

Systolic Video Stream Object Detector Using FPGA

Dr. Shefa A. Dawwd

shfadawwd@yahoo.com

Computer Engineering Department, College of Engineering, University of Mosul, Iraq.

Ula T. Salim

Abstract

Object detection is important operation used in multiple applications such as computer vision, image and video processing, security, artificial intelligent, and several other areas. However, in these applications, it is not easy to realize real-time frame rates and fast invariant detecting function under changing object states such as position and size using software implementations. So that to solves these problems and speed up the highly intensive calculation required, In this paper simple and efficient template matching algorithm architecture of a video streaming application for object detection is proposed, it is based on using Sum of Absolute Differences (SAD) with Pyramid Sum of Absolute Differences (PSAD) as similarity measures and a systolic array design using sliding window operation, where each video frame is divided into slides and feeds through the window by using a suitable first in first out(FIFO) buffers instead of the sliding window across the video frame. The implementation operation is done by using combination of software and hardware co-design that is based by using pipelining technique, data recirculation , and single instruction multiple data (SIMD) operations. The results for both SAD and PSAD algorithms showed the best match can be found at the template (window) size is 19×19 bits/pixel and with accuracy detection rate of 100%.

Keywords: FIFO, FPGA, Object detection, Pipeline, PSAD, SAD, Sliding window, Systolic array, Template matching, Video stream.

كاشف الكائن النبضي الفيديوي باستخدام البوابات المنطقية القابلة للبرمجة

علا طارق سالم

د. شفاء عبد الرحمن داوود

قسم هندسة الحاسوب / كلية الهندسة / جامعة الموصل- العراق

الخلاصة

كشف الكائن هي عملية مهمة وتستخدم في عدة تطبيقات مثل الرؤيا بالحاسوب ، معالجة الصور والفيديو، الأمن والعديد من المجالات الأخرى ومع ذلك في هذه التطبيقات ليس من السهل تحقيق عملية كشف سريعة وثابتة عند تغيير موقع وحجم صورة الكائن باستخدام عمليات التنفيذ البرمجي، لذلك ، لحل هذه المشاكل ولتسريع الحساب المطلوب بدقة عالية ، تم في هذا البحث اقتراح معمارية بسيطة وكفاءة لمطابقة القالب والمبنية باستخدام (SAD) مع (PSAD) كمقاييس لمطابقة القالب وتم تصميم المصفوفة النبضية باستخدام عملية إزاحة النافذة حيث أن كل إطار صوري يقسم إلى شرائح ترسل إلى النافذة باستخدام مخازن مناسبة (FIFO) بدلا من ترحيف النافذة على الإطار الصوري. وتمت عملية التنفيذ باستخدام التصميم المشترك لمجموعة من البرامج والمكونات المادية المبنية باستخدام تقنيات خط الأنابيب ، إعادة تدوير البيانات ، وعمليات بإيعاز واحد وبيانات متعددة (SIMD) . وأظهرت النتائج أن أفضل تطابق يمكن الحصول عليه عندما يكون حجم القالب 19×19 bits/pixel وبنسبة كشف 100%.

Received: 20 – 7 - 2013

Accepted: 13 – 10 - 2013

1 . Introduction

Video streaming can be defined as the process of applying some operations on video frames, streamed on a system picture by picture . The frames are usually transmitted pixel by pixel and therefore can be processed on a pixel by pixel basis. Most of the video streaming systems are built on a chain structure made up of a set of blocks. The first block on the chain is in charge of capturing the video frames, while the last block usually deals with the rendering of the frames. Between the first and the last modules, several operational blocks can be used according to the implemented algorithm [1].

There are different processes performed by Video processing applications such as video image enhancement [1], and object detection [2].

object detection is a process to know whether an object of interest is found in an image/video frame or not, regardless of its size, orientation, and environmental conditions. There are several algorithms used to perform detection operation, each of which has its own advantages and disadvantages and depend on the goal [3]. One of these algorithms is the template matching.

Template matching is a method for locating the position of a given small sub-image inside a large image. It is important method frequently used in the applications of pattern recognition, object detection, image registration, and image sequence coding.

