I.J. Image, Graphics and Signal Processing, **2012, 12, 19-25**
Published Online November 2012 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijigsp.2012.12.03

Real-time FPGA Based Implementation of Color Image Edge Detection

Sanjay Singh*, Anil Kumar Saini, Ravi Saini
IC Design Group
CSIR-Central Electronics Engineering Research Institute, Pilani - 333031, Rajasthan, India.
*E-mail: sanjaysingh@ceeri.ernet.in

***Abstract*—** Color Image edge detection is very basic and important step for many applications such as image segmentation, image analysis, facial analysis, objects identifications/tracking and many others. The main challenge for real-time implementation of color image edge detection is because of high volume of data to be processed (3 times as compared to gray images). This paper describes the real-time implementation of Sobel operator based color image edge detection using FPGA. Sobel operator is chosen for edge detection due to its property to counteract the noise sensitivity of the simple gradient operator. In order to achieve real-time performance, a parallel architecture is designed, which uses three processing elements to compute edge maps of R, G, and B color components. The architecture is coded using VHDL, simulated in ModelSim, synthesized using Xilinx ISE 10.1 and implemented on Xilinx ML510 (Virtex-5 FX130T) FPGA platform. The complete system is working at 27 MHz clock frequency. The measured performance of our system for standard PAL (720x576) size images is 50 fps (frames per second) and CIF (352x288) size images is 200 fps.

***Index Terms*—** Real-time Color Image Edge Detection, Sobel Operator, FPGA Implementation, VLSI Architecture, Color Edge Detection Processor

I. INTRODUCTION

High speed industrial applications require very accurate and real-time edge detection. Edge detection in gray images does not give very accurate results due to loss of color information during color to gray scale image conversion. Therefore, to achieve desired accuracy, detection of edges in color images is necessary. The main challenge for real-time implementation of color image edge detection is in processing of high volume of data (3 times as compared to gray images) within real-time constraints. Therefore, it is hard to achieve real-time performance of edge detection for PAL sizes color images with serial processors. Due to inherent parallelism property, FPGAs can deliver real-time time performance for such applications. Furthermore, FPGAs provide the possibility to perform algorithm modifications in later stages of the system development [1].

The main focus of most of the existing FPGA based implementations for edge detection using Sobel operator has been on achieving real-time performance for gray scale images by using various architectures and different design methodologies. As edge detection is low-level image processing operation, Single Instruction Multiple Data (SIMD) type architectures [2] are very suitable for edge detection to achieve real-time performance. These architectures use multiple data processing elements and therefore, require more FPGA resources. The architecture clock frequency can be improved by using pipelining. A pipelined architecture for real-time gray image edge detection is presented in [3]. Some computation optimized architectures are presented in [4, 5]. Few more architectures for real-time gray image edge detection are available in [6 - 11]. In [12, 13], the architectures are designed using MATLAB-Simulink based design methodology.

In this paper, we show that real-time Sobel operator based color image edge detection can be achieved by using a FPGA based parallel architecture. For each color component in RGB space, one specific edge computation processor is developed. As Sobel operator is sliding window operator, smart buffer based Memory architecture is used to move the incoming pixels in computing window. The specific datapaths are designed and controller is developed to perform the complete task. The design and simulation is done using VHDL. The design is targeted to Xilinx ML 510 (Virtex–5 FX130T) FPGA platform. The implementation is tested for real world scenario. It can robustly detect the edges in color images.

The rest of the paper is organized in the following way. In section 2 we describe the original Sobel operator based edge detection algorithm. We show, in section 3, the customized architecture for algorithm implementation and how each stage works. In section 4, practical tests are evaluated and synthesis results are shown taking into account system performance. Finally conclusions and discussions are presented in section 5.

II. EDGE DETECTION SCHEME

In this section the used algorithm is briefly described, for a more detailed description we refer to [14, 15]. The Sobel operator is widely used for edge detection in images. It is based on computing an approximation of the

Copyright © 2012 MECS

gradient of the image intensity function. The Sobel filter uses two 3x3 spatial masks which are convolved with the original image to calculate the approximations of the gradient. The Sobel operator uses two filters H_x and H_y.

$$H_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad (1)$$

$$H_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad (2)$$

These filters compute the gradient components across the neighboring lines or columns, respectively. The smoothing is performed over three lines or columns before computing the respective gradients. In this Sobel operator, the higher weights are assigned in smoothing part to current center line and column as compared to simple gradient operators. The local edge strength is defined as the gradient magnitude given by equation 3.

$$GM(x,y) = \sqrt{Hx^2 + Hy^2} \quad (3)$$

This equation 3 is computationally costly because of square and square root operations for every pixel. It is more suitable computationally to approximate the square and square root operations by absolute values.

$$GM(x,y) = |Hx| + |Hy| \quad (4)$$

This expression is much easy to compute and still preserves the relative changes in intensity (edges in images).

This above mentioned scheme is for gray scale images. For color images (RGB color space) this scheme is applied separately for each color component. Final color edge map of color image is computed by using the edge maps of each color component [16].

$$ColorEdge = (EdgeR \text{ or Edge G or Edge B}) \quad (5)$$

III. Proposed Architecture

To detect edges in real-time in PAL (720x576) size color images, dedicated hardware architecture is implemented for Sobel operator based edge detection scheme. Fig. 1 shows the conceptual block diagram of complete system. The hardware setup includes a video camera, a video daughter card, and FPGA board. The video output of camera connects to Digilent VDEC1 Video Decoder Board which is interfaced with Xilinx ML510 (Virtex–5 FX130T) board using Interface PCB. Display Monitor is connected to the board using DVI connector. The video signals are decoded in Camera Interface Module. The output RGB data of camera interface module is applied to edge detection block. The edge detected output from the Edge Detection Block is displayed on the display monitor using DVI controller. The camera interface module also generates video timing signals which are necessary for proper functioning of edge detection block and DVI controller. A more detailed description of this Camera Interface Design can be found in [17].

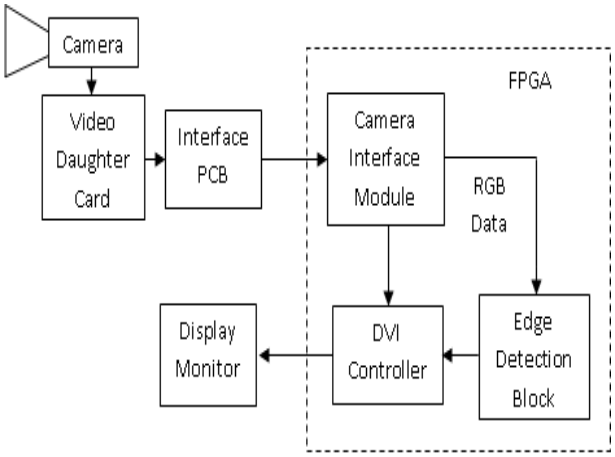

Figure 1. Complete System Block Diagram

Fig. 2 shows the basic block level data flow diagram for computation of edges in color images using Sobel operator. The goal is to perform edge detection three times, once each for red, green, and blue, and then the output is fused to form one edge map [16]. There are mainly four stages. First stage is Buffer Memory stage. In this stage, the three color components are separated and stored in three different memories. The second stage is gradient computation stage. In this stage the gradient is computed for each color component by adding absolute values of horizontal and vertical gradients. In third stage, the edge map is computed for each color component by comparing the gradient values with threshold. Final edge map is computed by combining the edge maps of each color components in stage four.

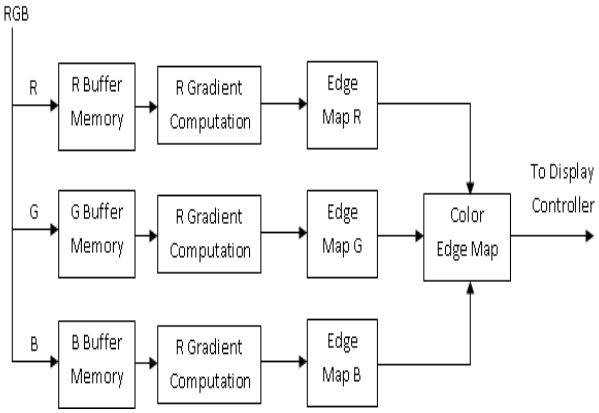

Figure 2. Edge Detection in Color Images using Sobel Operator

The streaming data processing cannot be used because the edge detection using Sobel operator algorithm is window based operation. Therefore, input data from camera is stored in on-chip memory (BRAM and Registers) before processing it on FPGA. The Sobel edge

Copyright © 2012 MECS

detection logic can begin processing as soon as two rows arrived in Buffer Memory. The Smart Buffer based Buffer Memory architecture [18] is used in the proposed Sobel operator based color edge detection implementation for data buffering. This approach (Fig. 3) works if one image pixel is coming from camera interface module in one clock cycle. The pixels are coming row by row. When buffers are filled, this architecture provides the access to the entire pixel neighborhood every clock cycle. The architecture places the highest demand on internal memory bandwidth. Because modern FPGA devices contain large amount of embedded memory, this approach does not cause problems [19]. The length of the shift registers depends on the width of input image. For PAL (720x576) size images the length of FIFOs is 717 (i.e. 720 - 3). For CIF (352x288) size images, it is 349 (i.e. 352 – 3).

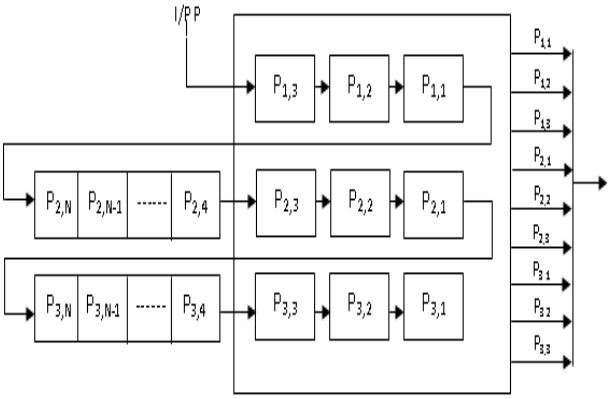

Figure 3. Sliding Window Memory Buffer Architecture

The absolute values of gradient H_x and H_y for pixel data $P_{2,2}$ is given by following expressions.

$$| Hx | = | (P_{1,3} - P_{1,1}) + (2 * (P_{2,3} - P_{2,1})) + (P_{3,3} - P_{3,1}) | \quad (6)$$

$$| Hy | = | (P_{3,1} - P_{1,1}) + (2 * (P_{3,2} - P_{1,2})) + (P_{3,3} - P_{1,3}) | \quad (7)$$

The processing modules architectures for computing these horizontal and vertical gradients for each color components are shown in Fig. 4. Both the architectures are same except the inputs applied to them at a particular time. Each processing module performs additions, subtractions, and multiplication. The multiplication is costly in digital hardware but multiplication by 2 is easily achieved by shift operation.

The complete architecture (Fig. 5) uses three processing elements in parallel (each for R, G, and B color components). The data coming from camera interface module is 24-bit RGB data. Incoming data is separated in three color components R, G, and B. Each color component data is of 8-bit (i.e. any value from 0 to 255). Three smart buffer based Sliding Window Memories are used to store two rows of the three color components. Each memory uses two First-in First-out (FIFO) shift-registers and 9 registers. The width of FIFOs and registers is 8-bit. Therefore, in total 6 FIFOs and 27 registers are required for designing Sliding Window Buffer Memory for RGB color image edge detection architecture. The designing of FIFOs using available registers in FPGA occupies large area in FPGA. Therefore, available Block RAMs on FPGA are used for designing the FIFOs. This resulted in efficient utilization of FPGA resources.

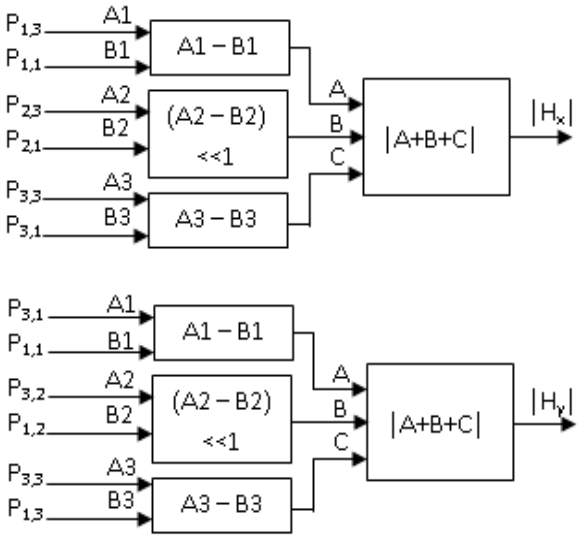

Figure 4. Gradient H_x and H_y Computation Module Architectures

For detecting edge in PAL (720x576) size color images, it takes 1440 (720x2) clock cycles to fill the two rows of image data in buffer memory. After this, in every clock cycle, each color component (R, G, and B) of new pixel is moved in their respective computing window (consists of 9 registers). The available 9 pixels in computing window ($P_{1,1}$, $P_{1,2}$, $P_{1,3}$, $P_{2,1}$, $P_{2,2}$, $P_{2,3}$, $P_{3,1}$, $P_{3,2}$, $P_{3,3}$) are used for computing the H_x and H_y gradient values. These are computed according to equations 6 and 7 by using the processing module architectures shown in Fig 4. The approximate magnitude of the gradient is computed along each color component by adding the absolute values of H_x and H_y. After this, the approximate gradient of each color component is compared with a user defined threshold value. If the approximate value of gradient is more than the user defined threshold, the comparator output for that color component is 1 else it is 0. The outputs of all three comparators (R, G, and B) are finally fused to find the final edge map. The final edge map is computed by ORing the Edge Map outputs of each color component. It requires one three input OR gate. If the final edge map output is one, the each color component value is set to 11111111 else it is set to 00000000. These values are used by DVI controller to display the result on display Monitor.

Copyright © 2012 MECS

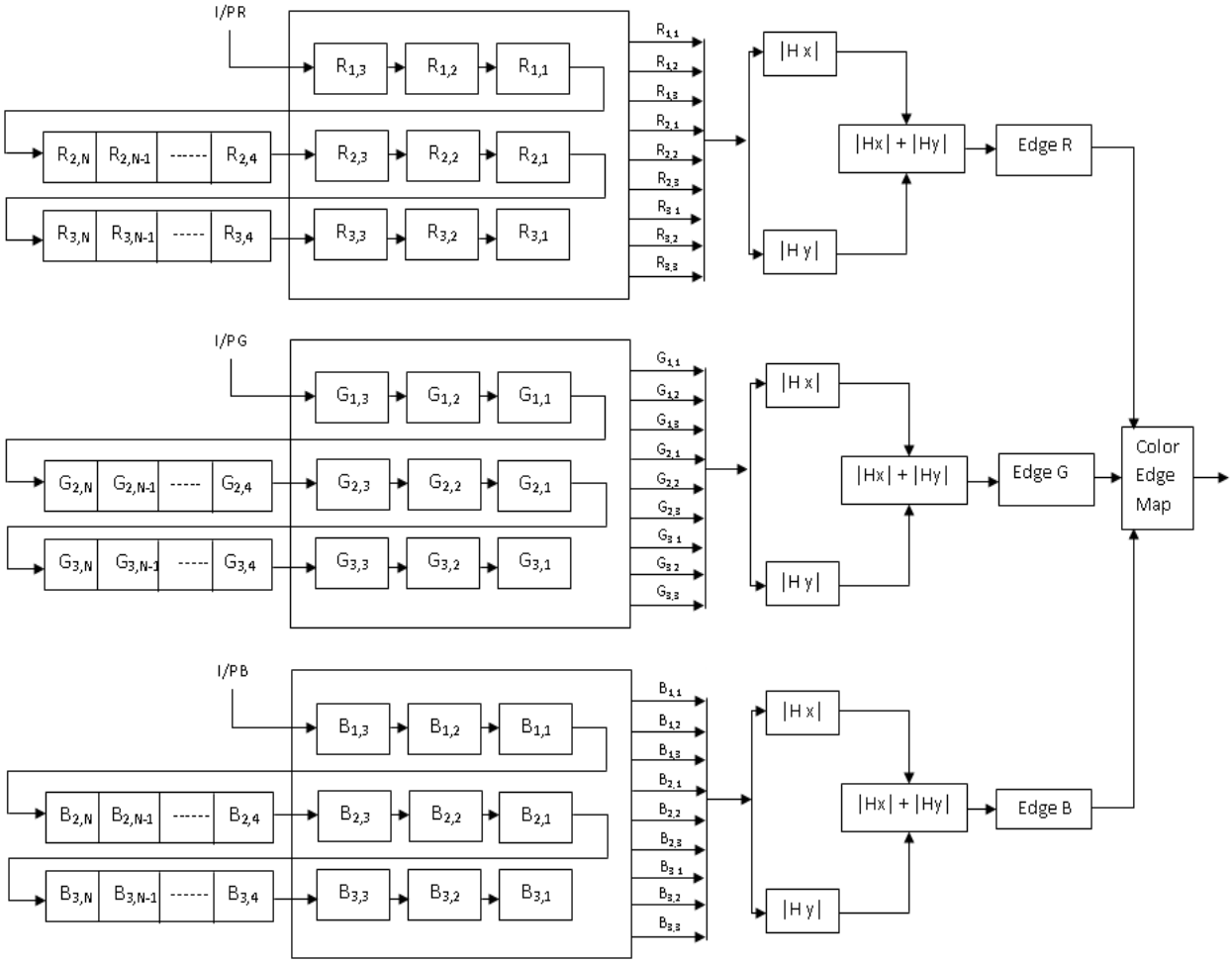

Figure 5. Complete Architecture for Color Image Edge Detection using Sobel Operator

IV. Results

The proposed architecture is designed using VHDL and simulated in ModelSim. Synthesis is carried out using Xilinx ISE 10.3. Final design is implemented on Xilinx ML510 (Virtex–5 FX130T) FPGA board. It utilizes 294 Slice Registers, 592 Slice LUTs, 206 FPGA Slices, 642 LUT Flip Flop Pairs, 116 Route-thrus and 3 Block RAMS. The synthesis results (Table I) reveal that the FPGA resources utilized by proposed architecture are approximately 1% of total available resources. The FPGA resource utilization table is only for proposed color image edge detection architecture (i.e. buffer memory, gradient computation, edge map computation) and excludes the resources utilized by camera interface and display logic. The measured performance of our system at 27 MHz operating frequency for PAL (720x576) size images is 50 fps (frames per second), CIF (352x288) size images is 200 fps and QCIF (176x144) size images is 800 fps. PAL and CIF images are most commonly used video formats. Therefore, implemented system can easily detect edges in color images in real-time.

TABLE I. Synthesis Results

Resources	Used	Available	Utilization
Slice Registers	294	81920	1%
Slice LUTs	592	81920	1%
Route-thrus	116	163840	1%
Occupied Slices	206	20840	1%
LUT Flip Flop Pair	642	-	-
BRAMs	3	298	1%

Copyright © 2012 MECS

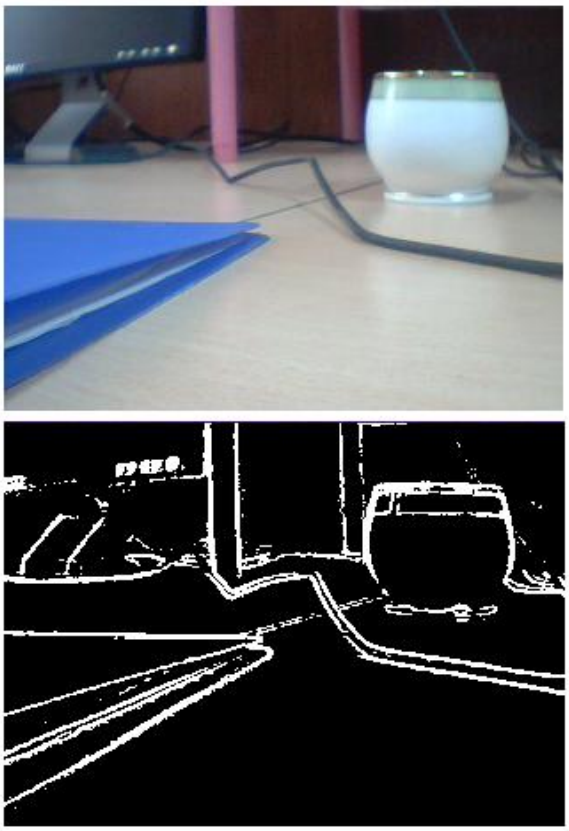

Figure 6. Input Color Image and Output Edge Detected Image

Figure 8. Input Color Image and Output Edge Detected Image

Figure 7. Input Color Image and Output Edge Detected Image

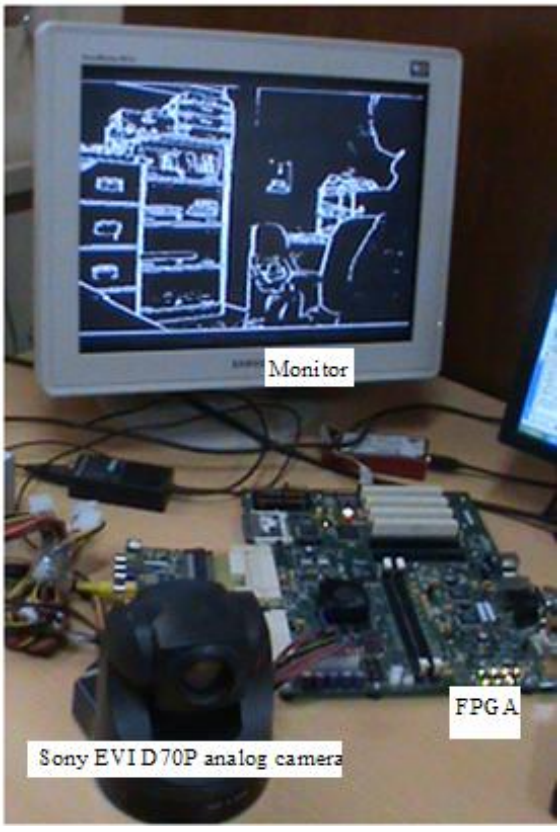

Figure 9. Complete System

In Fig. 6-8, the input PAL (720x576) size color test images taken from camera and respective output edge detected images produced by proposed architectures are shown. Fig. 9 shows the complete system. The images are captured by using Sony EVI D70P analog camera, processed by designed VLSI architecture running on FPGA, and displayed on monitor.

V. Conclusion

In this paper, the hardware architecture for Sobel operator based color image edge detection scheme has been presented. The architecture used approximately 1% of total FPGA resources and maintained real-time constraints of video processing. The system is tested for various real world situations and it robustly detects the edge in real-time with a frame rate of 50 fps for standard PAL video (60 fps for NTSC video) in color scale. The speed could be further improved by adding pipelining stages in gradient computation modules at the expense of increasing FPGA resources usage. The Xilinx ML510 (Virtex-5 FX130T) FPGA board is chosen for this implementation due to availability of large number of FPGA resources, Block RAMs, and PowerPC processor (for hardware-software co-design) so that same board can be used to implement other complex computer vision algorithms which make use of edge detection architecture. The proposed architecture is very suitable for high frame rate industrial applications. The future work will look at the use of this architecture for finding the focused areas in a scene for surveillance applications.

Acknowledgment

The authors express their deep sense of gratitude to Director, Dr. Chandra Shekhar, for encouraging research and development. Also the authors would like to express their sincere thanks to Dr. AS Mandal and Group Leader, Raj Singh, for their precious suggestions in refining the research work. Authors specially thank Mr. Sanjeev Kumar, Technical Officer, for tool related support. We thank to reviewers, whose constructive suggestions have improved the quality of this research paper.

References

[1] H. Jiang, H. Ardo, and V. Owall (2009), A Hardware Architecture for Real-Time Video Segmentation Utilizing Memory Reduction Techniques, IEEE Transactions on Circuits and Systems for Video Technology, vol. 19, no. 2, pp. 226–236.

[2] R.L. Rosas, A.D. Luca, and F.B. Santillan (2005), SIMD Architecture for Image Segmentation using Sobel Operators Implemented in FPGA Technology, In Proceedings of 2nd International Conference on Electrical and Electronics Engineering, pp. 77-80.

[3] T.A. Abbasi and M.U. Abbasi (2007), A Novel FPGA-based Architecture for Sobel Edge Detection Operator, International Journal of Electronics, vol. 94, no. 9, pp. 889-896, 2007.

[4] Z.E.M. Osman, F.A. Hussin, And N.B.Z. Ali (2010a), Hardware Implementation of an Optimized Processor Architecture for Sobel Image Edge Detection Operator, In Proceeding of International Conference on Intelligent and Advanced Systems (ICIAS), pp. 1-4.

[5] Z.E.M. Osman, F.A. Hussin, And N.B.Z. Ali (2010b), Optimization of Processor Architecture for Image Edge Detection Filter, In Proceeding of International Conference on Computer Modeling and Simulation, pp. 648-652

[6] I. Yasri, N.H. Hamid, And V.V. Yap (2008), Performance Analysis of FPGA based Sobel Edge Detection Operator, In Proceedings of International Conference on Electronic Design, pp. 1-4.

[7] V. Sanduja and R. Patial (2012), Sobel Edge Detection using Parallel Architecture based on FPGA, International Journal of Applied Information Systems, vol. 3, no. 4, pp. 20-24.

[8] G. Anusha, T.J. Prasad, and D.S. Narayana (2012), Implementation of SOBEL Edge Detection on FPGA, International Journal of Computer Trends and Technology, vol. 3, no. 3, pp. 472-475.

[9] L.P. Latha (2012), Design of Edge Detection Technique Using FPGA (Field Programmable Gate Array) and DSP (Digital Signal Processor), VSRD International Journal of Electrical, Electronics & Communication Engineering, vol. 2, no. 6, pp. 346-352.

[10] A.R. Ibrahim, N.A. Wahed, N. Shinwari, and M.A. Nasser (2011), Hardware Implementation of Real Time Video Edge Detection With Adjustable Threshold Level (Edge Sharpness) Using Xilinx Spartan-3A FPGA, Report.

[11] P.S. Chikkali and K. Prabhushetty (2011), FPGA based Image Edge Detection and Segmentation, International Journal of Advanced Engineering Sciences and Technologies, Vol. 9, Issue 2, pp. 187-192.

[12] R. Harinarayan, R. Pannerselvam, M.M. Ali, And D.K. Tripathi (2011), Feature extraction of Digital Aerial Images by FPGA based implementation of edge detection algorithms, In Proceedings of International Conference on Emerging Trends in Electrical and Computer Technology, pp. 631-635.

[13] K.C. Sudeep and J. Majumdar (2011), A Novel Architecture for Real Time Implementation of Edge Detectors on FPGA, International Journal of Computer Science Issues, vol. 8, no. 1, pp. 193-202.

[14] W. Burger and M.J. Burge (2008), Digital Image Processing: An Algorithmic Introduction Using Java, New York: Springer, 120-123.

Copyright © 2012 MECS

[15] R.C. Gonzalez, and R.E. Woods (2009), Digital Image Processing, India: Pearson Education, Inc., 187-190.

[16] M.A. Ruzon, A Short History of Color Edge Detection. http://ai.stanford.edu/~ruzon/compass/color.html

[17] S. Singh, A.K. Saini, R. Saini, A.S. Mandal, C. Shekhar, and A. Vohra (2012), Real-time Video Acquisition and PTZ Camera Movement Control for FPGA based Automated Video Surveillance System, International Journal of Research & Reviews in Computer Science, Vol. 3, No. 2, pp. 1572-1575.

[18] C. Moore, H. Devos, and D. Stroobandt (2009), Optimizing the FPGA Memory Design for a Sobel Edge Detector, In Proceedings of 20th Annual Workshop on Circuits, Systems and Signal Processing, pp. 496-499.

[19] Z. Vasicek, and L. Sekanina (2008), Novel Hardware Implementation of Adaptive Median Filters, In Proceedings of 11th IEEE workshop on Design and Diagnostics of Electronic Circuits and Systems, pp. 1-6.

Sanjay Singh is working as Scientist in CSIR-Central Electronics Engineering Research Institute, Pilani, Rajasthan, India. He is Member of IEEE - USA, IACSIT - Singapore, and IAENG - Hong Kong. Currently, he is involved in various projects sponsored by Indian Government on Computer Vision and Smart Cameras. His research interests are VLSI architectures for image & video processing algorithms, FPGA Prototyping, and Computer Vision. Prior to joining this research lab, he received his Master in Technology, Master in Electronics, and Bachelor in Science in 2007, 2005, and 2003 respectively from Kurukshetra University, Kurukshetra, Haryana, India. He earned Gold Medal (First Position in University) during his Master in Technology and Master in Electronics. He topped college during his Bachelor in Science. He received more than 20 Merit Certificates and Scholarships during his academic career.

Anil Kumar Saini is working as Scientist in CSIR-Central Electronics Engineering Research Institute, Pilani, Rajasthan, India. He is Member of IEEE – USA and IACSIT – Singapore. His research interests include Analog and Mixed Signal Design, Embedded System Design, and CMOS RF IC design. Prior to joining this research lab, he worked in Cadence, India. He received his Master in Technology and Master in Science in 2003 and 2000 respectively from IIT Roorkee, India.

Ravi Saini is working as Scientist in CSIR-Central Electronics Engineering Research Institute, Pilani, Rajasthan, India. He is Member of IEEE – USA. His research interests include VLSI Architectures, ASIC and ASIP Design, HDLs, and FPGA Prototyping. He received his Master in Technology in 2002 from Punjab University, India and Master in Electronics in 2000 from DAVV, Indore, India.

Copyright © 2012 MECS