ISSN 1996-3343

Asian Journal of Applied Sciences

http://knowledgiascientific.com

Knowledgia SCIENTIFIC
A Place to Publish Outstanding Research

Asian Journal of Applied Sciences 7 (1): 1-12, 2014
ISSN 1996-3343 / DOI: 10.3923/ajaps.2014.1.12

Hardware Implementation of PSO-Architecture for Image Segmentation on FPGA

[1]Fayçal Hamdaoui, [2,3]Anis Sakly and [1,3]Abdellatif Mtibaa
[1]Laboratory of EμE, Faculty of Sciences of Monastir, Monastir, Tunisia
[2]Reseach Unit, Industrial Systems Study and Renewable Energy (ESIER),
[3]National School of Engineers at Monastir, Av Ibn ElJazzar 5019, Monastir, Tunisia

Corresponding Author: Fayçal Hamdaoui, Laboratory of EμE, Faculty of Sciences of Monastir, Monastir, Tunisia

ABSTRACT

Segmentation prepares images to the detection, recognition and classification stages. These stages are widely used in various fields such as robotics and medical imaging. So, it becomes very important for researchers to succeed the segmentation step and it still a relevant research area in computer vision. In this study, a proposed hardware architecture based on PSO algorithm for images segmentation using Xilinx System Generator (XSG) was implemented on Field Programmable Gate Array (FPGA). The use of the visual development process models of Simulink facilitates the Register-Transfer Level (RTL) simulation and validation such as the synthesis of Very high-speed integrated circuits Hardware Design Language (VHDL) code. This architecture was implemented into FPGA, so it gives the advantages of the real time segmentation; a very fast speed to reach the threshold for segmentation and output a segmented binary image. Also, only 12% of logic resources and 38% of buffers are used to realize this architecture. Finally, performances of the proposed method are validated and demonstrated using a set of Benchmarks and medical images.

Key words: Segmentation, PSO algorithm, XSG, FPGA, real time image processing

INTRODUCTION

Digital image segmentation aims to identify the objects from the background in images. Also, it attempts to partition the image into semantically meaningful regions.

Digital image segmentation is, without doubt, considered as the most important low level vision operations. It has several leading areas application such as the computer vision (Bernd, 2000; Forsyth and Ponce, 2002), robotics and autonomous system (Hechri *et al*., 2011; Eklundh, 1998), benchmarks imaging (Lempitsky *et al*., 2009; Couprie *et al*., 2009) and medical imaging (Hamdaoui *et al*., 2013a, b). Therefore, hundreds of segmentation approches have been developped by researchers in the few last years to solve the segmentation task.

Genetic Algorithm (AG) (Maulik and Bandyopadhyay, 2000), Ant Colony Optimizatio (ACO) (Han and Shi, 2006), Ant Bee Colony (ABC) (Karaboga and Ozturk, 2011) and Particle Swarm Optimization (PSO) (Hamdaoui *et al*., 2013a; Holland, 1992) are metaheuristic algorithms that have been widely applied in the litterature. However, it is well known that and hardware implementation of segmentation algorithm on FPGA still not yet developed and inefficient because of the difficulty of designing hardware architectures. Also, processor's FPGAs have a minimum

resolution of number of bits. This requires a long computation time for image processing applications. These inconveniences are insignificant compared to the advantages provided by the implementation of FPGAs' architectures (Ladgham *et al.*, 2012). DSPs integrated into FPGA are the solution to solve this problem. They are digital processors dedicated to the calculation operations of image processing. Processors utilize instruction-level parallelism that allows the execution of multiple instructions simultaneously (Hamdaoui *et al.*, 2012). Other advantages of the implementation of the segmentation are the real-time, portability and robustness.

In this study, it is very important to benefit from the advantages of FPGAs by developing PSO algorithm of images segmentation using XSG tool. This architecture shows better results than the conventional variants of segmentation.