In general, the matching process includes sliding a template image over a search area, measuring the similarity between the template and the current search area, and locating the best match position. One of the Major similarity measures which are widely used in template matching is the sum of absolute differences (SAD) [5]. which is widely used for image compressing and object tracking. Actually, this measurement is widely used because it's simple, very easy to implement [6], take less number of operations[7] and can give high localization rate where the image is with high illumination variation. But it has disadvantage that it may be affected by variation in background [6].

Implementation operation can be in hardware or in software according to the aims. Now researchers direct their attention to parallel processing paradigms such as Graphics Processing Units (GPU) and High Performance Computing (HPC) systems [3].

Although software provides a great degree of flexibility, but still constraint and not fast enough to realize the challenging computation required in video application. ASICs can be used to implement the computational demanding parts, however ASIC does not provide the flexibility needed in many systems. Today's FPGAs offer enough logic to implement many modules of a video streaming chain [1].

The last few years have seen an unprecedented effort by researchers in the field of image processing based object detection operation using FPGA. In 2000 K., Hidai, et-al [4] propose a novel face detection method that can be used for practical intelligent environment and human-interactive robot and the method solved position, size, and brightness of changeable face image by using combining correlation-based pattern matching, histogram equalization, skin color extraction, and multiple scale images generation. The Experimental results show effectiveness of the proposed idea.

In 2002 J., Gause, et-al [5], presents A generic systolic array architecture reconfigurable computing approaches for a Shape-Adaptive Template Matching (SA-TM) method to retrieve arbitrarily shaped objects within images or video frames.

In 2012 N., Dawoud , et-al [6] proposed a fast template matching technique based on Optimized similarity measurement metrics namely: Sum of Absolute Difference (OSAD) and Sum of Square Difference (SSD) to overcome the drawback of NCC. This similarity measure application on face localization and with the other similarity measurements .the

results show the highest performance of OSAD compared with other measurements and the improvement of OSSD comparing with SSD as well.

In 2012 F., Alsaade, et-al [7] study and propose a combined approach to enhance the performance of template matching system using image pyramid in conjunction with the Sum of Absolute Difference (SAD) and Normalized Cross correlation (NCC) similarity measure and applied this approach on color and gray scale images with different sizes and different illumination. The results show that the proposed method offered improvements to the overall performance, execution time and accuracy under the noisy and clean data conditions.

In 2012 J., Fowers, et-al [8] compared performance and energy of sliding window applications when implemented on different accelerators under a variety of different use cases and present optimization strategies and analyze using cases where each device is most effective. The results show that FPGAs can achieve speedup of up to 11x and 57x compared to GPUs and multicores, respectively, while also using orders of magnitude less energy.

In this paper we propose a systolic architecture for object detection intended to be used for a video stream processing and practical intelligent environments. The Key idea of this architecture is combination of using sliding window operator, SAD based matching and pyramid downscale resolution to realize respectively invariant position and size of object situation, then the proposed architecture implementing on FPGA to realize fastest speed. The rest of the paper is organized as follow: Section 2 discusses methods and solutions. Section 3 describes hardware design. Section 4 describes an results and discussion for the implementation based upon solve the solutions. Section 5 is conclusion.

2. Methods And Solutions

2.1 Sliding Window

The sliding window operator usually process one pixel of the image at a time, changing its value by some function of a local region of pixels (covered by the window). The operator moves over the image to process all the pixels in the image. Each pixel in the output image is produced by sliding an $N \times M$ window over the input image and computing an operation according to the pixels included in the window with the chosen window operator. The result is a pixel value that is assigned to the center of the window in the output image, as shown below in Figure (1) [9].

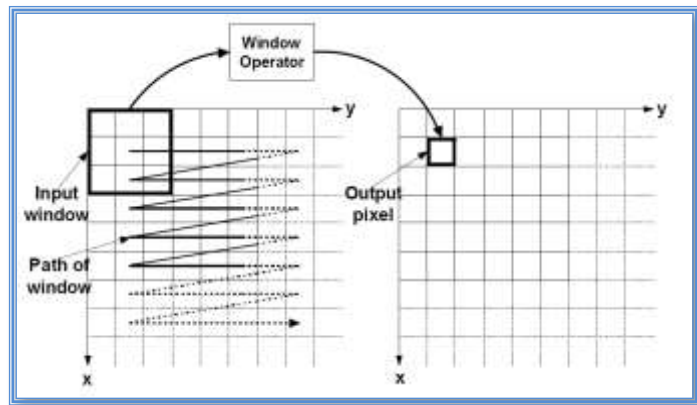


Figure (1): Conceptual example of 3x3 window operator

2.2 Sum of Absolute Differences (SAD)

The sum of absolute difference (SAD) is a simple algorithm for measuring the similarity between template image and sub-image in source image [7]. It based on using sliding window. The window function for this method calculates the absolute difference between each window pixel and template pixel, and then accumulates the differences for the entire window.

