IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Revue d'Intelligence Artificielle
Vol. 33, No. 4, August, 2019, pp. 305-311
Journal homepage: http://iieta.org/journals/ria

Compact Hardware of Running Gaussian Average Algorithm for Moving Object Detection Realized on FPGA and ASIC

Kaushal Kumar[1*], Durgesh Nandan[2], Ritesh Kumar Mishra[1]

[1] Department of ECE, National Institute of Technology, Patna 800005, India
[2] CL Educate Ltd., New Delhi 110044, India

Corresponding Author Email: kaushal.ec16@nitp.ac.in

https://doi.org/10.18280/ria.330407

Received: 16 March 2019
Accepted: 20 July 2019

Keywords:
ASIC, background subtraction, FPGA, moving object detection, running gaussian average, video processing

ABSTRACT

Real-time detection of moving object is essential to traffic monitoring, video surveillance and many other vital applications. The effectiveness of real-time detection hinges on background subtraction. This paper proposes a hardware-efficient architecture for the running Gaussian average (RGA) technique of background subtraction. The architecture was designed based on the field programmable gate array (FPGA) and application specific integrated circuit (ASIC). The FPGA was realized using Digilent ZedBoard, while the ASIC was implemented using Cadence Genus, Innovus, and Assura tools at 45 nm process technology. A prototype of our architecture was tested with a video containing multiple moving objects, and another containing a single moving object. The results show that the proposed architecture is efficient in terms of area, power and timing. Therefore, this paper provides a background subtraction approach that uses hardware resources effectively without sacrificing the detection accuracy.

1. INTRODUCTION

Automated video Surveillance System (VSS) and traffic monitoring are nowadays gaining significant concern among researchers in recent times. Real-time moving object detection has been progressively watched as an essential component in different significant applications, such as traffic monitoring and VSS. A critical part of all such applications is the accurate extraction of objects of interest from an insignificant background scene. The various object detection techniques can be classified into two broaden categories (1) adaptive techniques and (2) nonadaptive techniques. Nonadaptive techniques have a few constraints that make it unfit of nowadays prerequisites like the requirement for manual initialization. For conquering constraints of nonadaptive techniques, adaptive technique comes into the picture. Adaptive techniques are advantageous in the sense that it averages the incoming frames over time, bringing about the formation of approximate background. Various adaptive techniques exist like frame differencing, mean filtering, median filtering, Running Gaussian Average (RGA), Mixture of Gaussian (MoG), and Kernel Density Estimation (KDE). Adaptive algorithms perceive the moving objects by observing the difference between the current video frame and background model comprising of the non-dynamic section of the video. The difference is then compared with the threshold value, and if it is higher than the threshold value, then the pixel is recognized as foreground otherwise background. The objective of such techniques is to create the background model that ought to have the option to get updated according to changes in the background scene such as illumination changes, camera oscillation, etc.

RGA is a simple technique in contrast with existing background detection adaptive techniques. It provides high Frame per Second (fps) with acceptable accuracy. The hardware implementation has less memory requirement in comparisons of others and thus preferred for usage. With time, several researchers have done work for betterment in the performance of RGA. Tang et al. associated this technique with the frame difference technique to fill the gaps in which a moving object might be contained [1]. Jabri et al. [2] associated this technique with edge information to enhance the accuracy of such a scheme.

Many researchers have suggested different hardware architecture approach to address the problem of an efficient hardware implementation of moving object detection algorithms. Based on design methodologies, the hardware implementation can be classified as general purpose-based, digital signal processor-based, Complex Programmable Logic Device (CPLD) based, Application Specific Integrated Circuit (ASIC) based, and Field Programmable Gate Array (FPGA) based. Moving object detection is an essential constituent of an automated surveillance system to be utilized as a stand-alone system. The real-time video processing systems require highly complex computational units. Due to the parallel processing ability of FPGA, it is found to be suitable for the realization of image and video processing algorithms with high performance. FPGA provides the techniques to be directly realized onto the hardware. Since the instant prototyping and development of techniques are conceivable on FPGA, it is selected here for moving object detection.

The significant contributions of this paper are as follows:

– Composing hardware efficient architecture of RGA technique for moving object detection that permits speed improvement and low power consumption.
– Development of real-time moving object detection on Artix-7 FPGA device, achieving 60 frames per second (fps) for 640x480 pixels video.
– ASIC implementation of the proposed architecture at 45 nm process technology using Cadence Genus, Innovus and

Assura tools.

Rest of the paper is organized as follows: Section 2 explores related work of background subtraction, Section 3 explores the proposed approach of RGA implementation, Section 4 provides the results and analysis and Section 5 concluded entire works and scope for future work possibilities

2. RELATED WORK

The broad classification of background modeling techniques is shown in Figure 1. Background modeling techniques are incorporated by various techniques like subspace methods based [3], neural network-based models [4-5], parametric and nonparametric models, etc. This section is partitioned into two parts. The standard background subtraction techniques are discussed in the first part and its hardware implementation in the second part.

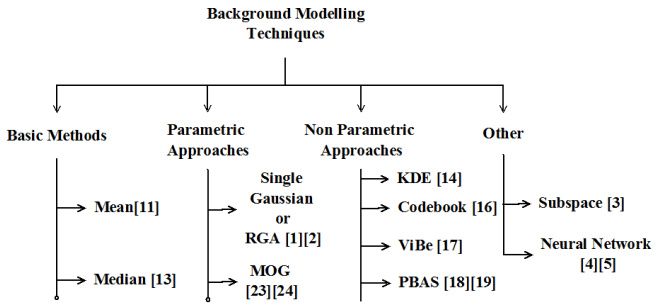

Figure 1. Classification of background modeling techniques

2.1 Background subtraction techniques

Various background subtraction techniques have been developed until now. Among them, the simplest one is frame differencing, which detects the moving object by examining the difference between the current frame and the reference model [6]. In this, the background frame is developed by averaging N, the number of previous frames as given in Eq. (1).

$$\beta(x,y,t) = \frac{1}{N}\sum_{i=1}^{N} I(x,y,t-i) \qquad (1)$$

Jacques, et al. [7] and Stefano, et al., [8] employed Running Gaussian Average for creating the background model, which is subtracted from incoming frames to detect the moving objects. The single Gaussian model has only one source of variation, i.e., additive Gaussian noise. The objective of such algorithms is to create a background model that is updated with time. For example, to update the background model, Ridder et al., [9] and Karmann and Brandt [10] utilizes a Kalman filter, Toyama et al. [11] used a Wiener filter, and Jacques et al., [12] and Hung et al. [13] recommended a method based on median filtering. These methods address the light intensity variation in the background. Elgammal et al., [14] used a non-parametric approach, i.e., KDE, which models the background by aggregating the histogram of pixels. There is a wide utilization of KDE in the case of multimodal PDF. Stauffer and Grimson [15] used Gaussian Mixture Models (GMM), which assumes that the background pixels follow multiple Gaussian distributions. It is preferred in an outdoor environment, where the pixel value is different in different frames due to more than one source of the noise. The pixel intensity is evaluated by a mixture of k Gaussian distributions. The likelihood that a specific pixel has intensity

x_t at 't' is given by Eq. (2).

$$P(x_t|B) = \sum_{i=1}^{k} w(i,t)G(x_t; u_{i,t}\sum(i,t)) \qquad (2)$$

where, w(i,t) is its weight, G is Gaussian function with mean, $u_{i,t}$ and covariance, $\sum(i,t) = \sigma^2_{i,t}I$. Kim et al. [16] used codebook technique, which associates each background pixel with a codeword. Barnich, et al. [17] used ViBe, which works at a very high speed and creates a background model using neighboring pixels and past pixels of the given pixel. Hofmann et al. [18] Kryjak et al. [19] used another background subtraction technique called Pixel-Based Adaptive Segmenter (PBAS) that maintains multiple background samples in the background model. In this, the decision threshold and learning parameters are dynamic per-pixel. Controllers are acquainted with steering these parameters to estimate the background. Subspace learning methods [3] have been developed to decrease the dimensionality of data, which reduces the computational complexity.

2.2 Hardware realization of various background subtraction techniques

Several researchers have done the hardware realization of background subtraction techniques. Jiang et al. have done FPGA implementation of video segmentation unit for automated video surveillance systems, capable of processing at 25 fps and reducing the memory requirement of more than 70% using word length reduction and pixel locality [20]. Jiang et al. proposed a hardware architecture that is capable of handling multimodal background conditions, and the VirtexII FPGA platform used for hardware realization. The system processes video sequences of resolution 1024 x 1024 at 38 fps and can achieve up to a 60% reduction in memory requirement [21]. Kristensen et al. [22] has improved the segmentation circuit proposed by Jiang et al. [21]. Genovese, et al. presented the FPGA realization of GMM, which consumes less energy and is faster in comparison with previous implementations. The authors maneuver the update equations to simplify hardware implementation [23, 24]. Cherian et al. [25] implemented RGA on FPGA for detecting moving objects, which can process video stream of 640 x 480 resolution. The algorithm realized on Digilent Atlys Spartan 6 FPGA board. VmodCAM is used to capture real-time video.

In brief, although several researchers have worked on the hardware implementation of different background subtraction techniques, yet not many of them have done the hardware implementation of the RGA technique. Due to the simplicity and high frame rate of the RGA technique with acceptable accuracy, it is considered in this paper, and efficient hardware architecture is proposed here.

3. PROPOSED APPROACH OF RGA IMPLEMENTATION

The crucial step in the detection of moving objects employing the background subtraction technique is the creation of the background frame. In the RGA method, Gaussian PDF is employed to develop the background. Running average is computed to avoid fitting the PDF from

the start of each new frame. For the creation and maintenance of background using RGA technique, we need to update the background model recursively using exponential forgetting, i.e.,

$$\beta_t = \alpha I_t + (1-\alpha)\beta_{t-1} \quad (3)$$

$$|I_t - \beta_t| > T \quad (4)$$

where, α is the learning rate which decides the speed of ignoring the background information, and its value lies between 0 and 1. Also 'I_t' is the current image at a time 't' and β_{t-1} is background image at (t-1). We can observe from Eq. (3) that the overall mean is a weighted mean of the current pixel value and past values of pixels. The difference of current frame and calculated background frame is then thresholded to detect the moving object present in the scene, given by Eq. (4). To evaluate the mean of past n samples, two floating-point multipliers, and an addition unit are required, which increases hardware utilization. The increase of hardware requirement adversely affects the power and area utilization. In the proposed approach, the floating-point manipulation units are replaced by decimal manipulation units, which reduce the hardware utilization to a great extent. The step by step process of the proposed technique for the realization of moving object detection using Running Gaussian Average technique is as follows:

Algorithm:

For an easy understanding of the proposed technique, it has developed an algorithm based on the mathematical model of design. It has been explained in 12 steps as given below:

Step 1	The 24-bit wide RGB pixel of an input video frame is first converted into 8-bit gray scale value using Figure 2.
Step 2	The first frame of video is used for background initialization and hence should be stored in memory.
Step 3	The upcoming frames are employed along with first frame for creating the background.
Step 4	The 8-bit wide gray pixel of current frame and previous frame is converted into 16-bit wide pixel by using an 8-bit shift operation.
Step 5	The learning rate, α, whose value lies between 0 and 1 is multiplied with 1000 and the resultant is directly used in our implementation process.
Step 6	The 16-bit wide pixel value of current frame is multiplied with α. The resultant is stored in 32-bit wide variable say $gray_{32}$. Radix 4 booth multiplier is used here for all multiplications required in this implementation.
Step 7	The 16-bit wide pixel value of previous frame is multiplied with resultant of 1000 * (1 $-\alpha$). The resultant is stored in 32-bit wide variable, say $background_{32}$.
Step 8	The 32-bit values, i.e., $gray_{32}$ and $background_{32}$ are then added to obtain updated background. For addition, 32-bit ripple carry adder is used in this paper.
Step 9	The 32-bit wide updated background is then divided by 1000 to compensate the modifications and the resultant is stored in a 16-bit wide variable.
Step 10	The 16-bit wide updated background is shifted to right by 8 bits and the resultant 8-bit value of updated background is obtained.
Step 11	The current frame pixel value which is 8-bit wide is subtracted from 8-bit wide updated background pixel to obtain the difference.
Step 12	The difference value is then thresholded to detect a moving object from a sequence of video frames.

Based on algorithm steps, an example is given below for an easy understanding of proposed concepts. Assuming first pixel of first frame be

$B = 24'b110010000110010011001000$ ($24'd13133000$) and first pixel of second frame be

$C = 24'b011001001100100001100100$ ($24'd6604900$). The 24 bit RGB pixel is now converted to 8 bit gray pixel which is given as:

$B = 8'b10001101$, $C = 8'b10011111$

$B_{16} = B << 8 = 16'b1000110100000000$,

$C_{16} = C << 8 = 16'b1001111100000000$

Let $\alpha = 0.01$ then $1000 \times \alpha = 1000x0.01 = 10$. Let $\alpha_1 = 10$ and $\alpha_2 = 1000 - \alpha_1 = 990$ $background_{32} = B_{16} \times \alpha_2 = 32'd35735040$, $gray_{32} = C_{16} \times \alpha_1 = 32'd407040$

$updated\_background_{32} = background_{32} + gray_{32} = 32'd36142080$

$updated\_background_{16} = updated\_background_{32} \div 1000 = 16'd36142$

$update\_background = updated\_background_{16} >> 8 = 8'b10001101 = 8'd141$

Let the incoming frame first pixel value is $8'd200$ then Difference $= 8'd200 - updated\_background = 8'd200 - 8'd141 = 8'd59$.

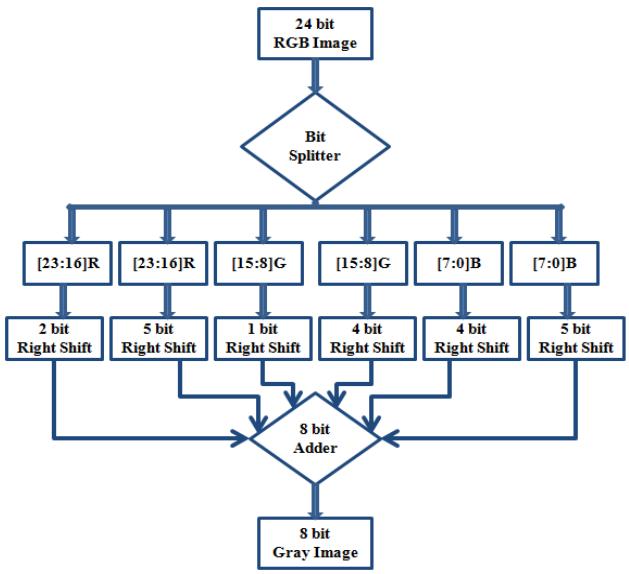

Figure 2. Block diagram of RGB image to gray conversion

Since the difference is less than the threshold (assuming 150), then output is the same as difference value. Various researchers have given different ways of RGB to gray conversion. In this paper, a novel architecture of RGB to gray conversion is proposed where transformation is accomplished by using shift and addition operations, as shown in Figure 2. The 24-bit RGB pixel is divided into separate 8 bit R, G, and B pixels, then the desired right shift is carried out on each of them, and finally, they are added up to obtain an 8-bit gray

image. Figure 3 shows the existing architecture of RGA implementation. The gray pixel obtained from RGB to gray conversion needs to be converted into floating-point value to carry out further computations. The translation of decimal value to floating-point value requires IEEE 754 floating-point conversion unit. Since there is a necessity of floating-point multiplication and addition blocks for computation, it leads to an increase in resource utilization. In the proposed architecture, there is no need for the IEEE 754 conversion unit as pixel value is directly fed in the system to carry out the computation process. Figure 4 provides an architecture view of the proposed methodology. Radix 4 booth multiplier is used to carry out all the required multiplications. Figure 5 provides the block diagram of the proposed methodology of RGA implementation.

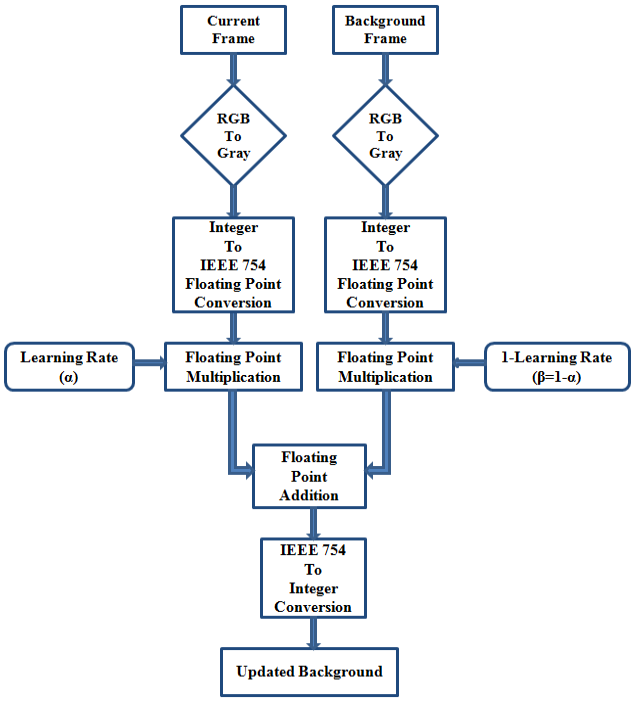

Figure 3. Existing architecture of RGA implementation

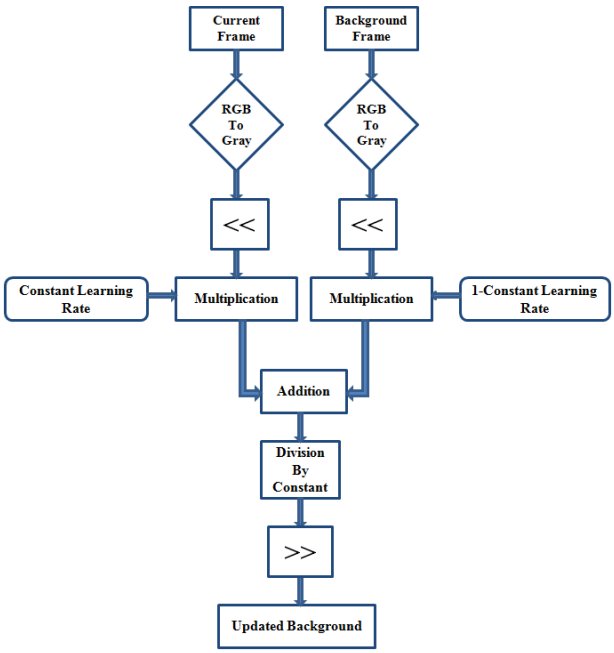

Figure 4. Proposed architecture of RGA implementation

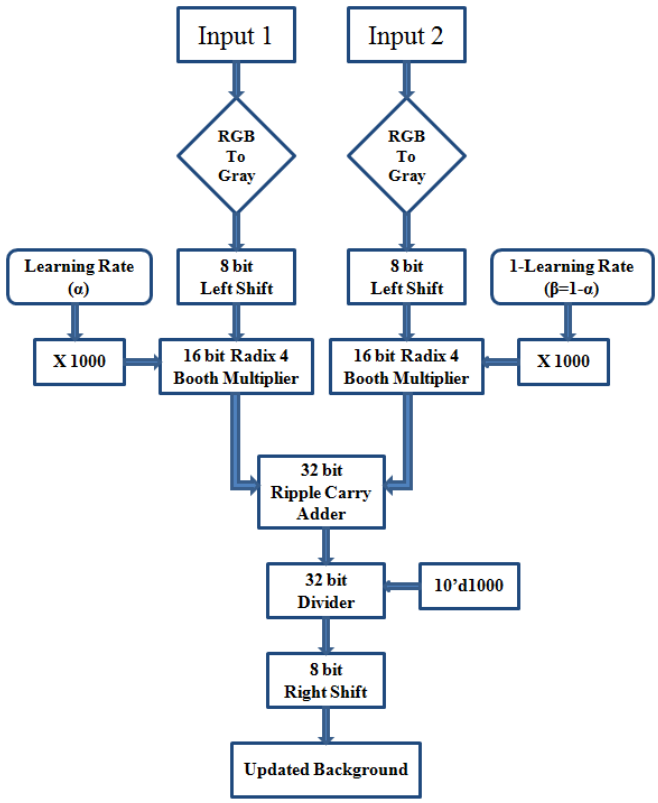

Figure 5. Block diagram of proposed RGA implementation

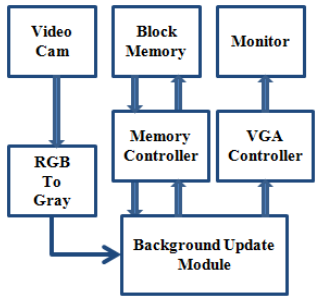

Figure 6. Conceptual view of real time FPGA implementation of background subtraction

Figure 6 shows the real-time implementation of background subtraction on FPGA, where the background update is carried out using the RGA method. Block memory present on FPGA board is used to store the frame pixels value during processing. This block memory requires an interfacing module for its usage. The resultant video frames containing detected moving objects obtain on a monitor of 640 x 480 resolution @60 Hz, which is accomplished using a VGA controller module.

4. RESULTS AND ANALYSIS

The FPGA implementation of the proposed framework is accomplished using the Xilinx Vivado 2016.2 tool. The design realized on Digilent ZedBoard (xc7z020clg484-1), which has a 28 nm Xilinx Artix-7 FPGA device. Vivado in-built logic simulator is used to perform the required circuit simulations. Table 1 provides the post-implementation resource utilization of proposed architecture for the background updation module. The existing architecture is also implemented on ZedBoard and a comparison made with the proposed architecture. The

proposed framework is capable of processing video in real-time with reduced hardware requirements compared to an existing architecture. Table 1 shows that the required LUT reduces to a large extent, but it also requires 156 FF. The VGA interfacing module also implemented on FPGA, and its resource utilization is given in Table 2. The implementation of the VGA interfacing module on ZedBoard requires 0.06% of LUT, 0.04% of FF, and 1 global buffer. Table 3 represents the variation of area, power, and timings concerning the frequency variation at 45nm process technology. Difference between data required time and data arrival time produces slack time. For avoiding timing violations, the slack time must be zero or positive. From Table 3, the proposed technique seems to have a positive slack of 4.44 ns at 100 MHz clock frequency with a total power consumption of 106.6 μW. As operating frequency increases, there is an increase in power consumption and a decrease in arrival time. We can observe that the slack time approaches 0 ns as the clock frequency approaches 357 MHz and power reaches to 446.1 μW. Figure 7 shows the slack variation to frequency change for the proposed architecture. Figure 7 shows that at 100 MHz clock frequency, the design produces a positive slack of 4.5 ns. Further, the slack time reduces to 0.01 ns at 312 MHz. With further increase of frequency, we get 0 ns slack time. While for the existing architecture, the system produces a positive slack of 5.8 ns at 10 MHz. The slack then reduces to 0 ns at 50 MHz, and the slack of 0 ns continues to remain till 100 MHz. The existing architecture produces a negative slack of -0.12 ns at 125 MHz, which further reduces to -1.3 ns at 166 MHz and -3.9 ns at 285 MHz.

It infers that the proposed architecture performs better at higher frequency in comparison to an existing architecture. Table 4 shows the resource utilization for the ASIC implementation of the background updation module at 45 nm process technology. The synthesis is accomplished using the Cadence Genus tool. Also, physical design and verification are performed using Cadence Innovus and Assura tools, respectively. It is observed from Table 4 that the total area for logic implementation is reduced by approximately 82%, and the total power consumption is reduced by 58% for the ASIC implementation of proposed architecture of background subtraction. Also, there is a reduction in 97% in area delay product (ADP) for the proposed architecture. Figure 8 shows the outcome of physical design, which includes floorplanning, placement, clock tree synthesis, and routing.

Table 1. Comparison of resource utilization for the proposed and existing architectures on FPGA

Resource	Proposed	Existing
LUT	157	1461
Flip Flop	156	186
IO	59	25
BRAM	100.5	100.5
BUFG	1	1

Table 2. Resource utilization of FPGA implementation for VGA module

Resource	Utilization	Available	Utilization %
LUT	34	53200	0.06
Flip Flop	40	106400	0.04
IO	37	200	18.50
BUFG	1	32	3.13

Table 3. Variation of area, power and delay with respect to frequency at 45nm process technology for proposed architecture

Parameters → Total Area (μm^2) / Frequency (MHz) ↓	Total Power (μW)	Required Time (ns)	Arrival Time (ns)
100	2929	106.6	9.88	5.44
166	2930	166.4	6.88	4.59
192	2932	204.2	5.08	4.54
217	2943	230.8	4.50	4.48
285	3085	320.1	3.39	3.38
312	3176	351.9	3.09	3.08
357	3413	446.1	2.68	2.68

Table 4. Performance comparison of ASIC implementation of proposed and existing architectures at 45 nm process technology

Parameters	Proposed	Existing
Total Area (μm^2)	3413	19305
Total Power (μW)	446.1	1072.8
Arrival Time (ns)	2.68	19.88
ADP ($um^2 * us$)	9.14	383.78

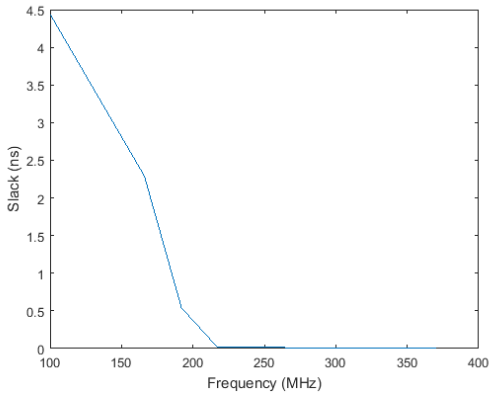

Figure 7. Variation of timing slack w.r.t. frequency for the proposed architecture

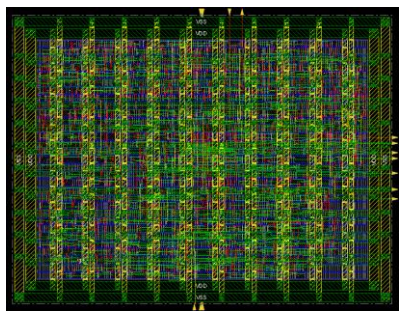

Figure 8. Physical design of proposed architecture of RGA

The experimental validation of proposed architecture is performed using two test videos from Actions as space-time shapes dataset [26]. One of the test videos contains multiple moving objects, and another test video consists of a single moving object. Figure 9 shows some of the frames of input test videos and the corresponding detected moving objects present in the videos. The detected moving objects seem to be in grayscale rather than color because of the initial conversion of the input video frames into the grayscale for further processing.

Figure 10 shows the FPGA realization of moving object detection using the proposed framework.

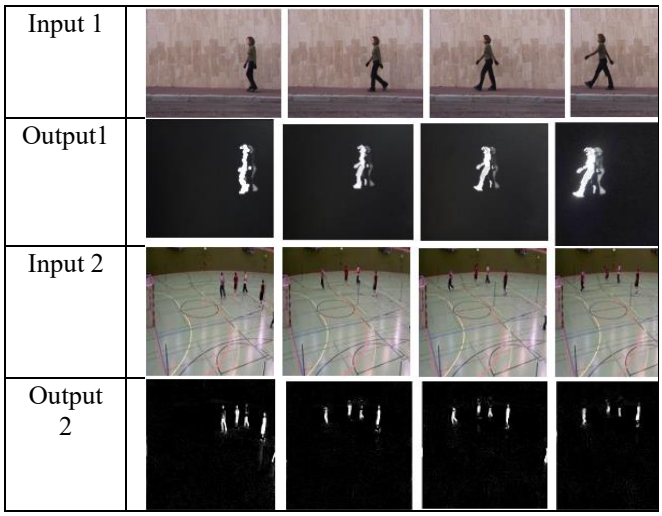

Figure 9. Some of the input and corresponding output frames of two test videos

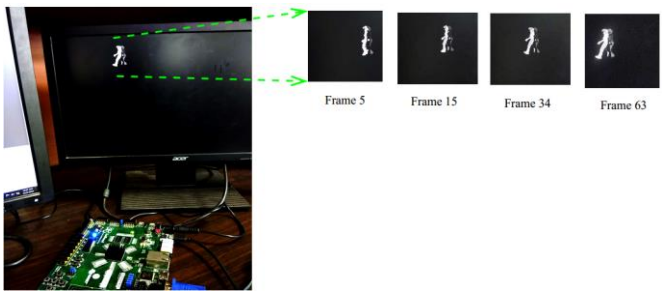

Figure 10. FPGA realization of RGA for moving object detection

5. CONCLUSION

This paper proposed an efficient hardware architecture for the FPGA and ASIC implementation of the RGA technique of background subtraction. FPGA realization is performed using Digilent ZedBoard, and ASIC implementation is performed using Cadence Genus, Innovus, and Assura tools at 45nm process technology. The proposed architecture of the RGA technique is found to be efficient in terms of area, power, and timing. The implemented prototype system is seen to detect relative motion from the incoming video stream in real-time scenario on VGA monitor of 640 x 480 resolution at 60 fps. Thus, the proposed architecture uses the hardware resource more effectively without a significant reduction in the accuracy of detection.

REFERENCES

[1] Tang, Z., Miao, Z., Wan, Y. (2007). Background subtraction using running gaussian average and frame difference. In: 2007 International Conference on Entertainment Computing (ICEC), pp. 411-414. https://doi.org/10.1007/978-3-540-74873-1_50

[2] Jabri, S., Duric, Z., Wechsler, H., Rosenfeld, A. (2000). Detection and location of people in video images using adaptive fusion of color and edge information. Proc - Int Conf Pattern Recognition, pp. 627-630. http://doi.org/10.1109/ICPR.2000.902997

[3] Berger, M., Seversky, L.M. (2014). Subspace tracking under dynamic dimensionality for online background subtraction. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1274-1281. http://doi.org/10.1109/CVPR.2014.166

[4] Braham, M., Van Droogenbroeck, M. (2016). Deep background subtraction with scene-specific convolutional neural networks. In: International Conference on Systems, Signals, and Image Processing, pp 1-4. http://dx.doi.org/10.1109/IWSSIP.2016.7502717

[5] Culibrk, D., Marques, O., Socek, D., Kalva, H., Furht, B. (2007). Neural network approach to background modeling for video object segmentation. IEEE Transactions on Neural Networks, 18(6): 1614-1627. http://doi.org/10.1109/TNN.2007.89686

[6] Sivabalakrishnan, M., Manjula, D. (2009). An efficient foreground detection algorithm for visual surveillance system. International Journal of Computer Science and Network Security, 9(5): 221-227.

[7] Jacques, J.C.S., Jung, C.R., Musse, S.R. (2006). A background subtraction model adapted to illumination changes. In: Proceedings - International Conference on Image Processing, ICIP, pp 1817-1820. http://doi.org/10.1109/ICIP.2006.312599

[8] Stefano, L.D., Mattoccia, S., Mola, M. (2003). A change-detection algorithm based on structure and colour. In: Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance, pp. 252-259. http://doi.org/10.1109/AVSS.2003.1217929

[9] Ridder, C., Munkelt, O., Kirchner, H. (1995). Adaptive background estimation and foreground detection using kalman-filtering. In: International Conference on Recent Advances in Mechatronics (ICRAM 1995), pp. 193-199.

[10] Karmann, K.P., Brandt, A. (1990). Moving object recognition using an adaptive background memory. In: Proceedings Time Varying Image Processing, pp. 289-307.

[11] Toyama, K., Krumm, J., Brumitt, B., Meyers, B. (1999). Wallflower: principles and practice of background maintenance. In: Proceedings of the Seventh IEEE International Conference on Computer Vision, pp. 255-261. http://doi.org/10.1109/ICCV.1999.791228

[12] Jacques, J.C.S., Jung, C.R., Musse, S.R. (2005). Background subtraction and shadow detection in grayscale video sequences. In: XVIII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'05), pp. 189-196. http://doi.org/10.1109/SIBGRAPI.2005.15

[13] Hung, M., Pan, J., Hsieh, C. (2010). Speed up temporal median filter for background subtraction. In: 2010 First International Conference on Pervasive Computing, Signal Processing and Applications, pp. 297-300. http://doi.org/10.1109/PCSPA.2010.79

[14] Elgammal, A., Duraiswami, R., Harwood, D., Davis, L.S. (2002). Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. Proceedings of the IEEE, 90(7): 1151-1163. http://doi.org/10.1109/JPROC.2002.801448

[15] Stauffer, C., Grimson, W.E.L. (1999). Adaptive background mixture models for real-time tracking. In:

Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149), pp. 246-252. http://doi.org/10.1109/CVPR.1999.784637

[16] Kim, K., Chalidabhongse, T.H., Harwood, D., Davis, L. (2005). Real-time foreground background segmentation using codebook model. Real-Time Imaging, 11(3): 72185. http://doi.org/10.1016/j.rti.2004.12.004

[17] Barnich, O., Van Droogenbroeck, M. (2011). Vibe: A universal background subtraction algorithm for video sequences. IEEE Transactions on Image Processing 20(6): 1709-1724. http://doi.org/10.1109/TIP.2010.2101613

[18] Hofmann, M., Tiefenbacher, P., Rigoll, G. (2012). Background segmentation with feedback: The pixel-based-adaptive segmenter. In: 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 38-43. http://doi.org/10.1109/CVPRW.2012.6238925

[19] Kryjak, T., Komorkiewicz, M., Gorgon, M. (2014). Real-time foreground object detection combining the PBAS background modelling algorithm and feedback from scene analysis module. International Journal of Electronics and Telecommunications, 60(1): 61-72. http://doi.org/10.2478/eletel-2014-0006

[20] Jiang, H., Ardo, H., Owall, V. (2009). A hardware architecture for real-time video segmentation utilizing memory reduction techniques. IEEE Transactions on Circuits and Systems for Video Technology, 19(2): 226-236. http://doi.org/10.1109/TCSVT.2008.2009244

[21] Jiang, H.T., Ardo, H., Owall, V. (2005). Hardware accelerator design for video segmentation with multimodal background modelling. In: 2005 IEEE International Symposium on Circuits and Systems, pp. 1142-1145. http://doi.org/10.1109/ISCAS.2005.1464795

[22] Kristensen, F., Hedberg, H., Jiang, H., Nilsson, P., Owall, V. (2008). An embedded real-time surveillance¨ system: Implementation and evaluation. Journal of Signal Processing Systems, 52(1): 75-94. http://doi.org/10.1007/s11265-007-0100-7

[23] Genovese, M., Napoli, E. (2014). Asic and FPGA implementation of the gaussian mixture model algorithm for real-time segmentation of high definition video. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 22(3): 537-547. http://doi.org/10.1109/TVLSI.2013.2249295

[24] Evangelio, R.H., Patzold, M., Keller, I., Sikora, T. (2014). Adaptively splitted gmm with feedback improvement for the task of background subtraction. IEEE Transactions on Information Forensics and Security, 9(5): 863–874. http://doi.org/10.1109/TIFS.2014.2313919

[25] Cherian, S., Singh, C.S., Manikandan M (2014). Implementation of real time moving object detection using background subtraction in FPGA. In: 2014 International Conference on Communication and Signal Processing, pp. 867-871. http://doi.org/10.1109/ICCSP.2014.6949967

[26] Gorelick, L., Blank, M., Shechtman, E., Irani, M., Basri, R. (2007). Actions as space-time shapes. Transactions on Pattern Analysis and Machine Intelligence, 29(12): 2247-2253.