MATERIALS AND METHODS

Materials: This study aims to implement image segmentation based PSO algorithm into FPGA. For this, materials needed are:

- **MATLAB 7.9 (32-bit):** Necessary needed to use Simulink
- **The XSG software tool:** Its library resources are used for FPGA
- **ISE XILINX 12.3 environment:** Used to generate VHDL code, to synthesize the code and finally generate the Bitstream code
- **Personal computer:** 2.96 GHz CPU, 4G Ram and Windows 7 system
- **Virtex V Xilinx ML507 FPGA:** XC5VSX35T -1ff556 platform

Xilinx system generator design flow: In this study, Xilinx System Generator (XSG) blocks are used to implement architecture of images segmentation based on PSO algorithm. XSG is an industrial leader tool used to develop high-level designs with high-performance systems for FPGAs. It is a toolbox developed by Xilinx ISE Design Suite and can be integrated into Simulink of MATLAB. XSG lets developers with little design experience quickly create systems and production quality architecture for highly parallel FPGA (XSG, 1998).

So, XSG is a key technology development to briefly work Simulink projects in an FPGA. It reads files (Mdl) created on Simulink models using the additional library and generates VHDL files. This technology enables the reduction of design cycles by using a hardware system in the development environment that share the same algorithm representation. Figure 1 shows the broad flow design Xilinx System Generator. As already mentioned, you can then move to the configuration file to program the FPGA (Delva *et al.*, 2008).

Concluding, the XSG tool is used to produce a model that will immediately run on the hardware once completed and validated (Mailloux, 2008).

Figure 1 explains well the benefits of using the XSG tool for segmentation based PSO application. Firstly, starting by implementing the hardware architecture using basics resources integrated with SIMULINK and also .m files MATLAB. Once completed, XSG automatically generates the VHDL code. After that using the ISE tool that is compatible with XSG, it is possible to generate the bitstream code. Similarly and using the same tool, simulation results can be done using ModelSim. Finally, by XILINX CHIPSCOP PRO tool, it will be easy to download the bitstream code already generated into the FPGA target.

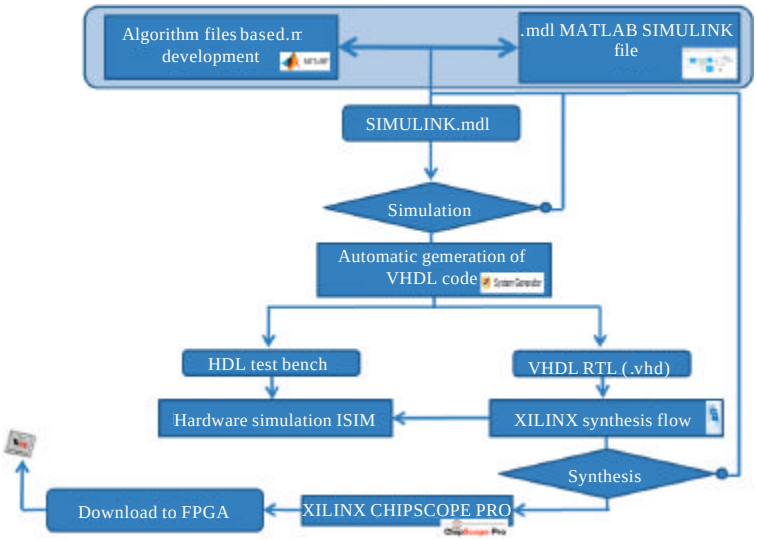

Fig. 1: Xilinx system generator (XSG) design flow

SEGMENTATION BASED PARTICLE SWARM OPTIMIZATION

The particle swarm optimization PSO is a new class of algorithm for searching inspired by social behavior of animals (Reynolds, 1987) that was proposed by Kennedy and Eberhart (1995) to solve problems with continuous variables. In groups of animals, each individual moves according to its own knowledge that can be accessed by moving its neighbors (Kennedy, 1997; Eberhart and Shi, 2000).

In PSO, each particle is characterized by its own position vector and velocity vector. The movement of these vectors in the search space is controlled by the following recursive Eq. 1 and 2:

$$v_{im} = wv_{im}+c_1*rand1*(p_{im}-x_{im})+c_2*rand2*(g_{gm}-x_{im}) \quad (1)$$

$$X_{im} = x_{im}+v_{im} \quad (2)$$

where, x_{i1} is the i^{th} position of the particle of the swarm, v_{im} the velocity of this particle, p_{im} the best previous position of the i^{th} particle, p_{gm} is the best position of particle in the swarm, $1 \leq m \leq M$ with M is the search space, rand1 and rand2 are the two independents random number with uniform distribution in the range [0, 1], c_1 and c_2 are two constants positives accelerations coefficients called cognitive and social parameter, respectively, w is called inertia weight and it is used to control the balance between exploration and search space exploitation.

