SoC and FPGA Oriented High-quality Stereo Vision System

Yanzhe Li, Kai Huang
Institute of VLSI Design
Zhejiang University
Hangzhou, China
Email: {liyz,huangk}@vlsi.zju.edu.cn

Luc Claesen
Engineering Technology - Electronics-ICT Dept.
Hasselt University
3590 Diepenbeek, Belgium
Email: luc.claesen@uhasselt.be

***Abstract*—Stereo matching is a crucial step for acquiring depth information from stereo images. However, it is still challenging to achieve good performance in both speed and accuracy for various stereo vision applications. In this paper, a hardware-compatible stereo matching algorithm is proposed; its associated hardware implementation is also presented. The proposed algorithm can produce high-quality disparity maps with the use of mini-census transform, segmentation-based adaptive support weight and effective refinement. Moreover, the proposed implementation is optimized as a fully pipelined and scalable hardware system. The proposed design is evaluated based on the Middlebury benchmarks and the average overall error rate is 6.10%. The experimental results indicate that the accuracy is competitive with some state-of-art software implementations.**

I. Introduction

Stereo vision is one of the most active research topics in computer vision and widely used in many applications. As a key role in a stereo vision system, stereo matching takes a pair of rectified images, estimates the movement of each pixel from one image to the other and expresses this movement in a disparity map. So it is a complicated and time-consuming procedure. Considering that many applications often require high performance and real-time processing speed, it is hard for software implementations of stereo matching algorithms on CPU to meet these constraints.

Hardware acceleration of stereo matching algorithms has been done extensively using DSPs, GPUs and dedicated hardware. However, DSPs do not provide enough computational power to support real-time processing, while GPUs consume excessive power. Dedicated hardware approaches that use FPGAs and ASICs can provide a balance between computational power and energy efficiency. A segmentation based design with adaptive support weight (ADSW) was proposed on FPGAs [1]. It is inspired by the algorithm proposed in [2], which was the best performing local method on the Middlebury benchmarks [3] for a long time. But the result of the hardware design suffers because of the small fixed window size. In [4], a hardware solution provided high-quality disparity results in ASICs based on the mini-census adaptive support weight (MCADSW) method. But it only targets low resolution images for real-time and requires high memory bandwidth. In [5], A. Akin proposed a hardware-oriented adaptive window size disparity estimation (AWDE) algorithm and its real-time reconfigurable hardware implementation for high resolutions. Although its result is outstanding among hardware implementations, the accuracy can not compete with current software implementations.

In this paper, the mini-census and segmentation-based ADSW algorithms are combined to achieve a high matching accuracy. Different from many hardware designs without refinement, we present a disparity refinement step with segmentation information and find that it could improve the quality of initial disparity maps significantly. Moreover, a fully pipelined and scalable architecture is implemented for the proposed algorithm. To make a tradeoff between accuracy and speed, some techniques such as simplified weight function and adaptive window size are adopted. The design is evaluated with the Middlebury benchmarks quantitatively and visual satisfactory results are also provided. The experimental results show that the accuracy can be competitive with some state-of-art software implementations.

II. Stereo Matching Algorithm

A. Algorithm Overview

Cost calculation, cost aggregation, disparity selection and disparity refinement are four well-defined steps in stereo matching algorithms [6]. Here we utilize the mini-census transform in the cost calculation step and the segmentation-based ADSW algorithm in the cost aggregation step. The refinement step consists of three stages: consistency check, disparity voting and invalid disparity inpainting. Finally, disparity maps of both images are finished simultaneously. The proposed algorithm is suggested in Fig. 1.

The mini-census transform extracts 6-pixel neighborhood information of a center pixel and encodes the information in a vector [4]. It reduces the memory utilization due to the lower amount of storage bits. Then the matching cost is defined as the Hamming distance among the output vectors.

The segmentation-based ADSW algorithm employs the segmentation information within the weight cost function to increase the robustness of the matching process [2]. The weight cost function generates the weight coefficients w_r and w_t as shown in (1).

