IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 6, 2018, accepted September 10, 2018, date of publication September 17, 2018, date of current version October 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2870672

A One-Shot FPGA Based Depth Acquisition System

ZHEN-NI LI[1], YU-LIANG GAO[1], ZE-KUN WANG[2], AND JIAO WANG[1]
[1]College of Information Science and Engineering, Northeastern University, Shenyang 110819, China
[2]Broadcom Corporation, Shanghai 201210, China

Corresponding author: Jiao Wang (wangjiao@ise.neu.edu.cn)

This work was supported by the Fundamental Research Funds for the Central Universities of China under Grant N170412022.

ABSTRACT This paper introduces a field-programmable gate array (FPGA)-based one-shot depth acquisition system, which is a kind of Kinect-like Depth sensing. The proposed implementation integrates the random binary pattern with an improved matching method, which is able to obtain dense and precise depth maps. Scene images captured by a camera are transferred to FPGA (Xilinx Virtex7 XC7VX690T-2FFG1761C) through a CamerLink interface, and the acquired depth results are sent to host computer through a high-speed USB3.0 interface. An elaborate FPGA architecture is designed to realize efficient and accurate calculations. The proposed system outperforms the existing stereo systems, not only on the quality of depth acquisition, but also on the reconstruction speed for which our system achieves 90 fps at a resolution of 1280 × 960. Besides, Kinect-like depth sensing has an inherent advantage in deployments, much easier than other kinds of systems. Our proposed system has extensive applications not only in the field of vision processing, but also in the fields where block matching is needed.

INDEX TERMS Block matching, depth sensing, FPGA, Gabor, Kinect.

I. INTRODUCTION

Stereo vision, a well-known topic in the field of computer vision, represents an important technique for extracting depth information and for 3D reconstruction. Normally, a stereo vision system has two cameras to capture images. And a disparity map is formed intuitively considering the pixels that are horizontally shifted. Moreover, the disparities between images are processed for depth acquisition. This process is very complicated and time-consuming, therefore it is hard to achieve real-time performance while considering the cost and power consumption issues. To solve this problem, during the past decades, many implementation platforms and corresponding algorithms are proposed and discussed [1] to achieve binocular stereo vision [2] or monocular stereo vision [3] with high performance. And lots of improvements in hardware technology have been achieved to reduce the computational complexity. The most widely used implementation platforms are GPU and field programmable gate array (FPGA). GPU supports parallel computing which is indispensable for achieving real-time performance, nevertheless, it is very power-consuming, therefore is not suitable for embedded applications. FPGA, in the contrast, is able to support stereo vision system computing with a relatively low power consumption and cost. Thus, FPGA design of stereo vision systems has become an active research topic in recent years.

With the progress of the stereo correspondence algorithms [4], [5], a large number of FPGA architectures have been proposed. In [6], the proposed system implemented on a Virtex-5 FPGA platform, yielding a real-time performance of 30 fps for a stereo image pair with the resolution of 640 × 480. However, the system could not achieve higher performance. Jin *et al.* proposed a real-time FPGA-based stereo vision system, which can generate disparity images with the same resolution at the designated frame rate. Their system utilizes a local census-based stereo matching method and it can achieve up to 60 fps for image pair at a 640 × 480 resolution [7]. Nevertheless, this design could not deal with images at higher resolutions. Zhang [8] implemented a system on a single FPGA EP3SL150, which have achieved the fastest processing speed (60 fps) for stereo matching at a 1024 × 768 resolution. In [9], images of size up to 1024 × 1024 pixels with 32 disparity levels could be dealt with at 47 fps. And the algorithms present low FPGA resource usage, along with noticeable performance in disparity map quality. Thomas *et al.* [10] reported a system mapped on Xilinx Virtex-5 XC5VLX110T FPGA, and achieves 87 fps for a full HD resolution of 1920 × 1200. Dehnavi and Eshghi [11] proposed a FPGA based real-time on-road stereo vision system, it operated at 53.5 fps for 1920 × 1080 images with 96 disparity levels. Puglia and Vigliar [12] proposed a real-time low-power FPGA architecture for stereo vision, and the design could reach the processing ability of 1024 × 768 images, 64 disparity levels and 30fps with a power usage of only 0.17W.

In this paper, we propose the design and implementation based on FPGA allowing a high speed at 65 fps with 1920 × 1080 pixel images and the paper addresses Kinect-like depth sensing [13]. Compared with the stereo vision systems mentioned above, the system computes at an extremely faster arithmetic speed and could obtain a higher operational accuracy in the experiments. This design supports the depth acquisition of different resolution images and the processing speed is proportional to the resolution. In addition, the implementation of the system based on FPGA is effective in preserving high matching accuracy, especially on the edge parts.

II. ALGORITHM

Block-matching algorithm has been widely applied in various fields, such as video coding [14] and image reconstruction [15]. The performance bottlenecks of the algorithm are computational complexity and accuracy (e.g., blocking artifacts). To decrease these negative influence, an optimized matching method has been proposed in [16], in which a binary pattern with periodic density and random positions is proposed, meanwhile Gabor filter is adopted to improve the depth results by typical Block Matching method. There are several steps in the work flow of the proposed algorithm.

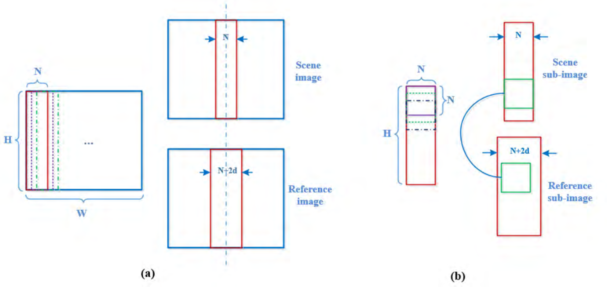


FIGURE 1. Block matching procedure: (a) Sub-image division. (b) Block matching.