The reasons behind their use are mainly in their ability to explore large and complex search spaces. PSO does not have genetic operators like crossover and mutation like with the Genetic Algorithm technique. Also, particles update themselves with the internal velocity to mode faster convergence to the best search value. Another advantage of using this algorithm is the less number of parameters to tune so less computation time to spend. Particles used in the PSO algorithm have memory, which is important to the algorithm and they don't die.

PSO is based on a number of particles that forming a swarm moving around in the search space to found the best solution. Each particle of the swarm is considered as one in N-dimensional space

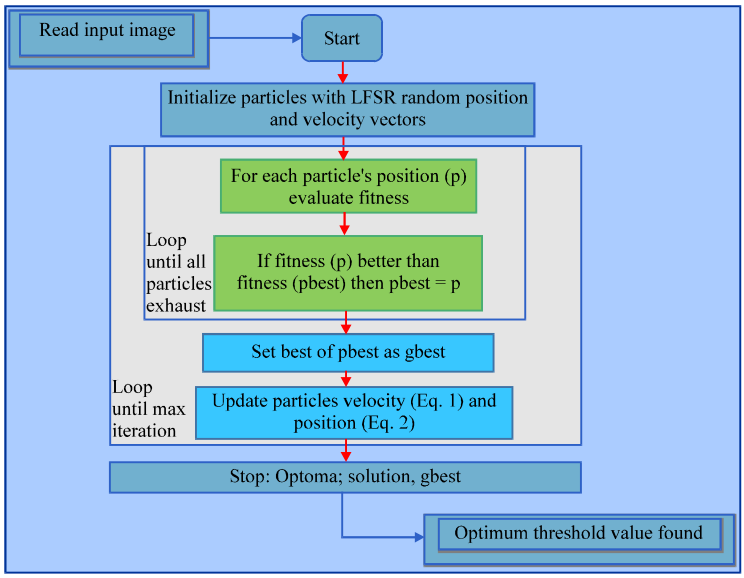

Fig. 2: Flow chart used for PSO's algorithm

which adjusts its coordinate in refers to its own coordinate as well as the coordinate of other particles called owned and others experiences. The principle is very simple and it is donate in Fig. 2.

Let particle swarm fly towards the best position in the searching space and let remember each particle's best known position called p best and global (swarm's) best known position called g best. For each particle, it keeps track of its coordinates in the solution space which are determinate by updating with the best solution (fitness) that has achieved so far by that particle. This value is called particle best, p best. Another best value that is tracked by the PSO is the best value obtained so far by any particle in the neighborhood of that particle. This value is called g best.

PSO approach is based on the memory and the social interaction among individuals. In the general case, the fitness function allows determining the best position for a particle i to make moves from its current x_i, t to the next x_it+1. Moving process is depends on three stages:

- The current velocity v_i, t
- The best performance (evaluated by the objective function: fitnessf_i, t)
- The best position of its neighbors (g,t)

HARDWARE IMPLEMENTATION

In this study, the aim is to implement hardware architecture of images segmentation based on PSO. Once the image was uploaded from the compact flash to be processed by the FPGA processor, start interruption was active. First operation will be done is the space conversion from RGB to Gray scale. After that, segmentation interruptions make the PSO segmentation in process to have finally a binary image to be displayed. Figure 3 shows all stages.

Now, the hardware implementation of PSO algorithm will be explained with details. It is decomposed into three principle operations: Initialization of particles, fitness evaluation and update of particle's best position, global best position and the velocity. Each of these operations is implemented in a separate hardware block. Each hardware block will be described, starting by the initialization block, then the fitness block and to finish by the update block.