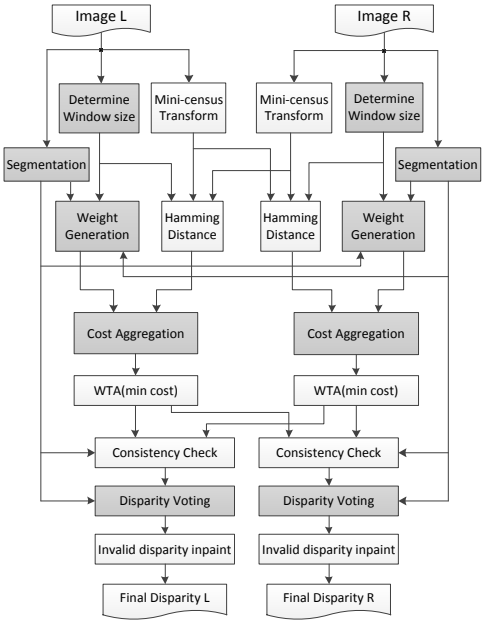

Fig. 1. Overview of the proposed algorithm.

$$w_{r,t} = \begin{cases} 1.0 & p_i \in S_c \\ \exp(-\frac{d_c(I_{r,t}(p_i), I_{r,t}(p_c))}{\gamma_c}) & \text{otherwise} \end{cases} \quad (1)$$

S_c is the segment where the central point of p_c (or q_c) lies, d_c is the Euclidean distance between two triplets in the CIELAB color space, and γ_c is a parameter. The final aggregated cost is given by summing up all the weighted matching costs in the support windows W_r and W_t then normalizing with the weights sum as shown in (2).

$$C(p_c, q_c) = \frac{\sum_{p_i \in W_r, q_i \in W_t} w_r(p_i, p_c) w_t(q_i, q_c) MC(p_i, q_i)}{\sum_{p_i \in W_r, q_i \in W_t} w_r(p_i, p_c) w_t(q_i, q_c)} \quad (2)$$

A tree-structure winner-takes-all (WTA) method is used in the disparity selection step. Then disparity refinement operates on the initial disparity maps. The consistency check is to verify whether the disparity maps satisfy (3).

$$d'_p(x - d_p(x,y), y) == d_p(x,y) \cap S_p == S'_p \quad (3)$$

The disparities of pixel p and p' are in the target and reference images respectively. S_p and S'_p are the corresponding segmentation information. If they are different, the consistency check fails and the disparity is marked as invalid. The disparity voting updates the center disparity with the most present valid disparity in its local support window. The disparity inpainting replaces the invalid disparity with the closest valid disparity on its scanline. The median filtering is not used in the design for its complicated hardware implementation with negligible quality improvement.

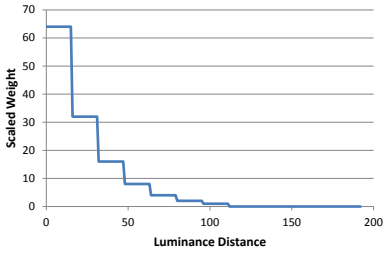

Fig. 2. Weight function.

B. Hardware-oriented Optimization

To reduce computation complexity and make the algorithm more hardware-compatible, we propose some optimizations shown in the shaded blocks in Fig. 1. Luminance (Y) color representation is adopted instead of CIELAB color representation. Manhattan distance is used rather than Euclidean distance to avoid square root computations. A very simple method which partitions the image into segments using thresholding [1] is adopted. By this way, luminance distance can be largely represented by segment distance. In weight generation, the weight allows the pixel with luminance similar to the center pixel to have more influence on the matching cost. A scale-and-truncate approximation of the weight function is proposed, and the curve is shown in Fig. 2. So the multiplication of the weight coefficients is reduced to a left shift operation.

To make a tradeoff between accuracy and speed, a method called AWDE [5] is introduced. It uses different window sizes for different textures on the image and determines the window size by the mean absolute deviation (MAD) of the pixel in the center of 7×7 block. 49 pixels are constantly sampled with different intervals so that low computation cost is utilized for large support window sizes.

When disparity voting is implemented, a 25×25 support window is applied to achieve a reliable result. To determine the most frequent disparity value in the window efficiently, a vertical-horizontal approach is applied. It firstly searches for the majority disparity in each column vertically, then picks out the majority one horizontally as the final disparity. In addition to the computation complexity reduction, the approach also reduces the internal bandwidth.

III. Hardware Implementation

A hardware architecture is designed for the proposed algorithm, as shown in Fig. 3. The key to improve throughput is to be a fully pipelined architecture, which can be decomposed into three stages: pre-processing, stereo-matching and post-processing. The pre-processing stage does not deal with any disparity information and performs pixel-based operations on each pixel for both images independently. The stereo-matching stage operates on the transformed data and computes disparity maps. Here the computing structure combines the disparity-level parallelism with the pixel-level parallelism for cost aggregation. The post-processing stage is implemented to refine the initial disparity maps when they are ready.

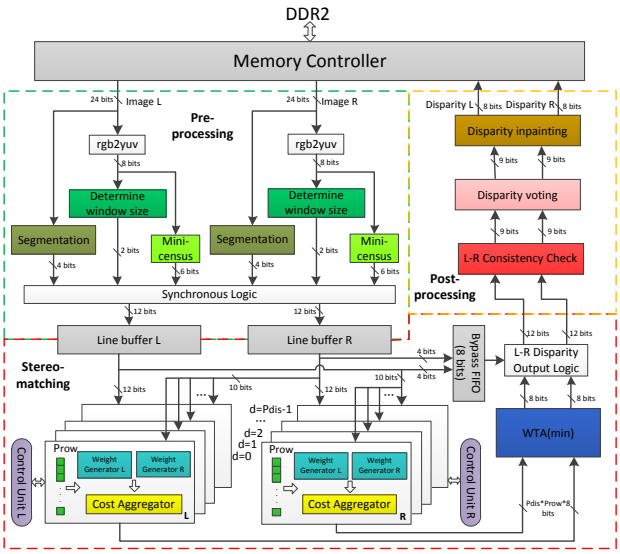

Fig. 3. Block diagram of the proposed architecture.

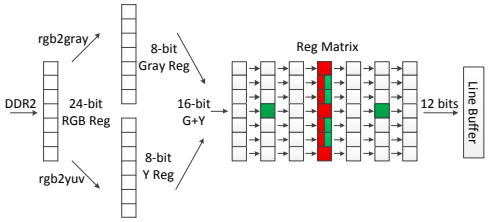

Fig. 4. Pipeline architecture in the pre-processing stage.

For both images, source image pixels are fetched in scanline order and disparity maps are generated through the pipeline. Each pixel is read only once from the external memory during the whole processing flow, so that memory bandwidth is not a limitation. The design can be scaled with image resolution, disparity range, parallelism degree and segmentation level to achieve maximum flexibility.

A. Pre-processing Stage

In the pre-processing stage, 24-bit source pixels of RGB are fetched hereafter the color space converters are implemented. 8-Bit grayscale values are used for the segmentation module. A number of segments k is given as input and a label is computed by a simple method that multiplies the grayscale value by the value of $k/256$. Here k is defined as a power of 2 so that the left shift operation is exploited. 8-Bit Y values are used for the mini-census transform and the window size determination. As the operations are all window-based, register matrix is employed to provide pipelined window content in Fig. 4. The whole register matrix of 7×7 is used for window size determination, meanwhile the six green ones are used for mini-census transform and the center column of red ones are used for segmentation. Finally, a result of 12 (2+6+4) bits is written into the line buffer.

To satisfy the pixel-level parallelism, P_{row} pixels in one column are processed in parallel. So the line buffer is composed of $(P_{row}+6)$ dual port BRAMs to build a wide throughout, and the size of register matrix is extended to $(P_{row}+6)\times 7$. Source data and temporary results in each column can be reused to reduce the computational requirements because a column is usually a part of multiple horizontally overlapping windows.