First, the sub-image division method is shown in Fig. 1(a) for the reference and scene image. Then the algorithm takes overlapped blocks from a scene sub-image, and searches the most similar match in the corresponding reference sub-image, as shown in Fig. 1(b). It is worth mentioning that the system adopts the normalized cross-correlation (NCC) over two blocks as the measurement [17]. The disparity is calculated as follows

$$\gamma = \frac{\sum(I_r - \bar{I}_r)(I_s - \bar{I}_s)}{\sqrt{(\sum(I_r - \bar{I}_r)^2)(\sum(I_s - \bar{I}_s)^2)}}, \tag{1}$$

Where I_s and I_r indicate the pixel in a block from scene sub-image and reference sub-image respectively, and $\bar{I}_s$ and $\bar{I}_r$ denote the average intensity in the two blocks. In order to boost performance of FPGA implementation, (1) can be converted to another form.

$$\gamma = \frac{N^2 \sum I_r I_s - \sum I_r \sum I_s}{\sqrt{(N^2 \sum I_r^2 - (\sum I_r)^2)(N^2 \sum I_s^2 - (\sum I_s)^2)}}. \tag{2}$$

There are several computational parts in the equation, including product sum of two blocks, sum and square sum of a block.

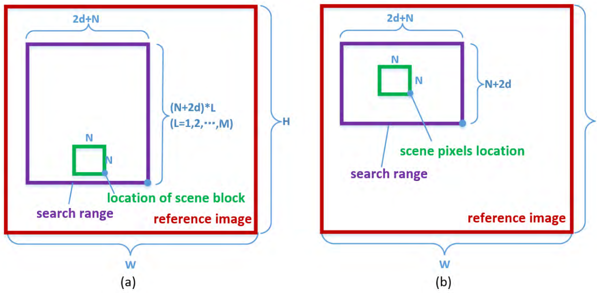


FIGURE 2. Search mode: (a) Brute-force mode. (b) Prediction mode.

After deriving the formula, we propose two kinds of search modes, comprising brute-force mode and prediction mode. The difference is the search area in corresponding reference image, as shown in Fig. 2. The reason using prediction mode is that neighboring blocks tend to have similar disparity due to surface smoothness. In practice, we firstly apply prediction mode to improve the search speed. If the highest NCC value found in the two modes is larger than threshold, we keep this value and the corresponding disparity for all pixels in the current scene block. Otherwise, we go back to brute-force mode.

Subsequently, after all scene sub-images get their disparity and correlation maps, the disparity on their overlapped pixels are determined by the one with the highest correlation value. After the block matching, we can obtain the period of phase difference between a pair of matching pixels, that is, C_0 in (3) by the following equation.

$$C_0 = \frac{k_1 \gamma'}{2\pi} \tag{3}$$

where γ' is the final block-matching result corresponding to the original captured image pixels. k_1 is the local frequency at the reference image pixel with the coordinate x_p.

To reduce the noise sensitivity, the algorithm adopts the Gabor filter to get phase values of the captured image and the reference image separately. For image filtering [18], two-dimensional Gabor Filter is essentially a convolution operation between original image and a Gabor template. Mathematically, the convolution arithmetic of time domain signal is equivalent to directly product calculation in frequency domain [19]. The design details of the coarse matching method can be found in [16]. The coordinate is roughly estimated by the following equation.

$$x_{pn} = \frac{[k_2 x_c + \phi_2]_{2\pi} + 2\pi C_0}{k_1} (\phi_1 = 0) \tag{4}$$

where C_0 is the period to be determined in (3). x_{pn} is a rough estimation of x_p. k_2 is the local frequency at the target image pixel with the coordinate x_c. ϕ_1 is the initial phase values of the reference image pixel at coordinate x_p. ϕ_2 is the initial phase values of the target image pixel at coordinate x_c.

Ultimately, to compensate the errors resulting from the initial phase which is assigned as 0, the fine matching method is carried out. Since x_p and x_{pn} are in the same period, we assume that $x_p = x_{pn} + \Delta x_p$. Then the fine matching formula is obtained as follows

$$x_p = x_{pn} + \frac{[k_2 x_c + \phi_2]_{2\pi} - [k_1 x_{pn} + \phi_1]_{2\pi}}{k_1} \tag{5}$$

Consequently, the correspondence between x_p in the reference image and x_c in the captured images is determined. After the fine matching, the depth value of each pixel in the image is calculated by triangulation of line-plane intersection.

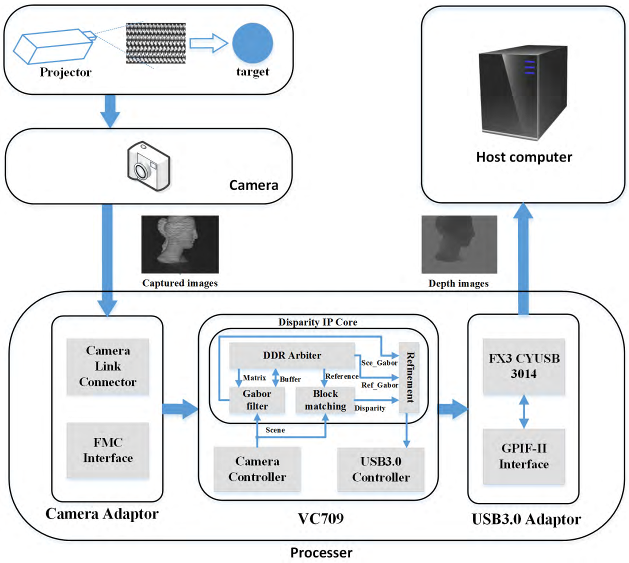


FIGURE 3. Schematic diagram of our proposed structure.