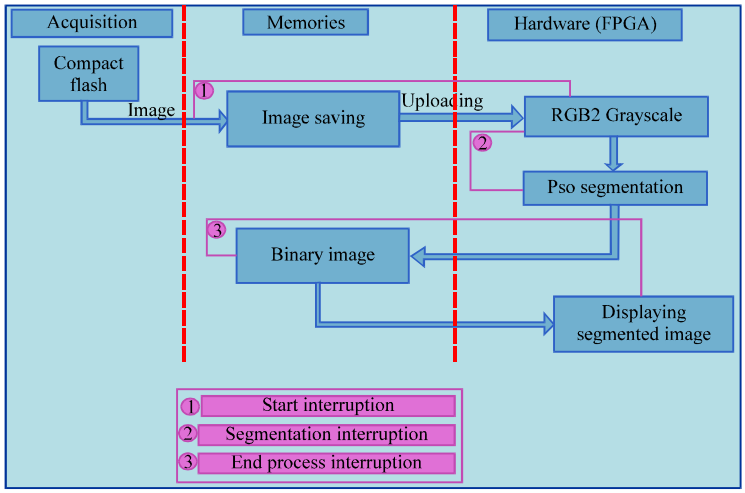

Fig. 3: Tasks allocation overview used by the hardware architecture of segmentation based on PSO algorithm

Initialization block:

L : The Histogram of the image
LP : The Histogram after normalization
X, V : Are respectively the position and the velocity. Those two vectors are initialized with random function
gbest and pbest : Are respectively best previous position of particle and best position of particle in the swarm and they are initialized with 0

$$x_{im} = x_{im}+(x_{max}-x_{min})*rand$$
$$v_{im} = v_{im}+(v_{max}-v_{min})*rand \quad (3)$$

Block implemented in MATLAB-Simulink using XSG in detailed in Fig. 4:

A software architecture based PSO often uses floating-point values. However, floating point operations typically require multiple logical resources for the same operation similar fixed point. In addition, it is common for FPGAs include a number of embedded multipliers that can be used to perform fixed-point multiplications without using programmable logic FPGAs. For these reasons, the implementation of the hardware architecture of segmentation based PSO uses the fixed-point representation for all values.

To succeed this proposed algorithm, two random numbers are required for each update speed. For i iterations and p particles will be equal to 2×p×i total number. The pseudo-random numbers are generated using the hardware tool PRNG. In all works, typically linear feedback shift registers (LFSR: Linear Feedback Shift Registers) and cellular automata (CA: Cellular automata) based PRNG are most commonly used (Zhang *et al.*, 2007). LFSRs are the most practical tool to implement and are used in most hardware implementations. An LFSR is illustrated in Fig. 5, the first bit in the left is calculated on the basis of the previous value and the generated bits in the register are shifted to the right.

