from complex arithmetic operations like multiplication and division. Our analysis shown the necessity of trade-offs between the accuracy and efficient hardware implementation. With hardware friendly approximation, we demonstrate the feasibility of implementing this expensive computational task on hardware to achieve frame-rate performance. Evaluation results have shown that our implementation is among one of the best performing local algorithms in terms of quality. In addition, the highly parallelized pipeline structure makes system be capable to handle 640 × 480 pixels image at over 51 fps. The adaptive cost aggregation units of this system can also be reused as bilateral filter for noise reduction in other vision systems.

In the future, the proposed system will be used for higher level vision applications such as autonomous vehicle navigation. Some improvements still could be extended. It is expected that the accuracy performance can be improved using the pre-processing step to reject the matches belonging to poorly textured areas. Moreover, with the fast evolvement of FPGA technology, it is possible to include soft processor core within an FPGA device. This customization enables the integrated design for higher-level control tasks.

7 Competing interests

The authors declare that they have no competing interests.

Acknowledgements

The authors would like to thank Xin Du for his thoughts and suggestions, and Xinhuan Wang for his technical assistance during the course of this study. The authors would also like to acknowledge the financial support from the National Natural Science Foundation of China via grant 61001171, 60534070, and 90820306, and the Natural Science Foundation of Zhejiang Province (Grant No. Y1090881).

Author details

[1]Department of Information Science and Electronic Engineering, Zhejiang University, Yuquan Campus, Hangzhou, 310027, PR China [2]College of Computer and Software Engineering, Hangzhou Dianzi University, Hangzhou, 310018, PR China [3]School of Electronic and Information, Hangzhou Dianzi University, Hangzhou, 310018, PR China

Received: 31 March 2011 Accepted: 7 December 2011
Published: 7 December 2011

References

1. Yoon KJ, Kweon IS IS: **Adaptive support-weight approach for correspondence search.** *IEEE Trans Pattern Anal Mach Intell* 2006, **28(4)**:650-656.
2. Chang NYC, Tsai TH, Hsu BH, Chen YC, Chang TS: **Algorithm and architecture of disparity estimation with mini-census adaptive support weight.** *IEEE Trans Circ Syst Video Technol* 2010, **20(6)**:792-805.
3. Scharstein D, Szeliski R: **A taxonomy and evaluation of dense two-frame stereo correspondence algorithms.** *Int J Comput Vis* 2002, **47**:7-42.
4. Amini AA, Weymouth TE, Jain RC: **Using dynamic programming for solving variational problems in vision.** *IEEE Trans Pattern Anal Mach Intell* 1990, **12(9)**:855-867.
5. Veksler O: **Stereo correspondence by dynamic programming on a tree.** *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)* 2005, **2**:384-390.
6. Boykov Y, Veksler O, Zabih R: **Fast approximate energy minimization via graph cuts.** *IEEE Trans Pattern Anal Mach Intell* 2001, **29**:1222-1239.
7. Sun J, Zheng NN, Shum HY: **Stereo matching using belief propagation.** *IEEE Trans Pattern Anal Mach Intell* 2003, **25(7)**:787-800.
8. Hirschmüller H: **Accurate and efficient stereo processing by semi-global matching and mutual information.** *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)* 2005, **2**:807-814.
9. Li G, Zucker S: **Surface geometric constraints for stereo in belief propagation.** *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06)* 2006, **2**:2355-2362.
10. Klaus A, Sormann M, Karner K: **Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure.** *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR'06)* 2006, **3**:15-18.
11. Yang Q, Wang L, Yang R, Stewenius H, Nister D: **Stereo matching with color-weighted correlation, hierachical belief propagation and occlusion handling.** *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06)* 2006, **2**:2347-2354.
12. Yang R, Pollefeys M, Li S: **Improved real-time stereo on commodity graphics hardware.** *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'05)* 2005, 36-43.
13. Yang R, Pollefeys M: **A versatile stereo implementation on commodity graphics hardware.** *Real-Time Imag* 2005, **11**:7-18.
14. Pérez J, Sanchez P, Martinez M: **High memory throughput FPGA architecture for high-definition Belief-Propagation stereo matching.** *Proceedings of the 3rd International Conference on Signals, Circuits and Systems (SCS)* 2009, 1-6, IEEE.
15. Liang C, Cheng C, Lai Y, Chen L, Chen H: **Hardware-efficient belief propagation.** *IEEE Trans Circ Syst Video Technol* 2009, **21(5)**:525-537.
16. Okutomi M, Kanade T: **A locally adaptive window for signal matching.** *Int J Comput Vis* , 2 1992, **7**:143-162.
17. Bobick A, Intille S: **Large occlusion stereo.** *Int J Comput Vis* , 3 1999, **33**:181-200.
18. Fusiello A, Roberto V, Trucco E: **Symmetric stereo with multiple windowing.** *Int J Pattern Recogn Artif Intell* , 8 2000, **14**:1053-1066.
19. Hirschmüller H, Innocent P, Garibaldi J: **Real-time correlation-based stereo vision with reduced border errors.** *Int J Comput Vis* 2002, **47**:229-246.
20. Boykov Y, Veksler O, Zabih R: **A variable window approach to early vision.** *IEEE Trans Pattern Anal Mach Intell* 1998, **20(12)**:1283-1294.
21. Veksler O: **Fast variable window for stereo correspondence using integral images.** *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'03)* 2003, **1**:556-561.
22. Tombari F, Mattoccia S, Di Stefano L, Addimanda E: **Classification and evaluation of cost aggregation methods for stereo correspondence.** *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)* 2008, 1-8.
23. Gong M, Yang R, Wang L: **A performance study on different cost aggregation approaches used in real-time stereo matching.** *Int J Comput Vis 2007* 2007, **75(2)**:283-296.

24. Mühlmann K, Maier D, Hesser J, Mnner R: **Calculating dense disparity maps from color stereo images, an efficient implementation.** *Int J Comput Vis* 2002, **47**:79-88.
25. Zinner C, Humenberger M, Ambrosch K, Kubinger W: **An optimized software-based implementation of a census-based stereo matching algorithm.** *Adv Visual Comput* 2008, **5358**:216-227.
26. Kanade T, Yoshida A, Oda K, Kano H, Tanaka M: **A stereo machine for video-rate dense depth mapping and its new application.** *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR'96)* 1996, 196-202.
27. Konolige K: **Small vision systems: hardware and implementation.** In *Proceedings of the Eighth International Symposium on Robotics Research. Volume 8*. MIT Press; 1997:203-212.
28. Yang R, Pollefeys M: **Multi-resolution real-time stereo on commodity graphics hardware.** *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'03)* 2003, **1**:211-217.
29. Grauer-Gray S, Kambhamettu C: **Hierarchical belief propagation to reduce search space using cuda for stereo and motion estimation.** *Proceedings of the Workshop on Applications of Computer Vision (WACV)* 2009, 1-8, IEEE.
30. Gong M, Yang YH: **Near real-time reliable stereo matching using programmable graphics hardware.** *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)* 2005, **1**:924-931.
31. Yang Q, Wang L, Yang R, Wang S, Liao M, Nister D: **Real-time global stereo matching using hierarchical belief propagation.** *Proceedings of the British Machine Vision Conference (BMVC'06)* 2006, 989-998.
32. Woodfill J, Von Herzen N: **Real-time stereo vision on the PARTS reconfigurable computer.** *Proceedings of the Symposium on FPGAs for Custom Computing Machines* 1997, 201-210, IEEE.
33. Miyajima Y, Maruyama T: **A real-time stereo vision system with FPGA.** *Field Programmable Logic And Application* Springer; 2003, 448-457.
34. Perri S, Colonna D, Zicari P, Corsonello P: **SAD-based stereo matching circuit for FPGAs.** *Proceedings of the International Conference on Electronics, Circuits and Systems (ICECS'06)* 2006, 846-849, IEEE.
35. Mingxiang L, Yunde J: **Stereo vision system on programmable chip (SVSoC) for small robot navigation.** *Proceedings of International Conference on Intelligent Robots and Systems* 2006, 1359-1365, IEEE.
36. Calderon H, Ortiz J, Fontaine J: **High parallel disparity map computing on FPGA.** *Proceedings of the International Conference on Mechatronics and Embedded Systems and Applications (MESA)* 2010, 307-312, IEEE.
37. Woodfill JI, Gordon G, Buck R: **Tyzx DeepSea high speed stereo vision system.** *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04)* 2004, **3**:41-48.
38. Ambrosch K, Kubinger W, Humenberger M, Steininger A: **Flexible hardware-based stereo matching.** *EURASIP J Embed Syst* 2008, **2008**:1-18.
39. Murphy C, Lindquist D, Rynning A, Cecil T, Leavitt S, Chang M: **Low-cost stereo vision on an FPGA.** *Proceedings of the Symposium on Field-Programmable Custom Computing Machines (FCCM)* 2007, 333-334, IEEE.
40. Darabiha A, Rose J, MacLean WJ: **Video-rate stereo depth measurement on programmable hardware.** *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)* 2003, **1**:203-210.
41. Darabiha A, MacLean W, Rose J: **Reconfigurable hardware implementation of a phase-correlation stereoalgorithm.** *Mach Vis Appl* , 2 2006, **17**:116-132.
42. Diaz J, Ros E, Carrillo R, Prieto A: **Real-time system for high-image resolution disparity estimation.** *IEEE Trans Image Proces* 2007, **16**:280-285.
43. Masrani D, MacLean W: **A real-time large disparity range stereo-system using FPGAs.** *Proceedings of the Asian Conference on Computer Vision (ACCV'06* Springer; 2006, 42-51.
44. Ambrosch K, Kubinger W: **Accurate hardware-based stereo vision.** *Comput Vis Image Understand* , 11 2010, **114**:1303-1316.
45. Jin S, Cho J, Dai Pham X, Lee KM, Park SK, Kim M, Jeon JW: **FPGA design and implementation of a real-time stereo vision system.** *IEEE Trans Circ Syst Video Technol* 2010, **20**:15-26.
46. Tomasi C, Manduchi R: **Bilateral filtering for gray and color images.** *Proceedings of International Conference on Computer Vision (ICCV'98)* 1998, 839-846, IEEE.
47. Pham T, Van Vliet L: **Separable bilateral filtering for fast video preprocessing.** *Proceedings of International Conference on Multimedia and Expo (ICME'05)* 2005, 4-8, IEEE.
48. Fusiello A, Trucco E, Verri A: **A compact algorithm for rectification of stereo pairs.** *Mach Vis Appl* 2000, **12**:16-22.
49. Shimizu M, Okutomi M: **Precise sub-pixel estimation on area-based matching.** *Proceedings of International Conference on Computer Vision (ICCV'01)* 2001, **1**:90-97, IEEE.
50. Di Stefano L, Marchionni M, Mattoccia S: **A fast area-based stereo matching algorithm.** *Image Vis Comput* 2004, **22(12)**:983-1005.
51. Bates G, Nooshabadi S: **FPGA implementation of a median filter.** *Proceedings of the IEEE Conference on Speech and Image Technologies for Computing and Telecommunications (TEN-CON'97)* 1997, **2**:437-440.
52. Banz C, Hesselbarth S, Flatt H, Blume H, Pirsch P: **Real-time stereo vision system using semi-global matching disparity estimation: architecture and FPGA-implementation.** *Proceedings of the International Conference on Embedded Computer Systems (SAMOS)* 2010, 93-101, IEEE.
53. Gehrig S, Eberli F, Meyer T: **A real-time low-power stereo vision engine using semi-global matching.** *Proceedings of the International Conference on Computer Vision Systems (ICVS)* Springer; 2009, 134-143.

doi:10.1186/1687-5281-2011-20
Cite this article as: Ding *et al.*: Real-time stereo vision system using adaptive weight cost aggregation approach. *EURASIP Journal on Image and Video Processing* 2011 **2011**:20.