III. FPGA IMPLEMENTATION

The schematic diagram of our proposed depth acquisition system is shown in Fig.3. The projector projects the pattern on the target and images are captured by a camera. The image data is transferred to VC709, an FPGA evaluation board (XC7VX690T-2FFG1761C), through Camera Link interface. Then the depth maps generated by FPGA are sent to host computer through USB3.0 interface, where CYPRESS EZ-USB FX3 board is employed.

Meanwhile, Fig. 3 also demonstrates the high-level architecture of FPGA implementation. The captured images, scene pixel stream, are imported into the Disparity IP Core simultaneously. Besides, DDR Arbiter module not only provides other requisite data, such as Reference image, Gabor Template Matrix and reference Gabor results, but also efficiently buffers immediate data. Ultimately, the Refinement submodule encompasses all calculated results and generates the final depth values.

A. BLOCK MATCHING

As shown in Fig. 4, all input pixels, including scene pixels and reference pixels, are streamed with a stationary data sequence. To facilitate algorithm implementation and achieve real-time processing, there are several core modules, which are Pre-processing, Disparity Control, Disparity Generation, Arbiter and Post-processing.

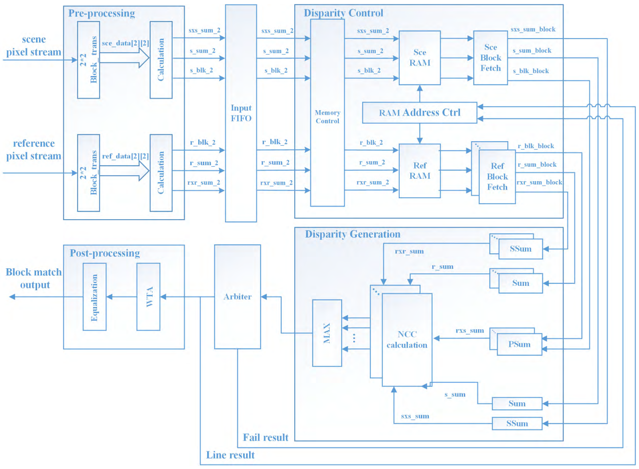
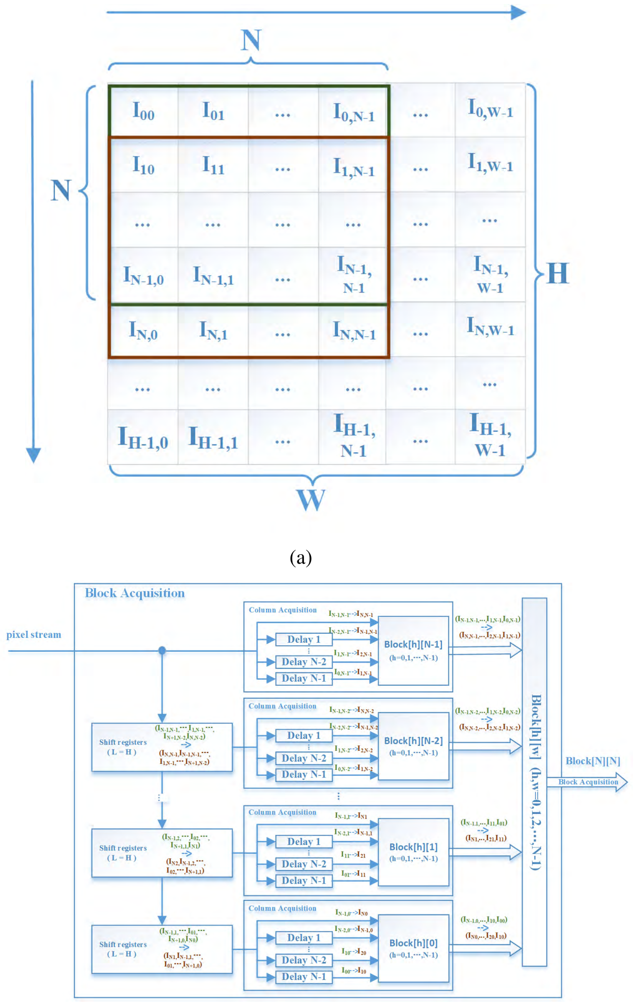


FIGURE 4. High-level architecture of Block matching.

At the beginning, pixel streams are clustered into 2 × 2 blocks in the Pre-processing module. Then the summation and square summation of the blocks are calculated. Note that it is a kind of optimization in order to save FPGA resource and promote the realization of the algorithm. Then the pixel stream is transmitted to the Disparity Control module through an Input FIFO. The Disparity Control module is a crucial part in our work, where the pixel stream are stored into RAM, allocated in the memory of FPGA in advance. Meanwhile, RAM Address Ctrl submodule efficiently provides required blocks to subsequent processing.