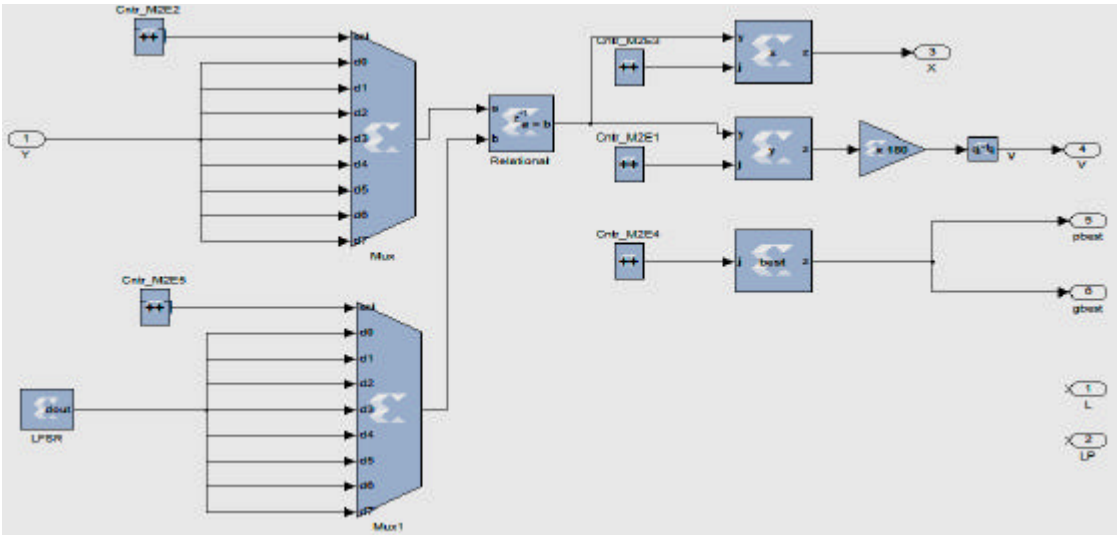

Fig. 4: Initialization block based LFSR architecture

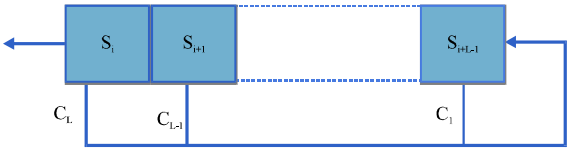

Fig. 5: Internal architecture of linear feedback shift block of LFSR

At each rising edge of the clock, the leftmost bit s_i is the output register and the others are shifted to the left, the new S_{i+L} bit set in the right cell of the register is given by a linear function as shown in Eq. 4 below:

$$S_{i+L} = c_1 S_{i+L} + c_2 S_{i+L-2} + \ldots + c_{L-1} S_{i+1} + c_L S_i \quad (4)$$

where, c_i are a binary coefficients.

Fitness function block: This study is founded on this fitness function which consists of the determination of the histogram. The appearance frequency of the pixels makes evaluation of the current position easily. Therefore the decision to go or not to the new position would be simpler.

Give the value of b, where $1 \prec b \prec h$ h is the histogram number and I_i is the intensity of pixel in the i^{th} position. The fitness function F(b) is given by the following equation:

$$F(b) = \frac{1}{n}\sum_{i=1}^{n}\delta(I_i - b) \quad (5)$$

With:

$$\delta(k) = \begin{cases} 1 \text{if } k = 0 \\ 0 \text{otherwise} \end{cases}$$

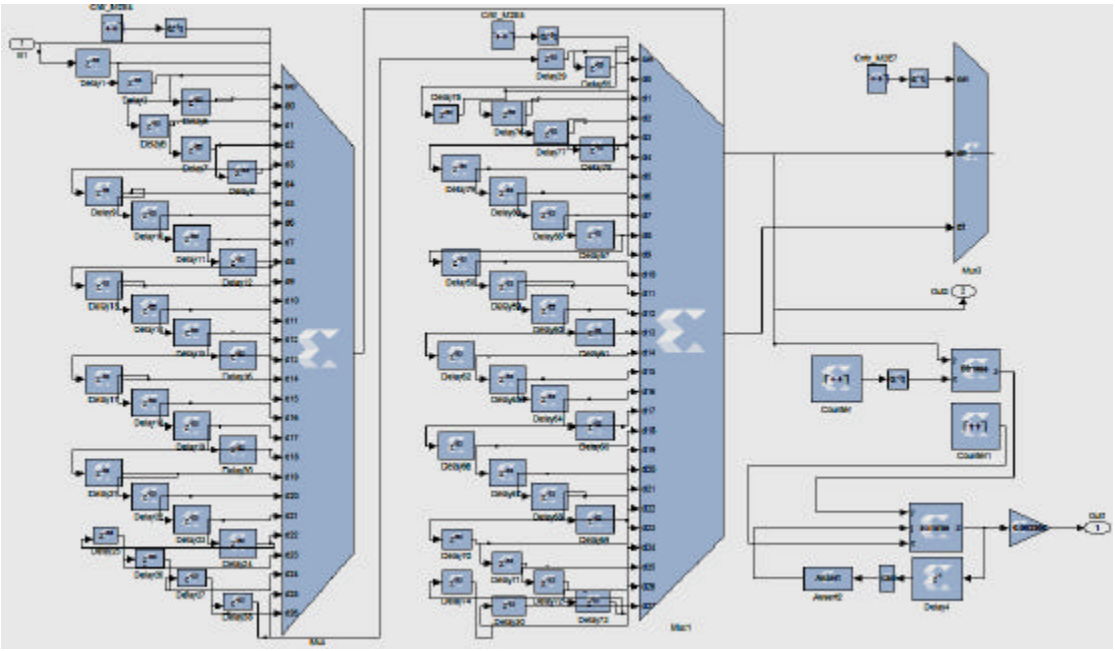

Fig. 6: Fitness function architecture implemented using XSG design tool

This fitness function based histogram was implemented on XSG and its architecture is showed on Fig. 6.