So, The search image or frame which consists of $W \times H$ pixels and the template object consisting of $w \times h$ pixels that can have any shape. SAD can be calculate using the equation

$$SAD(x, y) = \sum_{k=0}^{(W-1)} \sum_{l=0}^{(H-1)} |S(x+k, y+l) - T(k, l)| \quad (1)$$

The output upon completion is a two dimensional data structure of dimensions $(x-n+1) \times (y-m+1)$, which corresponds to the total number of windows.

For the entire image there are $(W-w+1) \times (H-h+1) \times w \times h$, absolute distance calculations need to be computed and accumulated. The matching should be found at a position (x, y) where $SAD(x, y)$ is minimum and also smaller than a certain threshold determined by the user [5].

2.3 Pyramid Sum of Absolute Differences (PSAD)

One of the multi resolution methods that based on taking average of neighbor pixels which is used to find the object that may appear at any scale within the original image.

In general PSAD involve S generating several image copies of an original input image as pyramid in regular downscale form and performs the matching process against these generated images If an object is found in the input image, there must be in an generated image in which the object size is similar to the template size. Thus the matching process calculates minimum similarity score and succeeds to detect the object [7].

2.4 Systolic arrays (SAs)

Systolic Arrays are a form of Single Instruction Multiple Data (SIMD) machines which is consisting of a set of Processing Elements (PEs). Each of them connected to neighbors in a regular structured manner. Each operate on multiple data with a single instruction without any intermediate or previous results [10].

The function of each Processing Element receive data from all it's input, process it in pipeline form and pump the results through the network in a rhythmic and continual manner. But communicating with the outside world can be allowing only for the boundary processing elements in the network can be. This leads to eliminate the classical memory, I/O bandwidth bottleneck [12].

Pure-SAs do not have a global data bus, and memory data enters the array through a single PE and is propagated to other PEs every clock cycle until the last PE receives the data. this advantage can be represent as seen in Figure 2. Figure 2 (a) shows how a single PE would have to access memory for every iteration, while Figure 2 (b) demonstrates how the data from memory can be propagated to every node instead of generating a memory access every time[11].

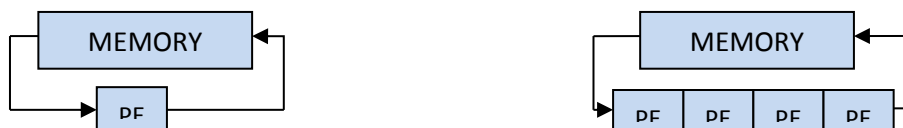


Figure (2) (a) Single Processing Element with a memory access in every execution loop.

2 (b) Multiple Processing Elements with single access to memory.

Figure (2): Comparison between a single PE and multiple PE in a systolic array structure.

The type of PE is not the only variable when designing a systolic array; there are also different topologies that can be implemented depending on the particular application .some of the possible topologies for systolic arrays :rectangle ,triangle and linear [10].

Based on the systolic features Network, Homogeneity, Locality, Boundary, Modularity, Rhythm, Synchrony, Extensibility, Pipelineability, each systolic algorithm implemented on the processing element passes through three distinct processing phases, Which are Data input, Algorithm processing and Data output. All three phases constitute what is known as a systolic cycle [13].

3. Hardware Design

Two architectures of high flexibility are proposed in this paper. Sliding window, Pipelining, parallelism and systolic techniques are used to make the computation performs in high efficiency. First architecture for the 2D-SAD algorithm and the second architecture for the 2D-PSAD with down scale algorithms are to be implemented in this paper.

In both architectures, for processing purposes, the straightforward approach is to store the entire input image into a frame buffer, accessing the neighborhood pixels and applying the function as needed to produce the output image. If real-time processing of the video stream is required, $M \times N$ pixel values are needed to perform the calculations each time the window is moved and each pixel in the image is read up to $M \times N$ times. Memory bandwidth constraints make obtaining all these pixels each clock cycle impossible unless some form of local caching is performed. Input data from the previous M -rows can be cached using a shift register (or FIFO buffer) when the window is scanned along subsequent lines. This leads to the block diagram as shown in Figure 3.

Note that, instead of sliding the window across the image, the above implementation now feeds the image through the window generator.

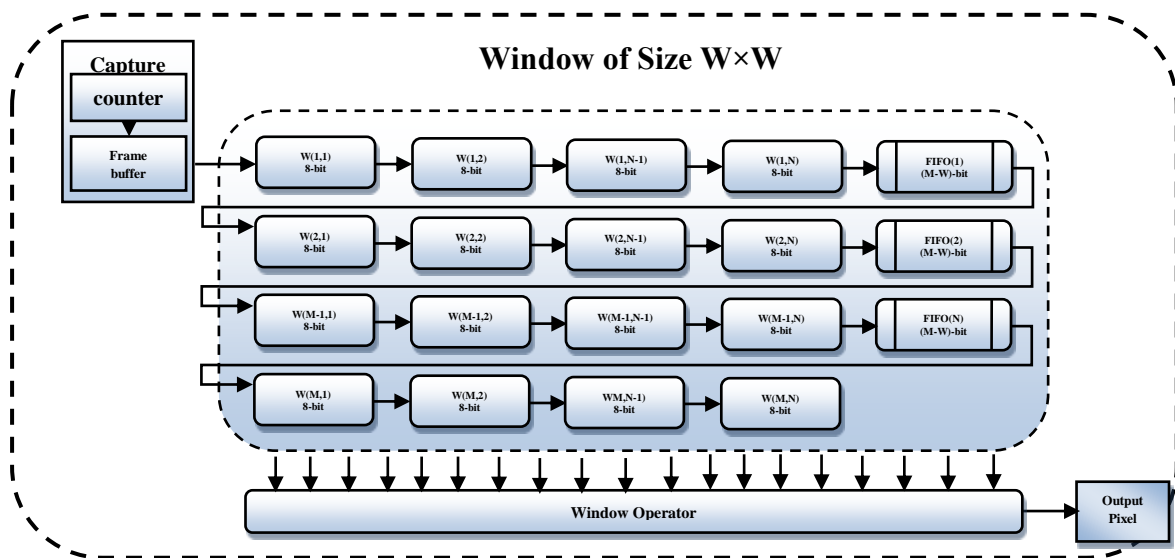


Figure (3): general block diagram for hardware implementation of systolic array for sliding window circuit

To use the sliding window in implementing 2D-SAD algorithm as seen in figure (4), the controller initially starts a sequential read from the Block ROM Memory (BROM) that stores the image. The window registers stores arriving pixels in a FIFO corresponding to each of the M row. After fulling the n th FIFO, a new window is created in every cycle. At this point, the system start to subtracts every corresponding pair of window and kernel pixels, and then takes the absolute value, which is stored in a register. The data then passes all $W \times W$ absolute differences to a pipelined adder tree that contains registers at each level. Then the adder tree

results the output of size equals input pixel width+log₂(w*w) bits. This result generates carry bits at each level of the tree to prevent the overflow occurrence, then this pixel enter the comparator circuit to find minimum value that represent the best match. After the initial pipeline latency, the system produces an output each cycle. The memory controller writes it to Block RAM Memory (BRAM).

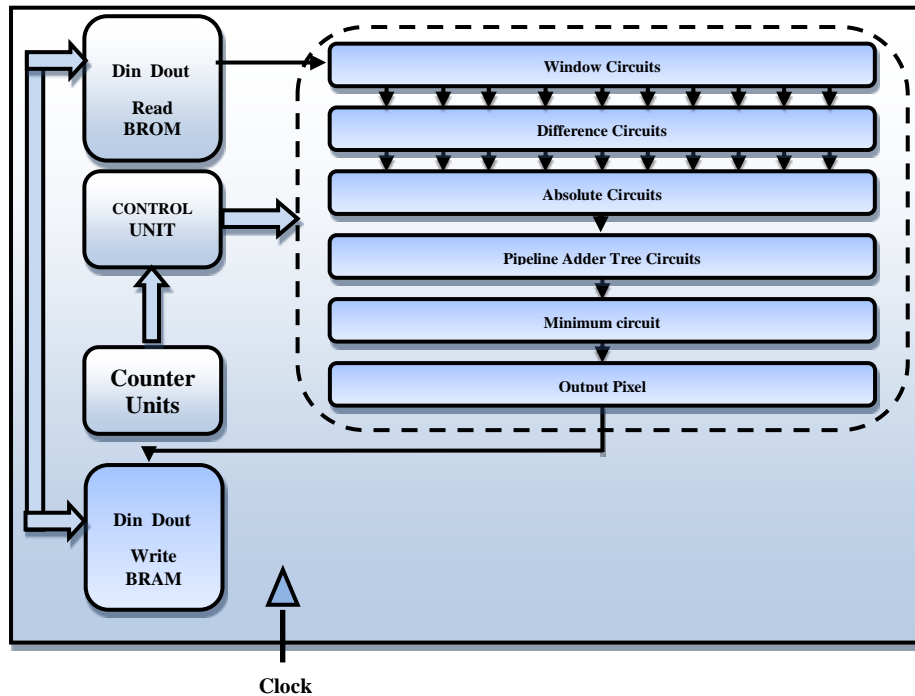


Figure 4. The architecture of 2D SAD using moving window operator

To improve the performance and detection operation for SAD algorithm and to make the system invariant with translation and scale, the downscaled image operation is applied. The idea behind that is illustrated in fig. (5).

In figure 5 one can see that each image scale enters the SAD block, then the output of the SAD for each scale enters to the comparator circuits to find the best match.

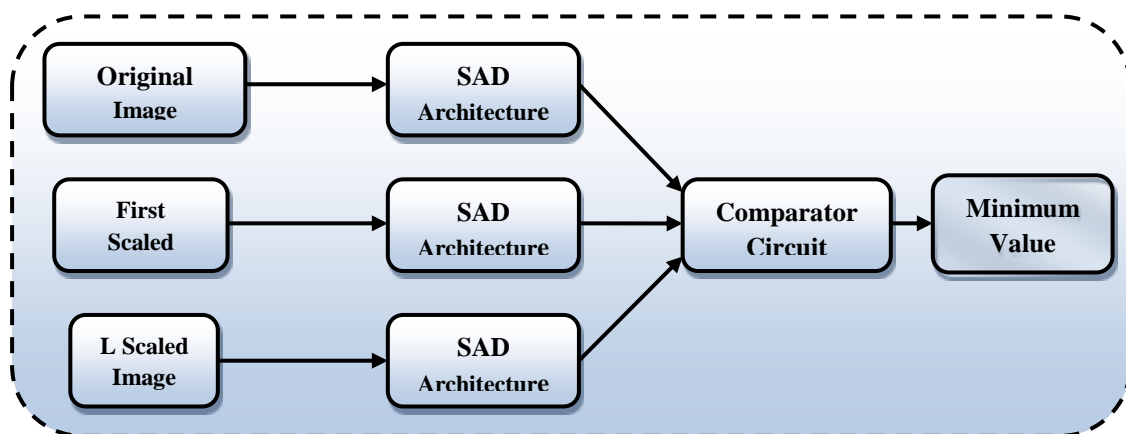


Figure (5): The architecture of 2D PSAD for using window operator

4. Results And Discussion

The design for the both mentioned architectures is modeled by using VHDL language and implemented on Spartan3-E FPGA chip through ISE12.1 program as a target technique and tested on the database consisting of 116 frames. The testing database is extracted from [14].

The hardware for both architectures that presented in the previous section is dedicated to window-based operation. The first architecture adapted for image of size 160x120x8 bit/pixel and the second architecture is adapted to image of three scales each of size 160x120x8 bit/pixel, 80x60x8 bit/pixel and 40x30x8 bit/pixel. For both architectures the window consisting of 8-bit grayscale template of varying sizes as 13x13, 16x16, 19x19 and 25x25. The main objective of these two architectures is to achieve real time operation; therefore performance analysis is required.

1.1. Performance metrics, limitations and constraints

1.1.1. Execution time

The time required to process an input image frame within a window-based operator is composed of two main times the latency time and the parallel processing time [15][16].

$$T = \text{Latency Time} + \text{Parallel Processing Time} \quad (2)$$

where:

$$\text{Latency Time} = [(L \times (w - 1)) + w] \times \frac{1}{f}$$

$$\text{Parallel Processing Time} = \text{Reading Window Time} + \text{Execution Time.}$$

$$\text{Reading Window Time} = [M - (w - 2)] \times N \times \frac{1}{f}$$

L : FIFO buffer length . M : no. of image rows . N : no. of image columns .
w : window length . f : FPGA operating frequency.

Execution Time: the time that is taken to perform difference value, absolute value, sum of the value and minimum value.

The total time required to process one frame video using the both proposed architectures can be seen in figure 6.

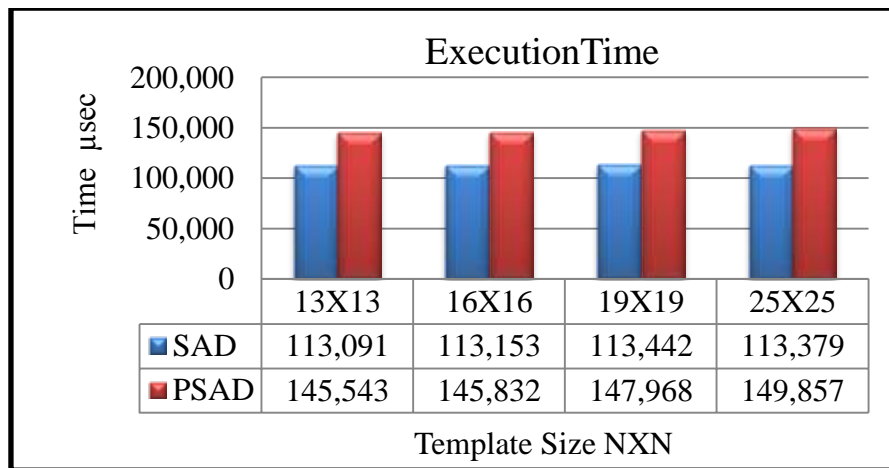


Figure (6): Performance of the SAD and PSAD implementations measured in Execution Time.

1.1.2. Speed

Speed can be measured by the maximum allowable clock frequency. For both architectures in Figure 7 one can see that with the increasing of window size , for SAD architecture the maximum allowable frequency can still be constant with the value of 174.004MHz while for PSAD architecture the maximum allowable frequency between 132.293 and 135.007 MHz .

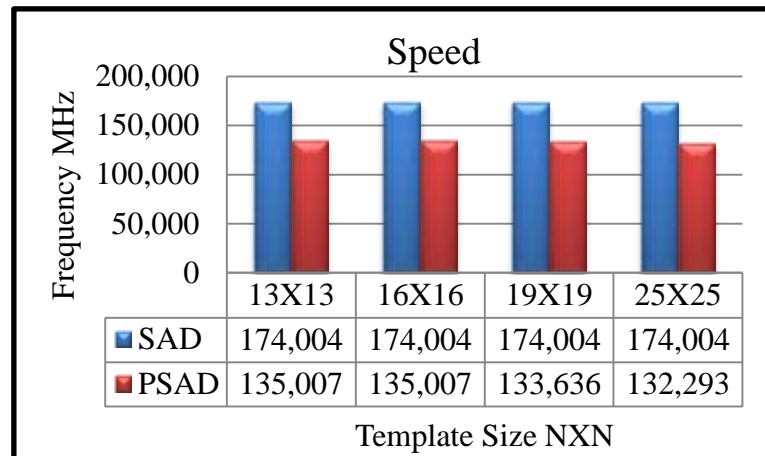


Figure (7): Performance of the SAD and PSAD implementations measured in Speed

1.1.3. Resources Utilization

Area is measured by number of slices occupied. In Table 1, For both architectures One can see that with the increasing of window size, the total silicon resources cost increases. That is because the parallelism (operations that implement simultaneously) increases. When the window size becomes larger than 13x13, the target FPGA cannot fit with these larger windows. Thus new model of larger gates should be used.

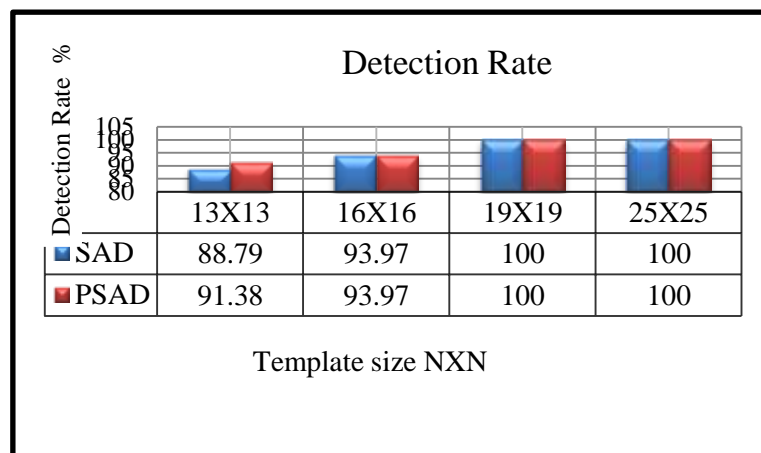
Table (1): Performance of the SAD and PSAD implementations measured in resources utilization.

Method	Window Size	Number of Slices		Number of Slice Flip Flops		Number of 4 input LUTs	
		Count	Usage %	Count	Usage %	Count	Usage %
SAD	13x13	4,518	97%	7,314	79%	5,925	64%
PSAD		12,690	273%	21,936	236%	17,692	190%
SAD	16x16	6,186	133%	10,699	115%	8,704	93%
PSAD		18,527	398%	32,091	345%	26,101	280%
SAD	19x19	8,489	182%	14,722	158%	12,058	129%
PSAD		25,421	546%	44,161	474%	36,169	388%
SAD	25x25	14,435	310%	24,675	265%	20,412	219%
PSAD		43,297	930%	74,119	796%	61,135	657%

1.1.4. Detection rate

For both architectures figure 8 that show when window size increases the detection rate increases and using the downscale technique improve the detection rate.

Figure (8): Performance of the SAD and PSAD implementations measured in detection rate



1.1.5. Frame rate

As shown in figure 9. The frame rates achieved with variant window sizes for the same image size are approximately equaled. That is because the circuit computes one window each cycle regardless of kernel size. For larger kernel sizes, the system complexity increase and FPGA performance would decrease linearly.

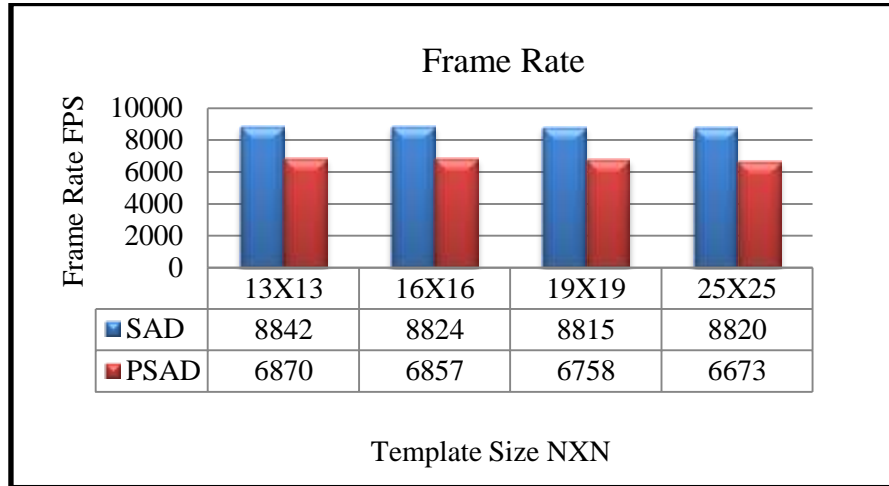


Figure (9): Performance of the SAD and PSAD implementations measured in Frame rate.

Table (2): comparison between the related work and the presented work using PSAD algorithm

	Alsaade	Hidai	Presented Work
Image Size	240×320	160×120	160×120
Template Size	32×48	16×16	16×16
Number of Input Images	1 Frames	48 Frames	116 Frames
Number of Scales	3	10	3
Detection Rate	100	21.1	93.97

Table (3): comparison between the related work and the presented work using SAD algorithm

	Fowers	Presented Work
FPGA	Altera Stratix III E260	Spartan 3E
Image Size	1920×1080	160×120
Speed (Frequency MHz)	100 and 115 MHz Depend on template size	174.007 MHz For all templates

Both architectures showed the best detection for a video stream at window of the size 19×19 bit/pixel and the results can be viewed in figure 10.











Template	Input Image	Detection using SAD	Detection using PSAD
			
Small Scale			
<p>19×19</p> 			
Medium Scale			
			
Large Scale			

Figure (10): object detection operation in video stream using SAD and PSAD Algorithms

5. Conclusion

Object detection implementation is an important and critical task in real time a video stream processing . In this work flexible and high performance systolic architecture object detector invariant against position and size is presented . The experimental results show that when using downscaled technique with SAD that known as (PSAD) , the overall performance of the system is improved and the invariant feature against translation and scaling is increased and the real time requirement is also preserved .

6. Reference

1. Bobda, C., Ahmadiania, A., Majer, M., and Ding, J., Teich, J., "Modular video streaming on a reconfigurable platform", In Proceedings of the IFIP International Conference on Very Large Scale Integration, Perth, Australia, 2005, pp. 103-108.
2. Mc Curry, P., Morgan, F., and Kilmartin, L., "Xilinx FPGA implementation of a pixel processor for object detection applications", In Proceeding of the Irish Signals and Systems Conference, UCD, Dublin, 2000, pp. 404-411.
3. Kyrkou, C., "Embedded Hardware Architectures for Object Detection", M. Sc. Diss. University of Cyprus, 2010, p. 116.
4. Hidai, K., Mizoguchi, H., Hiraoka, K., and Tanaka, M., "Robust face detection against brightness fluctuation and size variation." Intelligent Robots and Systems, 2000. (IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on. Vol. 2. IEEE, 2000, pp. 1397-1384.
5. Gause, J., Peter Y. K. Cheung, and Luk, W, "Reconfigurable shape-adaptive template matching architectures", Proc. IEEE Symp. on Field-Programmable Custom Computing Machines, IEEE Computer Society Press, V, 2002, pp. 98-107.
6. Dawoud, N., N., Samir, B., B., and Janier, J., "Fast Template Matching Method Based on Optimized Metrics for Face Localization", Proceedings of the International Multi Conference of Engineers and Computer Scientists, Vol. 1., March 2012.
7. Alsaade, F., "Fast and accurate template matching algorithm based on image pyramid and sum of absolute difference similarity measure". Research Journal of information Technology Vol. 4, No.4, 2012, pp. 204-211.
8. Fowers, J., Brown, G., Cooke, P., and Stitt, G., "A performance and energy comparison of FPGAs, GPUs, and multicore for sliding-window applications", in Proc. Int. Symp. Field Programmable Gate Arrays, 2012, pp. 47-56.
9. Johnston, C. ,T., Gribbon, K.,T., and Bailey, D., G. "Implementing Image Processing Algorithms on FPGAs" Proceedings of the Eleventh Electronics New Zealand Conference, ENZCon'04, Palmerston North, November 2004, pp. 118-123.
10. Juan, A. Field-Programmable Gate Array Implementation of a Scalable Integral Image Architecture Based on Systolic Arrays. Diss. Utah State University, 2011.
11. H. Kung, "Why systolic architectures" IEEE Computer, Vol. 15, No. 1, 1982, pp. 37-46.
12. N. Petkov, Systolic Parallel Processing, North-Holland, 1993.
13. Zajc, M., R. Serbec, and J. Tasic. "Efficient SoC design with homogeneous processor arrays." Canadian Journal of Electrical and Computer Engineering 26.3/4 (2001): 91-98.
14. <http://www-prima.inrialpes.fr/FGnet/data/03-Pointing/> .
15. G. Saldana and M. Arias-Estrad, "FPGA-Based Customizable Systolic Architecture for Image Processing Applications", IEEE International Conference on Reconfigurable Computing and FPGAs (ReConFig 2005) ISBN 0-7695-2456-7, 2005.
16. G. Deepak, R. Mahesh and A. Sluzek, "Design of an Area-Efficient Multiplierless Processing Element for Fast Two Dimensional Image Convolution", IEEE 2005.

The work was carried out at the college of Engineering. University of Mosul