The Disparity Generation module is the most computationally intensive part, which calculates the disparity of several fixed blocks between the reference image and the scene image. Then the highest NCC value and the coordinate of the corresponding pixel can be gained. The NCC computation takes most of the time and resources, so efficient mechanism is desired. In the normal processing stage, the Disparity Control and the Disparity Generation process the scene



(a)

(b)

FIGURE 5. Block Acquisition.

image pixel by column. At first, all the pixels of a column are processed in Prediction mode. Then the Arbiter module returns verdicts of these pixels that indicate the prediction is successful or failed, by deciding whether the highest NCC value is higher than a fixed threshold. Once failed, the system switches to Brute-force search mode on these failure pixels until all the pixels on the column have their disparities. Furthermore, all the disparity results obtained from current column will be used in prediction mode to search for the next column.

Finally, Post-processing module outputs the refined disparity. More specifically, WTA submodule is responsible for acquiring the highest correlation value of the disparity on their overlapped pixels, whereas Equalization submodule balances WTA results and outputs the final block matching results.

B. GABOR

As shown in Fig. 6, due to the requirement of FFT, the size of input data should be equal to power of 2. As a result, the Image Expansion module expands the original image to a proper size. For instance, the images with 1280 × 1024 resolution should be expanded to 2048 × 1024. Therefore, a counterpart module, Image Shrink, removes unnecessary intermediate data to recover the image to original resolution. Besides, DDR arbiter contains a logic multiplexer to determine which module controls the DDR bus, and Complex Multiplier is a sub-module to fulfill the operations for complex numbers.

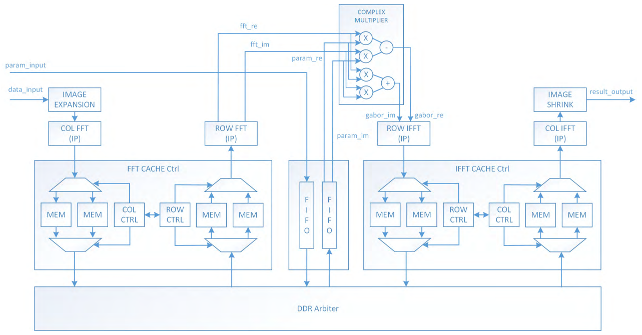


FIGURE 6. High-level architecture of Gabor.

The processing of FFT/IFFT on an image has two sequential steps, first by columns and then by rows on the processed results of columns, or vice versa. The FFT IP provided by Xilinx is highly optimized and resource efficient, which could deal with both FFT and IFFT processing by different input parameters. But each instantiation can only realize a single step, so a complete FFT/IFFT processing needs two instantiations of FFT IP, and the intermediate data should be buffered. However, on-chip high-speed SRAM cannot afford this, for example, a 2048 × 1024 image needs at least 128Mbit memory. So the data has to be buffered on external memory namely DDR3 SDRAM. Although DDR3 IP provides high bandwidth interface and runs at clock frequency up to 900MHz, the read/write performance would degrade significantly since the operation frequency keeps at a high level. As a result, the Fast Caching mechanism integrated in FFT/IFFT Cache Ctrl is well designed to facilitate continuous burst operations.

Fig. 7 explains the principle of Fast Caching. For DDR3, the maximal data width of burst read/write processing is 512-bit, and the data width of a physical row is 1024 × 64-bit. To achieve better performance in accessing DDR3, firstly, each 512-bit burst read/write processing should be in consecutive address in the same row, and secondly a physical row should not be switched until it is filled up. The input stream is the results of rows by FFT/IFFT with 64-bit width. In FPGA, RAM buffers the results of 32 columns, then merges every eight 64-bit elements to a single 512-bit one. As a result, the data of 32 columns is identified as H/32 blocks, and each block has 65,536 bits, exactly equal to the data width of a physical row of DDR3. After the row FFT is completed, each read processing completely loads entire data of a physical row successively to fill up another RAM, in which 512-bit data is read line by line and split into eight 64-bit data to feed column FFT.

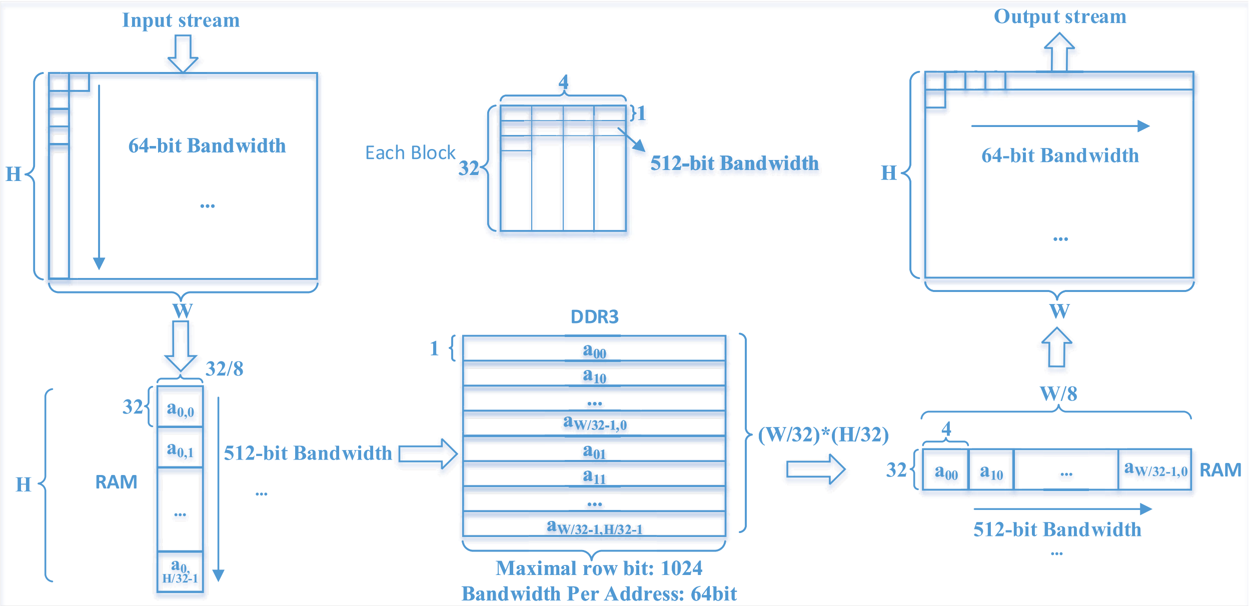


FIGURE 7. Data Flow Diagram of Fast Caching.

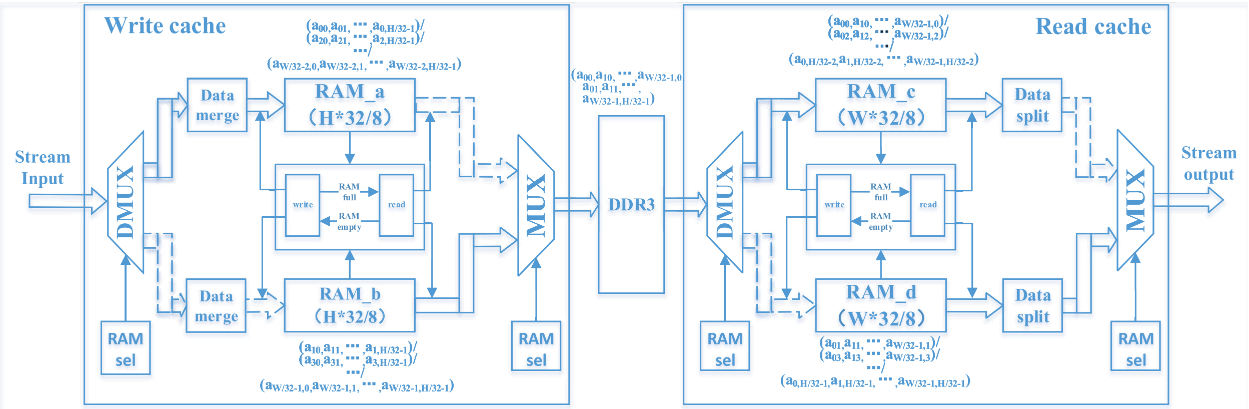

Write cache
Stream Input
DMUX
Data merge
Data merge
RAM_a (H*32/8)
RAM_b (H*32/8)
write
read
RAM full
RAM empty
MUX
RAM sel
RAM sel
$(a_{00},a_{01},\cdots,a_{0,H/32-1})/$
$(a_{20},a_{21},\cdots,a_{2,H/32-1})/$
$\cdots/$
$(a_{W/32-2,0},a_{W/32-2,1},\cdots,a_{W/32-2,H/32-1})$
$(a_{10},a_{11},\cdots,a_{1,H/32-1})/$
$(a_{30},a_{31},\cdots,a_{3,H/32-1})/$
$\cdots/$
$(a_{W/32-1,0},a_{W/32-1,1},\cdots,a_{W/32-1,H/32-1})$
$(a_{00},a_{10},\cdots,a_{W/32-1,0}$
$a_{01},a_{11},\cdots,$
$a_{W/32-1,H/32-1})$
DDR3
Read cache
DMUX
RAM_c (W*32/8)
RAM_d (W*32/8)
write
read
RAM full
RAM empty
Data split
Data split
MUX
Stream output
RAM sel
RAM sel
$(a_{00},a_{10},\cdots,a_{W/32-1,0})/$
$(a_{02},a_{12},\cdots,a_{W/32-1,2})/$
$\cdots/$
$(a_{0,H/32-2},a_{1,H/32-2},\cdots,a_{W/32-1,H/32-2})$
$(a_{01},a_{11},\cdots,a_{W/32-1,1})/$
$(a_{03},a_{13},\cdots,a_{W/32-1,3})/$
$\cdots/$
$(a_{0,H/32-1},a_{1,H/32-1},\cdots,a_{W/32-1,H/32-1})$

FIGURE 8. Modules of Fast Caching.

Fig. 8 shows the implementation details of the module in Fast Caching. A special design needs our concern that Ping-Pang operations are used both in write and read procedures, in order to achieve ideal performance. Therefore, two customized RAMs, DMUX and MUX, are added in write/read procedure, which are switched by RAM Sel module.

IV. EXPERIMENTAL RESULTS

With the elaborate FPGA design as described above, a real-time system is realized with advanced depth accuracy and reconstruction speed. Note that our system uses modulate speckle patterns designed in [16], and the FPGA used in the experiments is Xilinx Virtex-7 XC7VX690T-2FFG1761C embedded in VC709 development board to assure the low cost. Fig.9 shows a snapshot of the implemented system. The test sequences used in experiments are captured with a commodity CMOS camera that is calibrated with the laser emitters in advance. The camera has adjustable resolutions and 640 × 480 and 1280 × 960 are used to testify the processing speed of the system.