It should be noted that it is very difficult to achieve this block with System Generator. To do this, the determination of the histogram was limited on a part of swarm selection and not on the image entire. To achieve this architecture multiplexers, delay blocks, two counters, an accumulator and the comparator blocks are used for the slaving. In fact, the delay blocks and multiplexers are used to store the population 256 times so successive. Then, a counter and a comparator is used to compare each pixel of the population with the values between 0 to 255 of possible Gray scale level. Then, an adder, a comparator and a counter is used to determine the number of pixels at each Gray scale level value (histogram).

Update block: To update velocity and positions of particles, the concerned equations are already presented in the previous chapter, in Eq. 1 and 2. The realization of this block need the use of adders, subtracters, accumulators and multipliers to achieve the hardware architecture of each equation. It given in Fig. 7 the model of the architecture update velocities and positions stage.

RESULTS AND DISCUSSION

The performance of the PSO algorithm depends on many factors. These parameters are shown in Table 1 below. Values gives in this table are used to achieve this study and are applied to give experiments' results for performances demonstration.

For a given initial image converted to grayscale, it is necessary to start by giving a result of segmentation applied on Benchmarks images 250 x 250 pixels dimensions.

All results are given in the Fig. 8a.

To validate what preceded the architecture has been implemented on MATLAB using the same machine, the same FPGA target and the same features for the algorithm when using XSG. It is

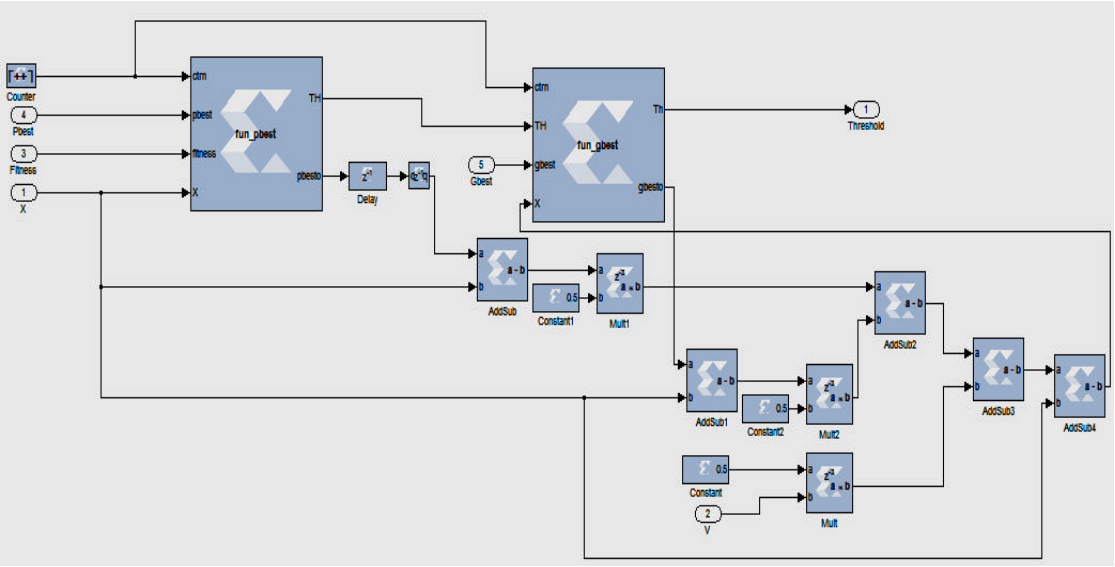

Fig. 7: Update block architecture implemented using XSG design tool

Table 1: Experimental parameters' values used in PSO algorithm

Parameters	Value
No. of particles (N)	80
No. of iterations (M)	100
Cognitive coefficient (c1)	0.5
Cognitive coefficient (c2)	0.5
Inertia weight (w)	0.5

obvious that the result given by MATLAB is better than that determined by XSG MATLAB since a 64-bit processor is used. Also, processors of used PC work with true real. Quantitatively, the result given by XSG is good; Fig. 8c, f, i, l show more details such as the mouth of Lena Fig. 8c and Mandrill Fig. 8i, samely for the eyes and nose Fig. 8c, i. The same case is gives for Hunter Fig. 8l, the hat appears with a few details and hands are clear. Comparing these results with the figures given by MATLAB Fig. 8b, e, h, k which are considered as rerferences, theyare not so far in terms of visual quality. That shows that this architecture is efficient and gives satisfactory results. To conclude, the figures after segmentation include several details, as they are rather clear and very similar relative to the figures gives after MATLAB segmentation.

