FPGA Implementation of Disparity Estimation Proceesing Architecture for Stereo Camera System

Hi-Seok Kim
dept. electronics engineering
Cheongju University
Cheongju, KOREA
khs8391@cju.ac.kr

Young-Hwan Kim, Sea-Ho Kim, Choong-Mo Youn
dept. electronics engineering
Pohang University, Cheongju University, Seoil University
Pohang, KOREA, Cheongju, KOREA, Seoul, KOREA
youngk@postech.ac.kr, kensean@cju.ac.kr, 5420chong@seoil.ac.kr

***Abstract*—With the advance of image processing and computer vision, the stereo vision system with two cameras has become the research of interest in many areas since its ability to realize the depth information is similar to human vision. Depth map algorithm allows camera system to estimate depth. It is a computation intensive algorithm, but can be implemented with high speed on hardware due to the parallelism property. In this paper, by analyzing digital image stabilization (DIS) algorithms, we propose an efficient disparity estimation architecture, which combines gray–scaled projection and Affine transformation model. We develop the architecture by describing the various computation units in hardware description language (Verilog) and synthesizing the design into a FPGA. The synthesis and experimental results for three video test images show that the proposed hardwired architecture is better than traditional sum of absolute difference (SAD) architecture, which based on block matching algorithm in terms of frame rate (frame/sec) while keeping the competitive PSNR results.**

Keywords -Gray-scale projection; Steroscopic; architecture; 3D

I. INTRODUCTION

Recently, industrial demand and interest of stereoscopic image systems are increased due to 3D movies and HD -TV. The depth information is the main element in 3D image systems. Stereo matching or disparity estimation is exploited to find the depth information from stereoscopic images. The goals of this paper are to propose a real-time processing architecture for disparity estimation and to show application systems based on real-time stereo camera system. The proposed system operates in the FPGA board environment with stereo camera.

It has potential uses in robotic navigation, 3D imaging, camera surveillance and object recognition systems. A typical depth estimation system consists of two cameras with overlapping field of view and a processing unit. To estimate the depth, several depth-map algorithms have been developed [1]. The idea is to find the displacement between two projections of the same object in the two images. From that, the distance is calculated based on the relative position of the two cameras and other dimensions such as focal length, angle between optical axes. The depth value is represented in the result image as pixel intensity. The depth-map algorithm is also called the disparity algorithm. The idea of a depth-map system is quite simple. However, the depth-map algorithm is computational and data intensive [2] because it has to perform an identical procedure on millions of pixel. Due to the computational complexity of the disparity algorithms, several attempts for video 3D tracking have been developed in recent years [3]. several attempts have been made[3-5], including systems implemented on personal computer, digital signal processor (DSP), field programmable gate array (FPGA) and application specific integrated circuit (ASIC). One of these attempts presented in 3D feature tracking and localization using stereo vision systems. The objective of feature localization is to localize the corresponding feature point in the right video sequence. Because the motion of the given feature point in the left and right video sequences is similar, this system is designed to obtain the motion vectors from acquired image frame of the left video sequence and to estimate the corresponding motion vectors for the feature in the right video sequence.

Various algorithms, such as projection algorithm (PA), bit-plane matching (BPM), and others, have been developed to estimate the motion vectors. In general, the gray scale projection algorithm can greatly reduce the complexity of computation in comparison with the other methods. In this paper, our focus is to develop an efficient architecture based on the motion vectors of the gray scale projection and Affine transformation model for practical implementation of 3D image processing.

This paper is organized as follows. Section II describes the algorithm of gray-scale projection and estimate motion vectors. Section III describes the design of the proposed architecture. Experimental results are shown in Section IV. Finally, the conclusion is given in section V.

II. GRAY SCALE PROJECTION

The gray-scale projection is an approach based on total gray-scale changes in the coordinate of an image to estimate Motion vectors between the current and reference frames. By doing a related operation using gray-scale projection algorithm, we can determine motion vector. A small amount of computation is one of the valuable characteristics of this algorithm. Normally, the gray-scale projection algorithm can

 ISBN: 978-1-61208-302-5

be divided into three steps: Image projection, projector filter, and its correlation image projection.

The gray scale projection uses gray information of image to compute the correlation between two frames. Its projection is divided into two one-dimensional wave shaped plane, these formulas are described as follows:

$$P_{Vk}(i) = \sum_{j=1}^{N} Y(i, j) \qquad (1)$$
$$P_{Hk}(j) = \sum_{j=1}^{M} Y(i, j)$$

In which, Y(I,j) is the gray value of point(i, j) at image frame k , Pv_k(i) is the vertical I projection at frame k, $P_{Hk}(j)$ is the horizontal j projection at frame k ,the size of frame is M×N.

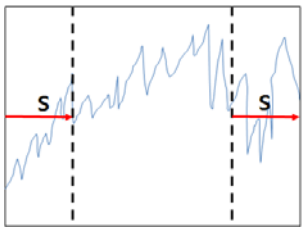

Figure 1. Results of horizontal and vertical gray projection curve at frame k.

Figure 1 shows results of the horizontal and vertical projection at frame k. S denotes the invalid width of search between the current frame and the reference frame.

A. *Correlation calculation*

In order to estimate the motion vectors, we can perform the correlated operations with vertical gray-scale projection curves [4] of left image frame and horizontal gray-scale projection curves of right image frame respectively. Thereby, we can obtain two cross-correlation curves. According to the local minimum value of two curves, we can get a motion vector between left image and right image [5]. The correlation computing formulas are described as follows:

$$C_V(w) = \sum_{i=1}^{M-2S+1} |P_V^{I_1}(i+w-1) - P_V^{I_2}(i+S)|, (1 \le w \le 2S+1) \qquad (2)$$
$$C_H(w) = \sum_{i=1}^{M-2S+1} |P_H^{I_1}(i+w-1) - P_H^{I_2}(i+S)|, (1 \le w \le 2S+1) \qquad (3)$$

In which, $P_V^{I_1}(i+w-1)$, $P_H^{I_1}(i+s)$ are the vertical and horizontal projection curves of the left frame and $P_V^{I_2}(i+w-1)$, $P_H^{I_2}(i+s)$ are the vertical and horizontal projection curves of the right frame . M, N is the vertical rows and horizontal columns at the left frame 1 and the right frame 2. The minimum of $C_V(w)$ is W_V^{MIN} and that of $C_H(w)$ is W_H^{MIN}, respectively. Then, we can get the motion vectors from the formulas (4) and (5):

$$Tx = S + 1 - W_H^{MIN} \qquad (4)$$
$$Ty = S + 1 - W_V^{MIN} \qquad (5)$$

Tx and Ty are the translation (motion) vectors, denoted as offset-x and offset-y. Consequently, we can perform compensation operation to the right frame using two offset values (Tx and Ty).

B. *AFFINE TRANSFORMATION MODEL*

The change of video image can be divided into translation and rotation. Often, we choose the Affine transformation model. In this paper, we will consider changes of rotation and translation. The angle variable of θ is the rotation parameter in Affine model. From the given left image and right image frame, by adopting Sobel gradient operator [6], we can compute the angle of θ which is rotation parameter at Y(i,j) in the left image frame and right image frame respectively. The two 3x3 templates are used by Sobel gradient operator. Every gray value in the image frame should use these two templates to do convolution. One of the two templates has a maximum response to the vertical edge and the other has a maximum response to the horizontal edge. Then, the Sobel operator can compute the vertical and horizontal edge orientations. We can get the angle orientations for the corresponding formula as follows:

$$\theta = \tan^{-1}\left(\frac{H_V}{H_H}\right) \qquad (6)$$

Here, H_V and H_H are the vertical and horizontal edge orientations. The size of the image frame is M x N.

III. ARCHITECTURE AND DESIGN

In this section, we develop an efficient hardware architecture based on the gray-scale projection algorithm and Affine transformation model. In particular, we focus on developing the architecture for stereoscopic image processing. The flow diagram of the computation for luminance (Y) component is shown in Fig. 2. The image processing system is designed to take an 8-bit gray-scale left and right input images. We note that the proposed stereoscopic image processing system can be applied equally well to 8-bit images by simply performing the luminance (Y) operation on the image while skipping the other color format operation. In Figure 2, we need to compute the edge orientation of each pixel of the left and right images at Y(i,j). We set up a table in RAM whose dimensions are equal to input images. Each table entry contains the computed angle orientation for the corresponding pixel of the input images at Y(i,j). Hence, we can calculate the histogram of the angel orientation. Subsequently, we find dominant angle orientation from the histogram. Let $H_{\max}$ denote the number of pixels that has dominant angle orientation in the left and right image

 ISBN: 978-1-61208-302-5

frames. If $H_{\max}$ >threshold, then we can obtain the current frame. Then, oriented with direction, $H_{\max}$ will be set as the dominant angle orientation. After fixing the dominant angle orientation in the left and right current frames, we can get the difference angle orientation as follows:

$$\Delta\theta(m,n) = |\theta_p(m) - \theta_p(n)| \quad (7)$$

Here, $\theta_p(m)$ and $\theta_p(n)$ are the left and right dominant angle orientations of the current frames, where m, n is the image sequence of the left and right frames (m, n = 0, 1, …, N).

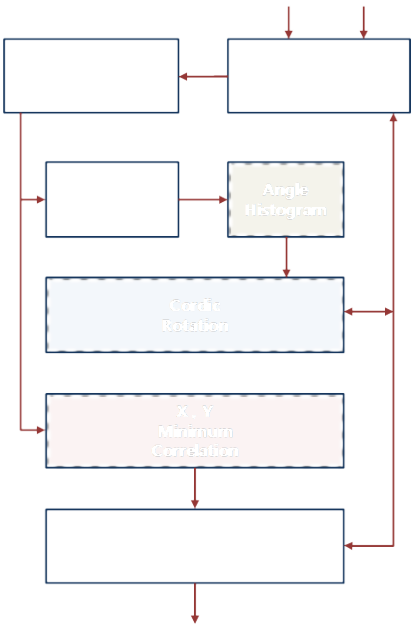

Figure 2. Data flow graph of computation required in motion and rotation vector

After $\Delta\theta(m,n)$ has been computed, we can compute the corresponding difference dominant orientation angle by adopting the CORDIC computing technique [7].

Computations of the rotations are performed using an additional angle accumulator, which is given by;

$$Z_i + 1 = Z_i - d_i \tan^{-1}(2^{-i}) \quad (8)$$

where, d_i = -1 if Z_i <0, otherwise d_i = +1

In particular, we notice that a small look up table (one entry per iteration) containing the $\tan^{-1}(2^{-i})$ is required for the angle computation. Then, we can get the angle orientation of θ, which is the rotation parameter of Affine transformation model described by section II. So the transformation model can be divided into two parameters of the Affine motion model described in the following formula:

$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} + \begin{bmatrix} T_X \\ T_y \end{bmatrix} \quad (9)$$

Here, X , Y and X' , Y' are the pixel points in the left image frame and the right image frame respectively. The $\cos\theta, \sin\theta$ are the rotation parameter. T_x is the final horizontal direction motion vector and T_y is the final vertical direction motion vector. After T_x and T_y have been computed, we compensate the current right image frame as shown in Figure. 2. Then, we can achieve the stereoscopic image process.

IV. EXPERIMENTAL RESULTS

In this paper, we respectively compare our proposed architecture with the traditional computing works written by C^{++} language. The proposed architecture is tested for 6 video sequences. Then, we select one frame image in video sequence as an experimental data, shown as Figure 3(a) and Figure 3(b).

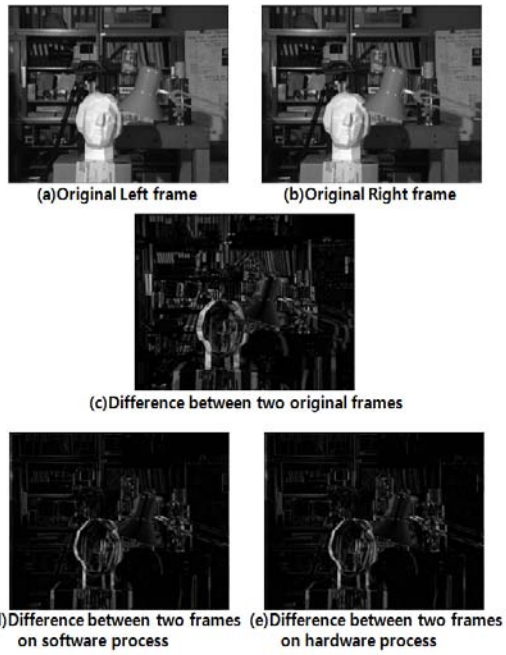

Figure 3. Original Left, Right Image and Result Image (tsukuba)

Similarity can be evaluated by examining the difference between two images. We can get rid of background as well using the difference image. If two frames have movements, the difference image is white. If two images are the same, the difference image is completely black. Figure 3(d) shows the difference image between the left frame and right frame derived from proposed algorithm written in C^{++} language. Similarly, Figure 3(e) shows the difference image between the left frame 101 and right frame 101, which we implement the proposed algorithm with hardware architecture. Comparing Figure 3(d) and 3(e), the results show almost

 ISBN: 978-1-61208-302-5

identical performance. Several blocks of our proposed algorithm are implemented in Xilinx Spartan -3 FPGA and their parameters are shown in table I. We processed four different videos adopting our new architecture. From table I, we can see that our proposed architecture can achieve real-time processing for various large size of videos. In Table II, comparison between the proposed architecture and other systems implemented by previous authors [8] are shown. From table II, we can see that our proposed architecture has faster speed (frame/s) than SAD based block matching architecture. We can't utilize the identical Xilinx FPGA implemented by D. Chaikalis, et al since our proposed system uses the internal memory block in Xilinx FPGA. Table III shows the resource usage for our proposed system.

TABLE I. HARDWARE IMPLEMENTATION DATA OF VIDEO

FPGA Device	Max (MHz) Frequency	Image Size	Total clock number	Frame /sec
Spartan-3 xc3a5000	51.287	380 x 340	386212	132
		400 x 400	478412	106
		640 x 480	919532	55
		1024 x 768	2355980	21

TABLE II. COMPARISON BETWEEN OUR ARCHITECTURE AND OTHER SYSTEM

Author	Algorithm Used	Platform Used	Frame Size	Frame /sec
Proposed System	- Gray-Scale Projection - Affine Transformation Model	Xilinx Spartan-3 xc3a5000	640 x 480	55
			1024 x 768	39
N.H. Tan ,et al	SAD	Altera DE2-70 Cyclone II	640 x 480	35
D. Chaikalis, et al	SAD	Xilinx Virtex XCV-2000E	1024 x 768	31

TABLE III. RESOURCE USAGE OF OUR SYSTEM

Logic Utilization	Used	Utilization
Number of Slices	769	2%
Number of Slice Flip Flops	691	1%
Number of 4 input LUTs	1354	2%
Number of bonded IOBs	157	24%
Number of BRAMs	6	5%

From table I and III, area/throughput estimates based on the synthesis results of a Verilog description of this architecture will be provided to show the feasibility of a single chip ASIC implementation. The peak signal-to-noise ratio (PSNR) between the stabilized frames is an important criterion to evaluate the fidelity of the DIS. The PSNR gives the relation between two frames in terms of their powers.

The higher the PSNR, the better is the fidelity of the DIS. The PSNR between left frame and right frame is defined as [4].

TABLE IV. HARDWARE IMPLEMENTATION DATA OF VIDEO

Image	Size	Frame /sec	PSNR	
Image 1 (tsukuba)	640x480	55	Original Left Image / Original Right Image	17.25
			Original Left Image / S/W Process	20.43
			Original Left Image/ H/W Process	20.41
			S/W Process / H/W Process	43.56
Image 2 (venus)	640x480	55	Original Left Image, Original Right Image	17.39
			Original Left Image, S/W Process	17.42
			Original Left Image, H/W Process	17.80
			S/W Process, H/W Process	20.49
Image 3 (teddy)	640x480	55	Original Left Image, Original Right Image	14.18
			Original Left Image, S/W Process	15.97
			Original Left Image, H/W Process	16.01
			S/W Process, H/W Process	19.06

Table IV shows the PSNR results for three video test images proposed by our hardwired architecture. In comparing three video images(S/W process right image) processed by DIS algorithm written in C^{++} with the one processed by our hardware architecture, our proposed hardware algorithm shows competitive value of PSNR with less computation time. The gray-scale projection algorithm is known to be inferior because of rough compensation. However, we can take this strategy in favor of reducing the amount of computation time, although there is a disadvantage in terms of the accuracy.

V. CONCLUSION AND FUTURE WORKS

In this paper, we proposed an efficient architecture, which combine gray-scale projection and Affine transformation model. The proposed architecture achieves real time processing speed of more than 30 fps. It is proved that implementation of the stereoscopic image processing system is feasible with the proposed architecture. For further work, it is recommended that our system is investigated more with other algorithm such as 3D tracking and depth Map algorithm.

ACKNOWLEDGMENT

This work was sponsored by ETRI SW-SoC R&BD Center, Human Resource Development Project.

 ISBN: 978-1-61208-302-5

REFERENCES

[1] Li-Wei Zheng, Yuan-Hsiang Chang, Zhen-Zhong Li. “A study of 3D feature tracking and localization using a stereo vision system”, Computer Symposium (ICS), Dec. 2010, pp. 402-407.

[2] Zhu Juan-juan, Guo Bao-long, Feng Zong-zhe. “A digital image stability algorithm based on the Gray-scale projection”. photon Journal, Oct. 2005, pp. 1266-1269.

[3] F. Vella , A.Castorina , M. Mancso et al.. “Digital image stabilization by adaptive block motion vectors filtering”. IEEE Transactions on Consumer Electronics, Aug. 2002, pp. 796-801.

[4] I.Yasri, N.H.Hamid, V.V.Yap, “Performance analysis of FPGA based Sobel edge detection operator”, Electronic Design, 2008. ICED 2008. International Conference on, Dec. 2008, pp. 1-4.

[5] P.K.Meher, S.Y.Park, “CORDIC Designs for Fixed Angle of Rotation”, Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, Feb. 2012, pp. 1-12.

[6] Ngo Hun Tan,Nor Hisham Hamid, Patrick Sebatian. “Resource Minimization in a Real-Time Depth-map Processing System on FPGA”, TENCON 2011 - 2011 IEEE Region 10 Conference, Nov. 2011, pp. 706-710.

[7] Angelos A. Amanatiadis, and Ioannis Andreadis, “Digital Image Stabilization by Independent Component Analysis,” IEEE Transactions on instrumentation and measurement, Vol. 59, No. 7, July. 2010, pp. 1755-1763.

[8] Masanori Hariyama and Michitaka Kameyama, “VLSI Processor for Re-liable Stereo Matching Based on Window-Parallel Logic-in-Memory Architecture”, Digest of Technical Paper 2004 Symposium on VLSI Circuits VLSI Symposium, June. 2004, pp. 166-16.

This work was sponsored by ETRI SW-SoC R&BD Center, Human Resource Development Project.