B. Stereo-matching Stage

The control unit begins to fetch the transformed data from the line buffers and allocate them to aggregation modules. The line buffers are exploited to replace the processed pixels with the new required pixels during the process. A total of P_{dis} aggregation modules are generated to deal with different disparities. In each module, P_{row} pixels are processed in parallel. Here the generate statement in Verilog is used for these two parameters so that the hardware can be scalable. The weight generator receives segmentation information from both images depending on the window size of the center pixel and computes weight coefficients. The cost aggregator calculates the Hamming distances then shifts them with the according weights. The final aggregated cost is computed by summing the weighted scores using a tree adder, then normalizing it.

Aggregated costs are sent to the tree-structure WTA module to select the disparity with the minimum cost. The Bypass FIFO is used to preserve segmentation information associated with each pixel for the next stage. The output is a data stream that consists of 8-bit initial disparity and 4-bit segmentation information for each image.

C. Post-processing Stage

In the post-processing stage, the consistency check module compares the initial disparity maps and segmentation information in order to generate one more bit which labels each disparity as valid or not. The disparity voting module updates the center disparity in the 25×25 support window using the vertical-horizontal method. Here, a bitwise fast voting technique [7] is applied to handle each column. It drives each bit of the most frequent disparity independently from the other bits so that the hardware cost depends on the number of disparity bits in binary. While counting bit votes, the valid information must be taken into account.

After disparity voting, disparity maps will be written back into the DDR2 memory. The disparity inpainting module checks the valid bit and replaces the invalid disparity with the closest valid disparity when writing it out.

IV. Experimental Results

To discuss the quality of our design, the disparity maps are evaluated based on the Middlebury benchmarks. Table I lists the accuracy comparison with some state-of-art implementations. The error rate is the overall error rate with the tolerance of one disparity level [6]. The 1st is the AD-Census algorithm implemented on GPU. It is challenging to realize it into an FPGA because of its multi disparity enhancement functions. The algorithms in [2] and [9], which are also software implementations of local methods, have a higher error rate than the proposed algorithm. The design

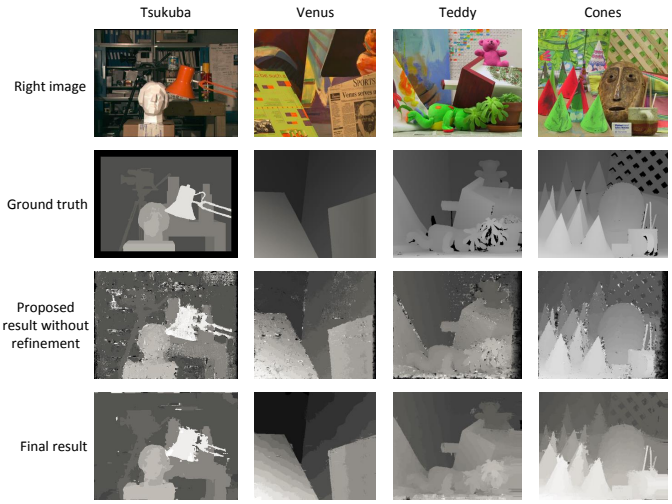

Fig. 5. True disparity maps and experimental results.

TABLE I
ACCURACY COMPARISON ON MIDDLEBURY BENCHAMRK

Evaluation	Tsukuba	Venus	Teddy	Cones	Average
AD-Census [8]	1.48	0.25	6.22	7.25	3.80
SegSupport [2]	1.62	0.64	14.2	9.87	6.58
VariableCross [9]	2.65	0.96	15.1	12.7	7.85
Wang et al. [10]	3.27	0.89	12.1	7.74	6.00
Jin et al. [11]	2.17	0.60	12.4	8.97	6.04
MCADSW [4]	2.80	0.64	13.7	10.1	6.81
AWDE-IR [5]	6.53	5.01	12.3	11.2	8.76
Ttofis et al. [1]	6.04	7.47	28.1	25.9	16.9
Proposed algorithm without refinement	11.3	7.64	19.3	15.3	13.4
Proposed algorithm with refinement	4.37	0.90	10.5	8.64	6.10

in [10] is the newest published hardware implementation on an FPGA and it utilizes variable support regions and semi-global optimization to have an accurate result. The system in [11] uses a local stereo matching method and shows comparable accuracy performance. The comparison shows that the accuracy of our design is not only among the best in hardware accelerated stereo systems, but also competitive with current software implementations.

To indicate the effect of the refinement step, the final disparity maps are compared with the initial disparity maps. The results are shown in Fig. 5. The refinement step contributes significantly to the final disparity maps and many visual improvements are obvious, including the elimination of speckle noise, few errors at the image borders and sharply delineated edges. The quantitative results also prove it.

To further evaluate the proposed algorithm, some high-definition images are used. Here the image pair of Art is at a 1110×1390 resolution for a 128-pixel disparity range. The result is shown in Fig. 6 and the error rate is 12.85%.

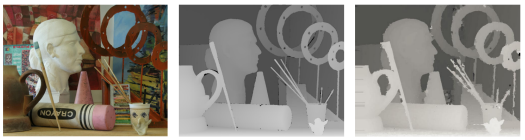

Fig. 6. Results of high-definition images. The 1st is the original image. The 2nd is the ground truth. The 3rd is the proposed result.

V. CONCLUSION

This paper has proposed a stereo matching algorithm based on mini-census transform and segmentation-based ADSW. The disparity refinement step with segmentation information is presented and improves the quality of disparity maps significantly. Furthermore, a fully pipelined and scalable hardware architecture is designed after some hardware-oriented optimizations. The design is evaluated with the Middlebury benchmarks and the average overall error rate is 6.10%. To improve the accuracy, a part of global matching algorithms can be introduced into the algorithm in the future.

ACKNOWLEDGMENT

The research in this paper was sponsored in part by the Belgian FWO (Flemish Research Council) and the Chinese MOST (Ministry of Science and Technology) bilateral cooperation project number G.0524.13.

REFERENCES

[1] C. Ttofis and T. Theocharides, "Towards accurate hardware stereo correspondence: A real-time fpga implementation of a segmentation-based adaptive support weight algorithm," in *in Proc. Design Autom. Test Eur. Conf. Exhibit. (DATE)*, 2012, pp. 703–708.

[2] F. Tombari, S. Mattoccia, and L. Di Stefano, "Segmentation-based adaptive support for accurate stereo correspondence," in *Proceedings of the 2Nd Pacific Rim Conference on Advances in Image and Video Technology*, 2007, pp. 427–438.

[3] http://vision.middlebury.edu/stereo/.

[4] N. Y.-C. Chang, T.-H. Tsai, B.-H. Hsu, Y.-C. Chen, and T.-S. Chang, "Algorithm and architecture of disparity estimation with mini-census adaptive support weight," *IEEE Trans. Cir. and Sys. for Video Technol.*, vol. 20, no. 6, pp. 792–805, Jun. 2010.

[5] A. Akin, I. Baz, A. Schmid, and Y. Leblebici, "Dynamically adaptive real-time disparity estimation hardware using iterative refinement," *Integration, the VLSI Journal*, vol. 47, no. 3, pp. 365 – 376, 2014.

[6] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision*, vol. 47, no. 1, pp. 7–42, 2012.

[7] K. Zhang, J. Lu, G. Lafruit, R. Lauwereins, and L. V. Gool, "Real-time accurate stereo with bitwise fast voting on cuda," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, Sept 2009, pp. 794–800.

[8] X. Mei, X. Sun, M. Zhou, S. Jiao, H. Wang, and X. Zhang, "On building an accurate stereo matching system on graphics hardware," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, Nov 2011, pp. 467–474.

[9] K. Zhang, J. Lu, and G. Lafruit, "Cross-based local stereo matching using orthogonal integral images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 7, pp. 1073–1079, July 2009.

[10] W. Wang, J. Yan, N. Xu, Y. Wang, and F. H. Hsu, "Real-time high-quality stereo vision system in fpga," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 10, pp. 1696–1708, Oct 2015.

[11] M. Jin and T. Maruyama, "A fast and high quality stereo matching algorithm on fpga," in *Field Programmable Logic and Applications (FPL), 2012 22nd International Conference on*, Aug 2012, pp. 507–510.