To make further reassurance in terms of quality of segmentation results provided by the developed architecture, T1-weighted brain image was applied. The result is given in Fig. 9.

Figure 9 shows the T1 weighted MR image of brain composed of three different zones which are the Cerebrospinal Fluid (CSF), the Gray Matter (GM) and the White Matter (WM). Details in Fig. 9c are clearer since it retains as much information as possible to the CSF area. In addition, GM is clear same as the other method of segmentation based MATLAB and gives in Fig. 9b.

So, similarly, the segmentation result given by MATLAB is always the best. But it can also claims that the quality of image segmentation by XSG is acceptable. Indeed, the details in cerebrospinal area are clear and sharp. Always focusing on the same figure, the gray matter area shows details which prove the splendid quality of the segmentation.

Fig. 8(a-l): Comparison of experimental results of PSO segmentation given MATLAB and XSG tool, (a) Lenna, (d) Peppers, (g) Mandrill, (j) Hunter, (b, e, h, k) Experimental results of PSO segmentation using MATLAB, (c, f, i, l) Experimental results of PSO segmentation using XSG

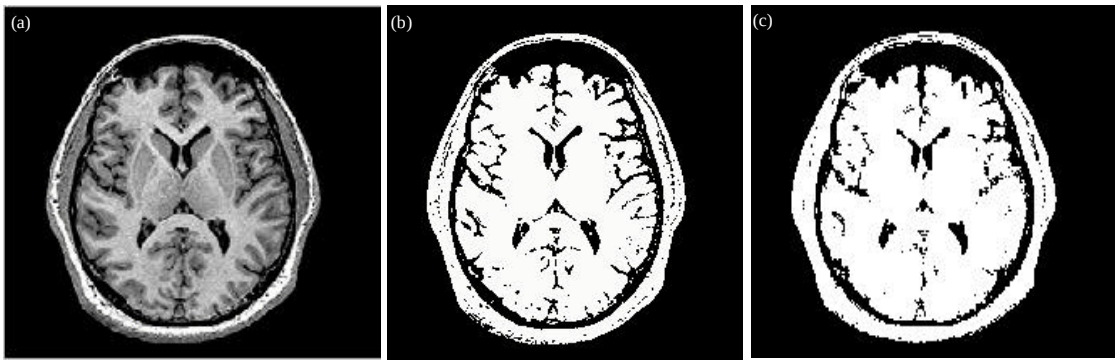

Fig. 9(a-c): Comparison of experimental results of PSO segmentation given by MATLAB and XSG tool, (a) Original image of T1 Brain image, (b) Experimental result of PSO segmentation using MATLAB and (c) Experimental result of PSO segmentation using XSG

Table 2: List of FPGA resources utilization synthesis given by the PSO architecture of segmentation

Logic slice utilization	Used	Available	Utilization(%)
No. of slice registers	2611	21.760	12
No. used as flip flops	2611		
No. of slice LUTs	3514	21.760	16
No. used as logic	982	21.760	4
No. used as memory	2246	8.320	27
No. of occupied slices	1002	5.440	18
No. of LUT flip flop pairs used	3618		
No. of bonded IOBs	155	360.0	43
No. of BlockRAM/FIFO	32	84.0	38
No. of using BlockRAM only	14		

The FPGA target of the images segmentation based PSO architecture operates with a 100 MHz clock. There are many parameters wich prove the quality of the developed architecture. But, only both parameters are with the most important influence and they are respectively the number of flip flops used in FPGA and the of memory ressources. In this case, 12% of logic resources and 38% of buffers are used. In Table 2, it gives the resources utilization of the hardware implemented architecture.

CONCLUSION

In this study, a new segmentation architecture based on PSO algorithm using the Xilinx System Generator tool was proposed. It is a digital image processing that offers an architecture based design for segmentation to be implemented on FPGA.

Even steps are designed in reference to mathimatecal equations such as fitness function and update velocity and position of particles. They supported MATLAB codes using customizable blocks and basic ressources of FPGA processor like adders, multipliers, the comparators, the multiplixeurs. XSG automtically generates necessary files for implementation in Xilinx FPGA platform with robust, speed and automatic area minimization. The design given in this work is implemented on Virtex V platform kit and it could be extended to real time medical image processing with proper user configuration.

The proposed architecture of segmentation was applied on benchmarks images and some medical images and results was compared to the same algorithm developed on MATLAB. Results given by the proposed method showed that the proposed sheme is efficient and real time excution with acceptable visual quality of images given after segmentation for both benchmarks and medical images.

REFERENCES

Bernd, J., 2000. Computer Vision and Applications: A Guide for Students and Practitioners. 1st Edn., Academic Press, New York, USA., ISBN: 9780123797773, pp: 5-30.

Couprie, C., L. Grady, L. Najman and H. Talbot, 2009. Power watersheds: A new image segmentation framework extending graph cuts, random walker and optimal spanning forest. Proceedings of the IEEE 12th International Conference on Computer Vision, September 29-Octber 2, 2009, Kyoto, Japan, pp: 731-738.

Delva, J., A. Chirila-Rus, B. Chan and S. Seng, 2008. Using system generator for systematic HDL design, verification and validation. Hite Paper: Xilinx System Generator, WP283 (v1.0) January 17, 2008. http://www.xilinx.com/support/documentation/white_papers/wp283.pdf.

Eberhart, R.C. and Y. Shi, 2000. Comparing inertia weights and constriction factors in particle swarm optimization. Proceedings of the Congress on Evolutionary Computation, July 16-19, 2000, IEEE Service Center, San Diego, California, pp: 84-88.

Eklundh, J.O., 1998. Vision in robotics: How a robot can segment ?gure from ground. Proceedings of the European Conference on Artificial Intelligence, August 23-28, 1998, Brighton, UK., pp: 689-693.

Forsyth, D.A. and J. Ponce, 2002. Computer Vision: A Modern Approach. Prentice Hall, London, UK., ISBN: 0130851981.

Hamdaoui, F., A. Ladgham, A. Sakly and A. Mtibaa, 2012. Real time implementation of medical images segmentation using Xilinx system generator. Int. Rev. Comput. Software, 7: 2861-2867.

Hamdaoui, F., A. Khlifa, A. Sakly and A. Mtibaa, 2013a. Real time implementation of medical images segmentation based PSO. Proceedings of the International Conference on Control, Decision and Information Technologies, April 11-13, 2013, Hammamet, Tunisia.

Hamdaoui, F., A. Ladgham, A. Sakly and A. Mtibaa, 2013b. A new images segmentation method based on modified particle swarm optimization algorithm. Int. J. Imaging Syst. Technol., 23: 265-271.

Han, Y. and P. Shi, 2006. An improved ant colony algorithm for fuzzy clustering in image segmentation. Neurocomputing, 70: 665-671.

Hechri, A., F. Hamdaoui, A. Ladgham and A. Mtibaa, 2011. Using fuzzy logic path tracking for an autonomous robot. Int. Rev. Autom. Control, 4: 115-123.

Holland, J., 1992. Adaptation in Natural and Artificial Systems. 2nd Edn., MIT Press, Cambridge, MA., Pages: 211.

Karaboga, D. and C. Ozturk, 2011. A novel clustering approach: Artificial Bee Colony (ABC) algorithm. Applied Soft Comput., 11: 652-657.

Kennedy, J. and R. Eberhart, 1995. Particle swarm optimization. Proceedings of the International Conference on Neural Networks, Volume 4, November 27-December 1, 1995, Perth, WA., USA., pp: 1942-1948.

Kennedy, J., 1997. The particle swarm: Social adaptation of knowledge. Proceedings of the International Conference on Evolutionary Computation, April 13-16, 1997, Indianapolis, IN., pp: 303-308.

Ladgham, A., F. