Proceedings of The 2016 IEEE International
Conference on Real-time Computing and Robotics
June 6-9, 2016, Angkor Wat, Cambodia

Real-time Implementation of Panoramic Mosaic Camera based on FPGA

Weiguo Zhou*, Yunhui Liu†*,*Fellow,IEEE*, Congyi Lyu‡†,
Weihua Zhou§†, Jianqing Peng*, Ruijia Yang*, Haiyang Shang*
*School of Mechanical Engineering and Automation, Harbin Institute of Technology Shenzhen
Graduate School, Shenzhen, China. Email: zhouweiguo1991@gmail.com
†Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong
‡School of Mechatronical Engineering, Beijing Institute of Technology, Beijing, China
§Shool of Automation, Northwestern Polytechnical University, Xi'an, China

***Abstract*—With regard to Automated Guided Vehicles (AGVs) and autonomous navigation of Micro Aerial Vehicles (MAVs) in complicated environments, it is vital to detect not only in the direction of ongoing but also the entire environment. It is very significant for obtaining a broader view of a scene than the current single view which has been used in a wide range of applications such as satellite imaging, medical imaging and so on.**

As the conventional omnidirectional camera based on PC is cumbersome and power consuming, then a novel light-weight structure comprising of four cameras to detect targets in all directions based on FPGA is presented in our design.

We will discuss the implementation of image mosaic algorithm which is comprised of image registration and image fusion and some other image preprocessing algorithms for better effect such as median filter algorithm, color filter algorithm, image enhancement algorithm, etc., on the Xilinx Zynq-7020 FPGA device using Vivado Design Suite and Software Development Kit which is embedded on ZedBoard developed board. The system is able to handle more than 60 frames per second (fps) freely and still in a low power consuming.

I. Introduction

Images contain a lot of information which are widely used in many applications such as robotics [1], military [2], medicine [3] and surveillance [4] [5] etc. Many overlapped views can be combined to produced a single image with a much broader field [6] to obtain more information for analysis. In such cases, image mosaic technique [7] which is comprised of image registration and image fusion will be used to combine those overlapped views. And it becomes the focus of the robot vision for its widely used in many domains such as space exploration, remote sensing image, medical image analysis and so on [8]. Omnidirectional camera can help robot perceive a $360-$ degree view but it need to be accomplished in real time, which is exactly the theme of our International Conference on Real-time Computing and Robotic (RCAR).

There are plenty of vision algorithm implemented on PC based on MATLAB [9], OpenCV [10], etc. However, it is very clumsy, power consuming and low compute performance to utilize PC in the real applications. With the increase in resolution and frame rate in recent sensors, it is hard for DSP and GPU to achieve the processing process. The algorithms implemented on Field Programmable Gate Arrays (FPGAs) can operate at a low frequency but maintain a high compute performance. In addition, it is very flexible for the FPGA to be implemented different algorithms depends on the specific applications. Due to such advantages, FPGA has been widely noticed as an embedded platform such as smart surveillance cameras and vehicular cameras [11] [12] [13] [14].

Thus, We presented a image mosaic algorithms based on SAD (Sum of Absolute Difference) [15] implemented on the Zedboard developed board from Xilix company in this paper. Four broad view image sensors (OV5640) are used to comprise the omnidirectional view.

This paper is structured as follows. In section II the overall embedded vision systems is introduced and the hardware employed in our work is discussed. In section III, we focus on the image mosaic algorithm based on the sum of absolute differences and its method of implemented on the FPGA. And the processes to create a mosaiced image on FPGA will be illustrated in section IV and some implemented results are presented. At the end, we will give the conclusion and some future work (section V).

II. Embedded Vision Systems

In the following section, We describe the algorithms implemented for image mosaic on embedded systems. Fig. 1 demonstrates the standalone embedded vision systems. The images from image sensors are processed in real-time by the processing module and the results are displayed on-line via the VideoOut Module on high resolution LCD screen. Commonly, The processing module is composed of video processing flow which is a streaming processing that sequentially applies several processes for each image (Fig. 1). The images are stored in frame-buffer outside the FPGA chip and several types of algorithms are used to process them.

In some ways, FPGA is very suitable for image processing [16] [17]. Most of the vision algorithms are computationally intensive, so it is wise to implemented them on the embedded systems for their parallel processing ability. Another obvious advantage of the FPGA-based system is that the hardware and software components of the system are field re-programmable. We can single facilitate the development of general-purpose processing systems without any hardware changes.

978-1-4673-8959-4/16/$31.00 © 2016 IEEE

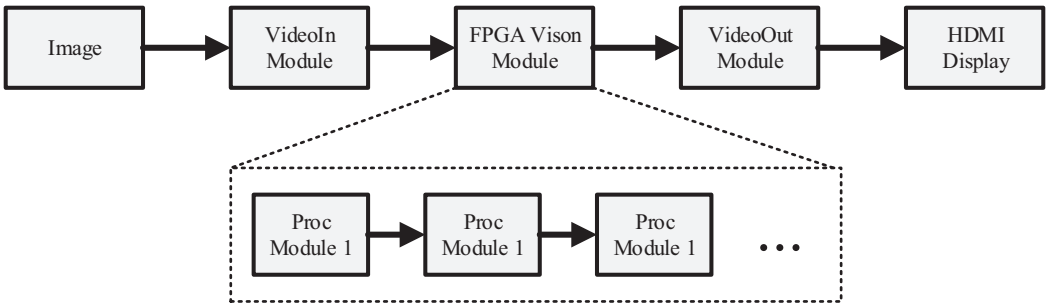

Fig. 1. Standalone embedded vision system.

The Dual-core ARM 9 processors and FPGA work independently in Zynq 7000 SoC. Zynq has two sections called a PS (Processing System) and PL (Programmable Logic). Development of the hardware system involving designing peripheral blocks and other logic resources to be implemented in the PL part. The PS is a section for the dual-core CPU, and the PL is a section for the FPGA. External 256 MB DDR3 memory is implemented and connected to the PS part.

In the PL system, the signals from the four sensors are connected to each peripheral which supports AXI4(Advanced eXtensible Interface 4) interface for data transmission. The registers of the image sensors are configured via AXI IIC provided by Xilinx while the other modules are coded in Verilog HDL.

1) Overall Design: The VideoIn module acquires the data from the four image sensors which provide four-channel parallel data in raw format. The frame rate is 30fps at VGA (640×480) resolution.

The VideoIn module moves the stream of images to the external memory (DDR3 SDRAM) so that PS can directly access the memory region for further image processing which is illustrated in Fig. 2. In addition, the VideoIn module also sends the start and end signal of the image frame to PS to notify whether it is time for reading image. FPGA received the data from the senor after the initialization via IIC communication protocol of the OV5640 sensors. The original data cannot be directly displayed on the LCD display for the different operating clock domain. Then data cache is completed through the external memory which can be illustrated in Fig. 2.

2) Image Sensor: We select OV05640-A71A color CMOS QSXGA (5 megapixel) image sensor from OmniVision technology which is widely used in many applications such as cellular phones, PC multimedia, robot vision ,etc. The sensor supports for output formats include RAW RGB, RGB565/555/444, CCIR656, YUV422/420, YCbCr422 and compression. The maximum transfer rate: 15fps at QSXGA (2592×1944), 30fps at 1080p, 90fps at VGA (640×480). Image acquisition module includes two parts. Part one receives pixel data and control signals which generated by the camera, then convert the signals into AXI-Stream formate. The second part is is the module with regards to registers configuration. The connections between OV5640 and FPGA is illustrated in Fig. 3. The base address of the OV5640 is 0x78. The camera

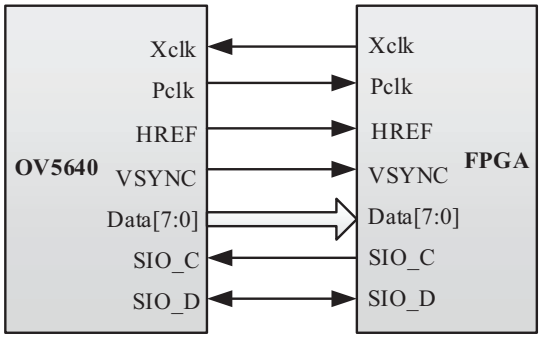

Fig. 3. Signals communicate between the OV5640 image sensor and FPGA chip

is configured via I2C protocol through SIO_C and SIO_D signals. Linear differential method is used to recover RGB from the original bayer array.

III. Image Mosaic Algorithm

The proposed mosaicing algorithm is based on Sum of the Absolute Differences (SAD) by finding the the minimum value of the corresponding area of the two images matched. Actually, there are several other algorithms and architectures that can be adopted for image mosaicing such as Harris corner detector, SIFT or SURF detectors/descriptors. Then the algorithm is intend to be implemented on embedded system to be accelerate the processing speed.

The following diagram in Fig.4 shows the overview of the image mosaic algorithm using SAD. The architect has four cameras inputs including Pclk as the input pixel clock, HREF and VSYNC as the control signals, Data[7:0] are the pixel data of the cmos at Raw8 format. The images processed in our paper is at 640×480 pixel resolution and the architecture of all the individual modules are shown in detail in the following subsections.

Sum of the Absolute Differences is a pixel-based matching method which is the method that obtain the corresponding matched position by comparing a group of pixels called a window from one image in another image. The SAD algorithm, showing in Equation 1, taking the absolute difference between the two pixel value of the window which generated by the following subsection buffer module part to get a SAD result

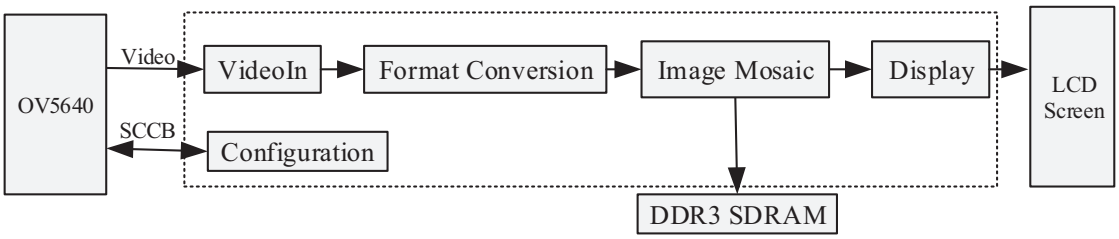

Fig. 2. Diagram of the overall structure of the data stream

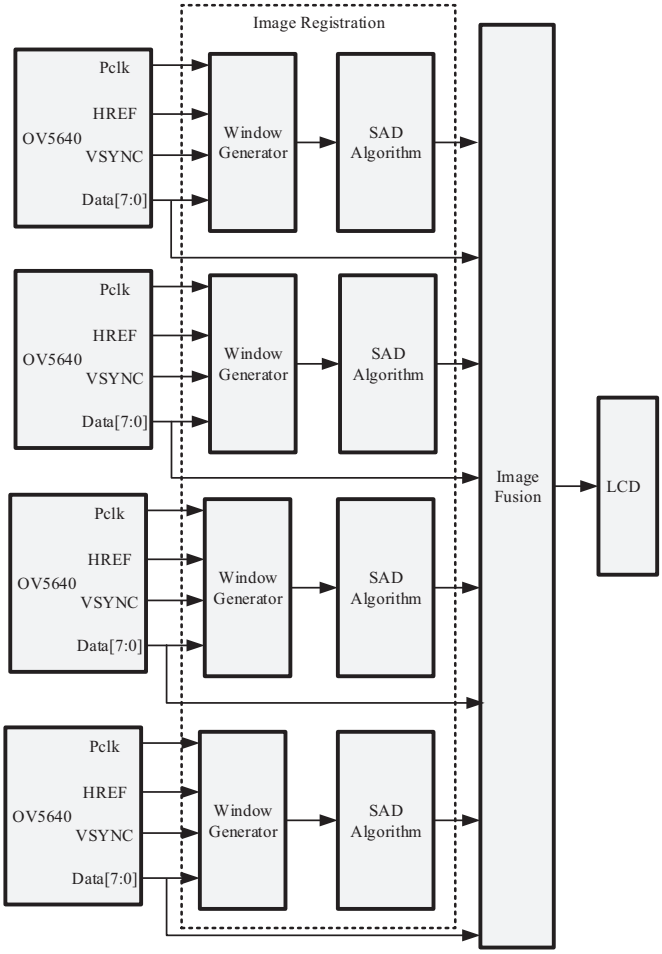

Fig. 4. Overview of the image mosaic algorithm using SAD

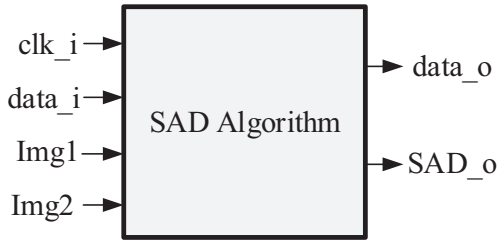

Fig. 5. Top level entity of the SAD implementation

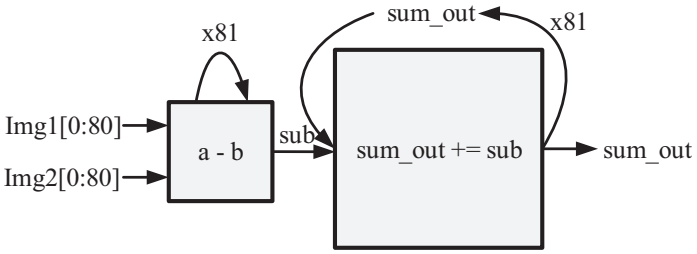

Fig. 6. Simplified version of the process SAD

value. Position can be found when we obtain the minimum value of the SAD within a constant of a frame of image.

SAD is one of the commonest measures to calculate the absolute differences between two windows by subtracting pixels between the first Img1 and the other Img2 followed by using the Equation 1 within the square window which used in our design is 9×9 square window. Figure 5 illustrates the top level entity of the our SAD implementation used. This module has a clocked input (clk_i) signal and a 1 bit data input (data_i) to tell the algorithm to begin calculating the equation. The Img1 and Img2 will send 81 (9×9) bytes to the SAD Algorithm entity. The 1 bit data output signal (data_o) indicates whether the calculation is completed and ready for the next set of input. The calculated result of SAD value is sent out through SAD_o. In our paper, there is a slight difference between SAD algorithm we used and the standard one. Instead of subtracting two pixel values and taking the absolute difference between them, these implementations in this paper will firstly find the greater one and then send the two value to the subtracter module. The subtracter will complete the subtraction between the two arguments and return the difference, sub, which is greater or equal to zero. By this method, allows us to get the absolute value without having to deal with signed values and the additional bits needed to account for the signed portion of the negative values. Beginning one clock cycle after the differences start to be calculated, sub, sum_out is added to itself and sub. This process also occurs 81 times, one addition for each clock cycle. Figure 6 illustrates a diagram of the process which the differences between corresponding pixels is calculated.

$$(x, y) = argmin \sum_{(i,j) \in W} |I_1(i, j) - I_2(x + i, y + j)| \quad (1)$$

To get the former 9×9 slide window, a very appropriate hardware architecture for window-based image processing was

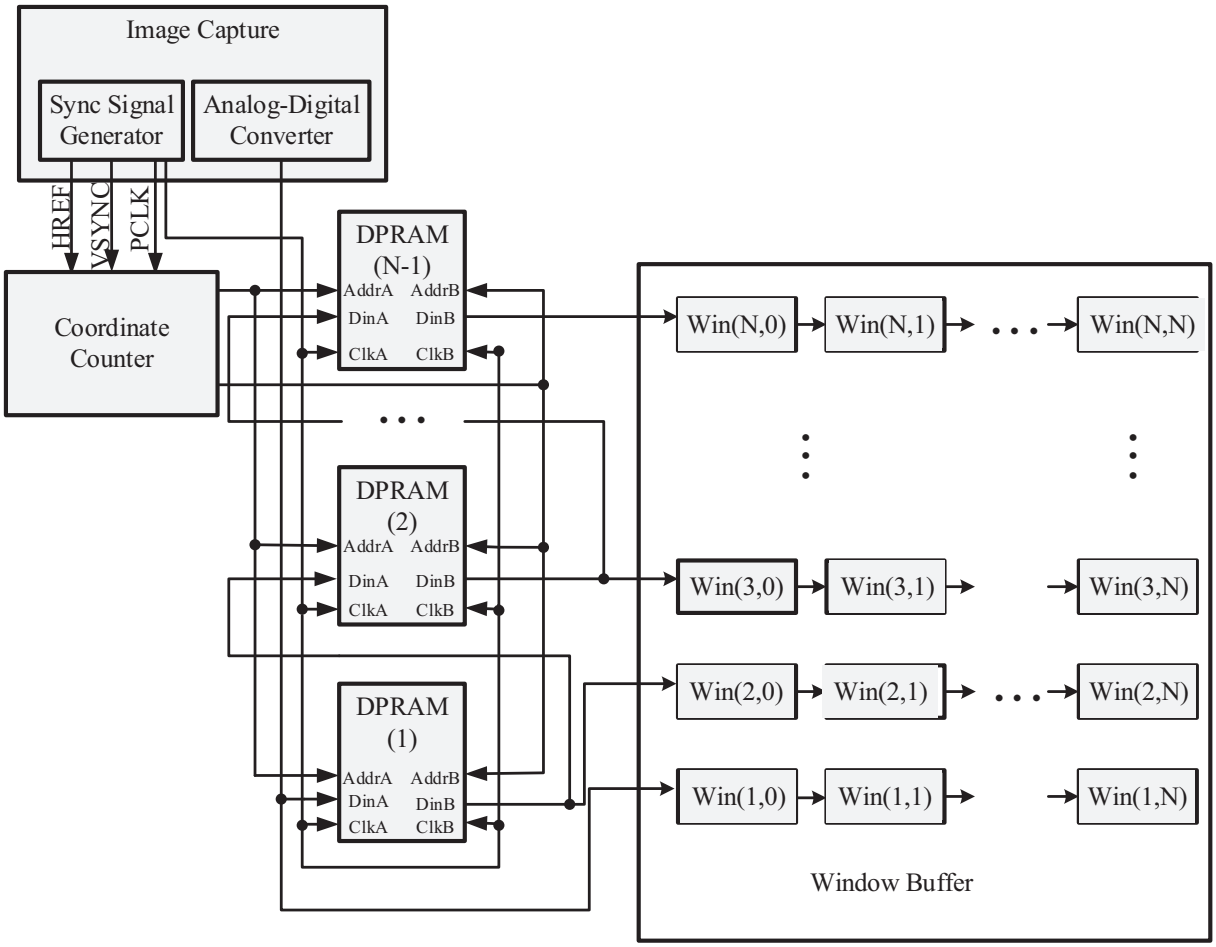

Fig. 7. Block diagram of the window-based image processing architecture

used in [18]. Fig.7 also shows the block diagram of the window-based image processing architecture. First, a dual port random access memory (DPRAM) inside the FPGA stores the latest N lines generated by the camera. And at the same time, the output port of the DPRAM generates $1 \times$ N pixels using the current horizontal coordinate as an address. These pixels are stored inside the window buffer which consists of registers. The advantage of these registers is that we can ensure that the pixels in the window buffer can be accessed simultaneously.

IV. Experimental Results

In order to verify the image mosaic algorithm based on SAD in our paper. It is implemented on the MATLAB2013 firstly which is illustrated in figure 8. The source images Img1 and Img2 were listed in the top viewed from left and right respectively. The process of the image registration took about $25s$ while the image size is only $450(H) \times 470(V)$ which is very time-consuming. Considering the time cost in image registration which implemented in PC although it shows good results, this is the reason why we choose FPGA as the processing platform.

The report of the flow summary of used resources are listed in Table I. Compared with traditional PC based implemented method, our method based on FPGA has the advantage of requiring fewer chip resources and power-saving, only (2.143 W in our design).

An implementation of the proposed system is shown in Fig.9. It contains the FPGA board, HDMI screen and the panoramic cameras printed by 3D printer. In Fig.10, the ila ip core was used to detect the signals such as the Hsync_HDMI, Vsync_HDMI, Den_HDMI, and the minimum value computed from the SAD module and the corresponding coordinate, coordinate_x and coordinate_y. Then we obtain the minmun value and the corresponding coordinate. The hardware architecture of our design contains all the necessary modules for capturing images, showing the resulted mosaiced image on a HDMI screen and also some other modules necessary for image processing, communication with PC via serial port RS232.

V. Conclusions and Future Work

Due to the flexibility of FPGA that is the the ability of reconfiguration which can implemented some complex image algorithm on a single chip while doesn't need to change the hardware. Therefore, we implemented the whole image mosaic algorithm on a single FPGA chip which obtained

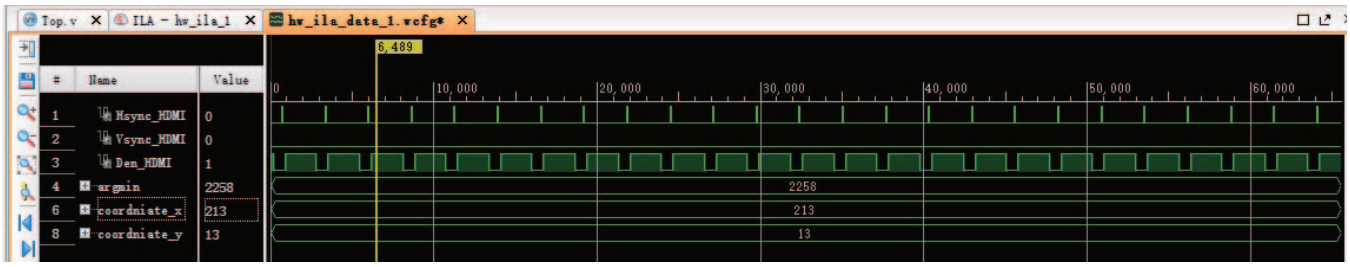

Fig. 10. Result waveform captured by the ila core

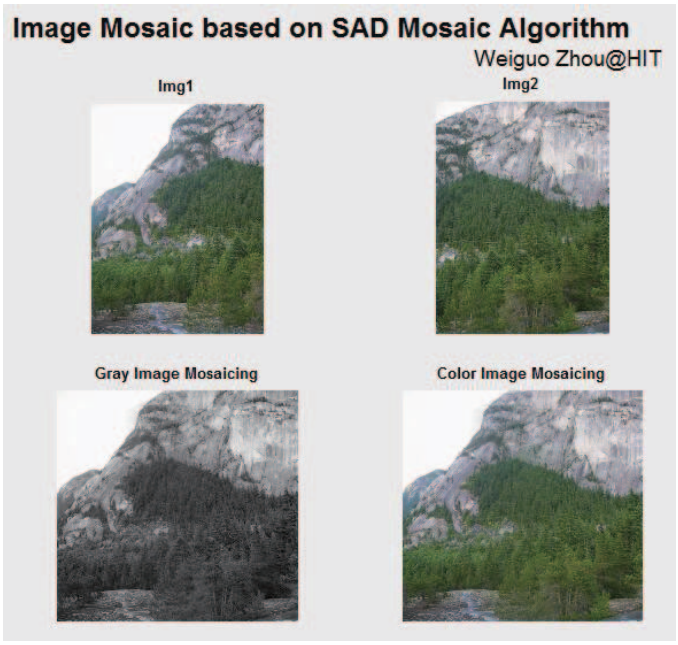

Fig. 8. Image mosaic algorithm based on SAD (MATLAB2013)

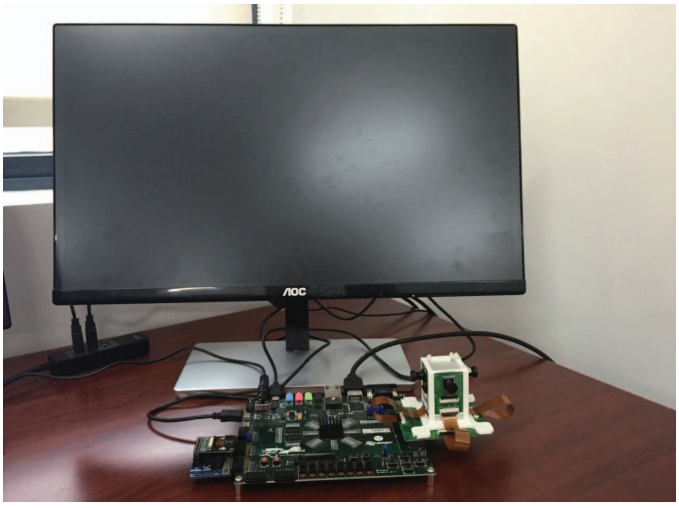

Fig. 9. Our experimental equipments

implementation results are similar to the simulation results of Matlab. In addition, we also look over the used resource and power consumed for implemented the system. The system is able to process more than 60 frames per second (fps) freely and maintaining a low power budget (just 2.143 W).

Based on the results we achieved in this paper, we will implement the image mosaic based on other algorithms such as Harris, SURF, SIFT and so on.

TABLE I
FLOW SUMMARY OF USED RESOURCES.

Resource	Utilization	Available	Utilization(%)
FF	43871	106400	41.23
LUT	39789	53200	74.79
Memory LUT	9500	17400	54.60
I/O	82	200	41.00
BRAM	60	140	42.86
DSP48	48	220	21.82
BUFG	6	32	18.75
MMCM	1	4	25.00

Total On-Chip Power: 2.143 W

ACKNOWLEDGMENT

The work supported by Shenzhen Peacock Plan Team grant KQTD20140630150243062, Shenzhen Fundamental Research grant JCYJ20140417172417120 and JCYJ20140417172417145, Shenzhen Key Laboratory grant ZDSYS20140508161825065.

REFERENCES

[1] P. Greisen, S. Heinzle, M. Gross, and A. P. Burg, "An fpga-based processing pipeline for high-definition stereo video," *EURASIP Journal on Image and Video Processing*, vol. 2011, no. 1, pp. 1–13, 2011.

[2] J. Phillips, "Choosing the right video interface for military vision systems," in *SPIE Sensing Technology+ Applications*. International Society for Optics and Photonics, 2015, pp. 948 111–948 111.

[3] O. Osibote, R. Dendere, S. Krishnan, and T. Douglas, "Automated focusing in bright-field microscopy for tuberculosis detection," *Journal of microscopy*, vol. 240, no. 2, pp. 155–163, 2010.

[4] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, "A real-time computer vision system for vehicle tracking and traffic surveillance," *Transportation Research Part C: Emerging Technologies*, vol. 6, no. 4, pp. 271–288, 1998.

[5] B. T. Morris and M. M. Trivedi, "A survey of vision-based trajectory learning and analysis for surveillance," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 18, no. 8, pp. 1114–1127, 2008.

[6] P. Gohl, D. Honegger, S. Omari, M. Achtelik, M. Pollefeys, and R. Siegwart, "Omnidirectional visual obstacle detection using embedded fpga," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 3938–3943.

[7] H. Pan, Y. Zhang, C. Li, and H. Wang, "An adaptive harris corner detection algorithm for image mosaic," in *Pattern Recognition*. Springer, 2014, pp. 53–62.

[8] D. Bheda, M. Joshi, and V. Agrawal, "A study on features extraction techniques for image mosaicing," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 2, no. 3, pp. 3432–3437, 2014.

[9] T. Ehang, M. A. Othman, N. H. Mahmood, and M. A. A. Razak, "Autofocus microscope system using contrast measurement approach," *Jurnal Teknologi*, vol. 74, no. 6, 2015.

[10] K. Pulli, A. Baksheev, K. Kornyakov, and V. Eruhimov, "Real-time computer vision with opencv," *Communications of the ACM*, vol. 55, no. 6, pp. 61–69, 2012.

[11] P.-Y. Hsiao, J.-H. Hong, C.-C. Hsu, H.-P. Lin, and S.-S. Huang, "A lane departure warning system with fpga modular design," in *Vehicular Electronics and Safety (ICVES), 2012 IEEE International Conference on*. IEEE, 2012, pp. 201–204.

[12] C. Ttofis, S. Hadjitheophanous, A. S. Georghiades, and T. Theocharides, "Edge-directed hardware architecture for real-time disparity map computation," *Computers, IEEE Transactions on*, vol. 62, no. 4, pp. 690–704, 2013.

[13] M. Samarawickrama, A. Pasqual, and R. Rodrigo, "Fpga-based compact and flexible architecture for real-time embedded vision systems," in *Industrial and Information Systems (ICIIS), 2009 International Conference on*. IEEE, 2009, pp. 337–342.

[14] C. Farabet, C. Poulet, and Y. LeCun, "An fpga-based stream processor for embedded real-time vision with convolutional networks," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 878–885.

[15] C. Citron, "Stereo vision system module for low-cost fpgas for autonomous mobile robots," 2014.

[16] S. Ma, Y. Shang, W. Zhang, Y. Guan, Q. Song, and D. Xu, "Design of panoramic mosaic camera based on fpga using optimal mosaic algorithm," *Journal of Computers*, vol. 6, no. 7, pp. 1378–1385, 2011.

[17] B. A. WHITE, "Using fpgas to perform embedded image registration," Ph.D. dissertation, University of Central Florida Orlando, Florida, 2009.

[18] S. H. Jin *et al.*, "The dynamic thresholding circuit of a real-time window-based image processing structure," *Asia Pacific System on a Chip*, pp. 631–634, 2004.