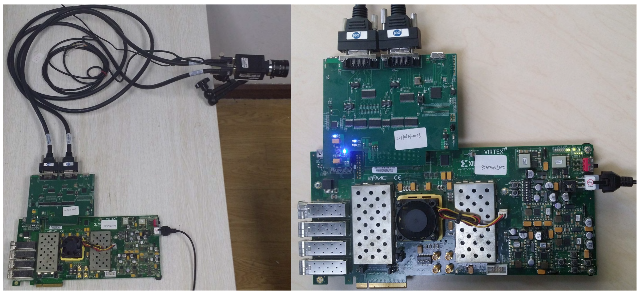


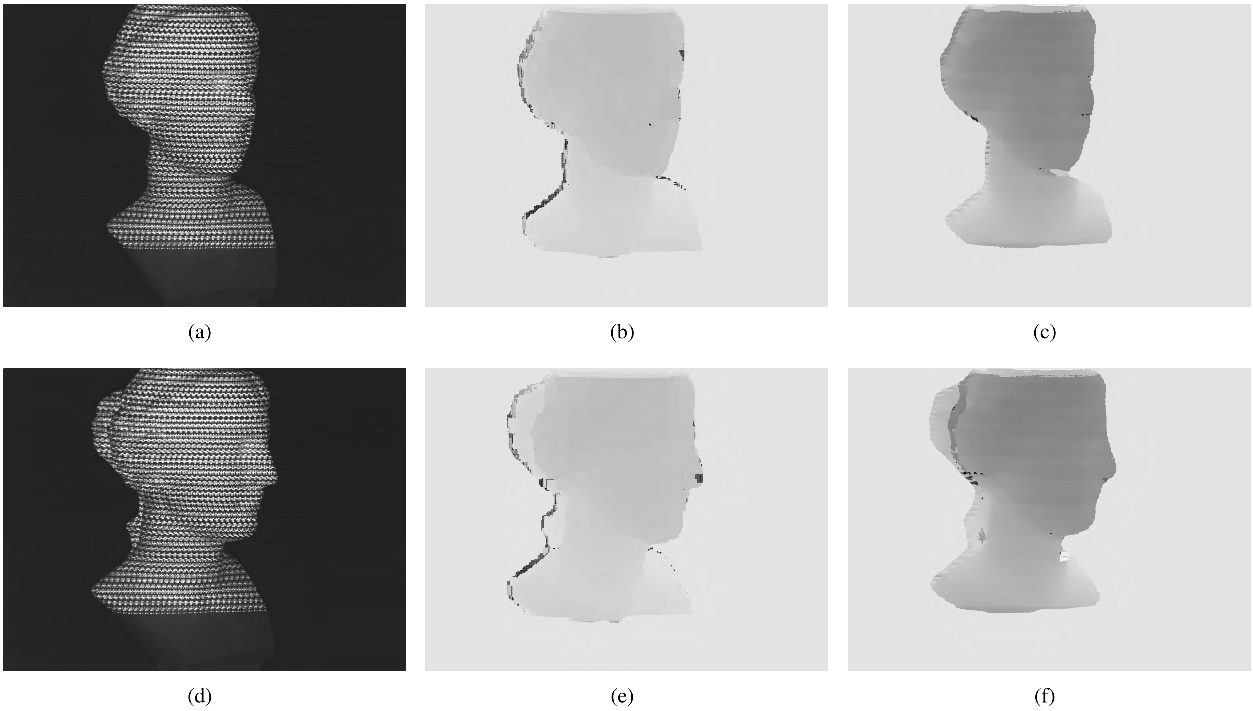
FIGURE 9. Snapshot of proposed system.

TABLE 1. Performance comparison of different resolutions.

Resolution	640×480		1280×960		1920×1080		Resource Available
Maximal Settings	N=24, M=24		N=24, M=15		N=24, M=24		
Performance	356fps		90fps		65fps		
Resource Utilization	Used by System	Used by Core	Used by System	Used by Core	Used by System	Used by Core	
LUT	284,867(66%)	219,448(51%)	223,820(52%)	156,373(36%)	240,844(56%)	174,759(40%)	433,200
BRAM	1,267(86%)	581.5(40%)	1,391(95%)	415.5(28%)	1,444(98%)	468.5(32%)	1,470
DSP48	2,946(82%)	2,749(76%)	1,974(55%)	1,777(49%)	2,274(63%)	2,077(55%)	3,600

TABLE 2. Performance comparison of different implementation methods.

Method	Implementation	FPS
Niitsuma *et al.* 2005 [9]	Xilinx XC2V6000 FPGA	47@1024×1024
Jin *et al.* 2010 [7]	Altera Stratix IV GX DE4	60@640×480
Zhang *et al.* 2010 [8]	Altera EP3SL150 FPGA	60@1024×768
Ttofis *et al.* 2012 [6]	Xilinx Virtex-5 LX110T FPGA	30@640×480
Thomas *et al.* 2013 [10]	Xilinx Virtex-5 XC5VLX110T FPGA	87@1920×1200
Dehnavi *et al.* 2017 [11]	Xilinx Virtex-6 XC6VLX240T FPGA	53@1920×1080
Puglia *et al.* 2017 [12]	Xilinx Zynq-7000 XC7Z020CLG484-1 FPGA	30@1024×768
Proposed	Xilinx Virtex-7 XC7VX690T-2FFG1761C FPGA	356@640×480 90@1280 ×960 65@1920×1080


(a) (b) (c)

(d) (e) (f)

FIGURE 10. 640 × 480 resolution. (a)(d) Scene image. (b)(e) Disparity image by Block Matching with N = 24, M = 24. (c)(f) Disparity image after adding Gabor Filter.

A. PERFORMANCE

On the FPGA board, the designed Disparity IP Core runs at 160 MHz, which generates up to 160-M depth pixels per second. Therefore, in the normal processing stage, the speed upper bound is 520 fps for an image with resolution of 640 × 480, 130 fps for 1280 × 960 and 77 fps

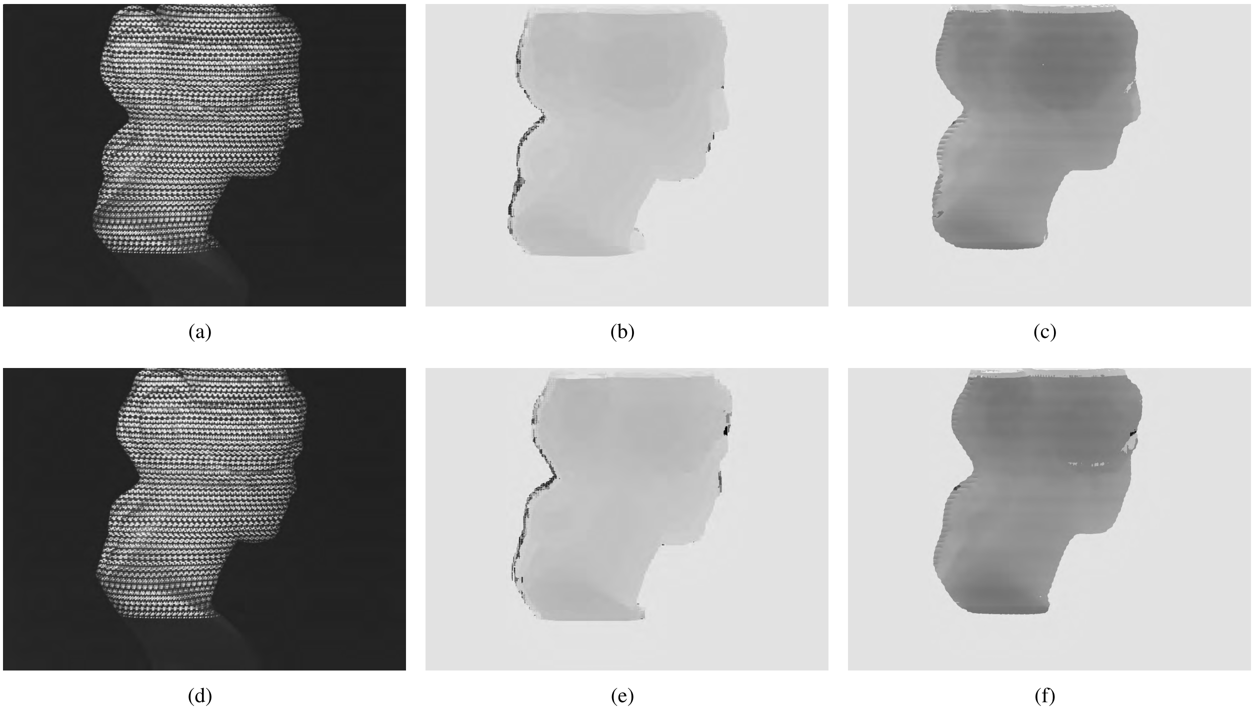

(a) (b) (c)
(d) (e) (f)

FIGURE 11. 1280 × 960 resolution. (a)(d) Scene image. (b)(e) Disparity image by Block Matching with N = 24, M = 15. (c)(f) Disparity image after adding Gabor Filter.

for 1920 × 1080. While the camera could work at up to 90 fps with resolution 640 × 480 and 60 fps with resolution 1280 × 960. In order to evaluate the speed performance as well as resource utilization of the proposed FPGA system, Gabor Filter core is full-pipelined and Block Matching core is nearly full-pipelined, mainly because the Brute-force search is highly resource consumed. Table 1 shows the processing speed of the system, and also gives the resource utilization statics. From the table, we can see that the system gains 356fps@640 × 480, 90fps@1280 × 960 and 65fps@ 1920 × 1080, with corresponding maximal settings of N and M. In addition to image resolutions, there are two most important factors in Block Matching core to determine the resource utilization, which are block size N in calculating NCC and expansion width M in WTA. The other parameters in search are set to fixed values, as shown in Fig. 2.

The resource utilization of the proposed FPGA system and the maximal setting of N/M at different resolutions are presented, as seen in Table 1. From the results, we could find that with an efficient design of the quasi full-pipelined system, the resource utilization of FPGA is acceptable. For example, the utilization of LUTs by the whole system is nearly 60% for the two resolutions. Moreover, LUT and BRAM are also two critical criterions to evaluate the resource utilization. From the results, we could find that BRAM is constrained for the proposed implementation.

For a more complete comparison of different stereo vision systems, the speed performance of different implementations are summarized and listed in Table 2.

B. ACCURACY

To further evaluate the accuracy performance of the proposed FPGA system, different depth sensing results of various resolutions are demonstrated. Fig. 10 shows the depth results of a scenario of a portrait with resolution of 640 × 480, using not only typical Block Matching, but also improved method we implemented. Obviously, the reconstruction quality is improved by our method, where the sketch of head has more details. Under 640 × 480 resolution, both N and M are set to 24. If dealing with larger resolution images, the two parameters have to be set to smaller values in the consideration of FPGA resources. Fig. 11 shows the construction results with resolution 1280 × 960 and we could find that our implemented method obtains better results than typical Block Matching, where N is set to 24 and M is set to 15.

V. CONCLUSION AND DISCUSSION

Stereo vision systems need to extract tridimensional information of objects in a scene. And depth information is one of the most important part of it. Kinect, as a low-cost solution for depth sensing, is very popular and has been widely used in diverse applications. It uses active structured light illumination and laser pattern generator, and has a better accuracy over the other depth sensing methods.

Based on the principle of Kinect sensing methodology, a one-shot depth acquisition system based on FPGA is proposed in this paper. The method implemented is advanced and could acquire high-quality depth results, where Gabor Filter is applied to improve the original Block Matching.

With an elaborate hardware architecture, an improved depth accuracy is achieved with high reconstruction speed (90fps@ 1280 × 960) on a normal FPGA, over other implementation methods. The system we proposed could not only be used on visual processing, but also has a wide application in all kinds of fields when block matching is needed.

REFERENCES

[1] K. Jawed, J. Morris, T. Khan, and G. Gimelfarb, "Real time rectification for stereo correspondence," in *Proc. Int. Conf. Comput. Sci. Eng.*, Vancouver, BC, Canada, Aug. 2009, pp. 277–284.

[2] Q. Du, X. Shi, B. Dai, and E. Lu, "Binocular stereo vision system for a humanoid robot," *Int. J. Comput. Appl. Technol.*, vol. 46, no. 4, pp. 316–322, Apr. 2013.

[3] Z. Wang, F. Zhang, and Y. Quan, "On-machine 3D reconstruction using monocular stereo vision system," in *Proc. ISISE*, Austin, TX, USA, Dec. 2009, pp. 392–395.

[4] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Int. J. Comput. Vis.*, vol. 47, nos. 1–3, pp. 7–42, Apr. 2002.

[5] R. A. Hamzah and H. Ibrahim, "Literature survey on stereo vision disparity map algorithms," *J. Sensors*, vol. 2016, Nov. 2016, Art. no. 8742920.

[6] C. Ttofis and T. Theocharides, "Towards accurate hardware stereo correspondence: A real-time FPGA implementation of a segmentation-based adaptive support weight algorithm," in *Proc. DATE*, Dresden, Germany, Mar. 2012, pp. 703–708.

[7] N. Isakova, S. Başak, and A. C. Sönmez, "FPGA design and implementation of a real-time stereo vision system," in *Proc. Int. Symp. Innov. Intell. Syst. Appl.*, Trabzon, Turkey, Jul. 2012, pp. 15–26.

[8] L. Zhang, "Design and implementation of real-time high-defnition stereo matching SoC on FPGA," M.S. thesis, Dept. Microelectron. Comput. Eng., Delft Univ. Technol., Delft, The Netherlands, 2010.

[9] H. Niitsuma and T. Maruyama, "High speed computation of the optical flow," in *Proc. ICIAP*, Cagliari, Italy, 2005, pp. 287–295.

[10] S. Thomas, K. Papadimitriou, and A. Dollas, "Architecture and implementation of real-time 3D stereo vision on a Xilinx FPGA," in *Proc. VLSI-SoC*, Istanbul, Turkey, Oct. 2013, pp. 186–191.

[11] M. Dehnavi and M. Eshghi, "FPGA based real-time on-road stereo vision system," *J. Syst. Archit.*, vol. 81, pp. 32–43, Nov. 2017.

[12] L. Puglia, M. Vigliar, and G. Raiconi, "Real-time low-power FPGA architecture for stereo vision," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 64, no. 11, pp. 1307–1311, Nov. 2017.

[13] J. Wang, Z. Xiong, Z. Wang, Y. Zhang, and F. Wu, "FPGA design and implementation of Kinect-like depth sensing," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 6, pp. 1175–1186, Jun. 2016.

[14] I. Nahhas, "Fast computational four-neighborhood search algorithm for block matching motion estimation," *Int. J. Eng. Res. Appl.*, vol. 5, pp. 20–24, Jan. 2015.

[15] Y.-Y. Sun, X.-H. He, H.-Y. Song, and W.-L. Chen, "A block-matching image registration algorithm for video super-resolution reconstruction," *Acta Autom. Sinica*, vol. 37, no. 1, pp. 37–43, Jan. 2011.

[16] L. Qin *et al.*, "One-shot depth acquisition with a random binary pattern," *Appl. Opt.*, vol. 53, no. 30, pp. 7095–7102, Oct. 2014.

[17] S. Satoh, "Simple low-dimensional features approximating NCC-based image matching," *Pattern Recognit. Lett.*, vol. 32, no. 14, pp. 1902–1911, Oct. 2011.

[18] A. Ahmmed, "Image denoising using Gabor filter banks," in *Proc. IEEE Symp. Comput. Inform.*, Kuala Lumpur, Malaysia, Mar. 2011, pp. 215–218.

[19] P. Karas and D. Svoboda, "Convolution of large 3D images on GPU and its decomposition," *EURASIP J. Adv. Signal Process.*, vol. 2011, no. 1, p. 120, Dec. 2011.

ZHEN-NI LI was born in 1982. She received the Engineering and M.S. degrees in electronics from ENSEEIHT, France, in 2008, and the M.S. degree from Beihang University in 2009, and the Ph.D. degree in detection technology and automatic equipment from Northeastern University, Shenyang, China, in 2018.

She is currently a Lecturer with Northeastern University. Her research interests include embedded system and video processing, asynchronous NoC design, and intelligent perception.

YU-LIANG GAO received the B.S. degree in electronic science and technology from Northeastern University, Shenyang, China, in 2017, where he is currently pursuing the master's degree in circuits and systems.

His research interests include hardware computing, wireless communications, and visible light communications.

ZE-KUN WANG received the M.S. degree in circuits and systems from Northeastern University, Shenyang, China, in 2014.

He is currently an IC Design Engineer with Broadcom Corporation.

JIAO WANG received the Ph.D. degree in pattern recognition and intelligent systems from Northeastern University, Shenyang, China, in 2006.

He is currently a Professor with the College of Information Science and Engineering, Northeastern University. His research interests include hardware computing and computer games.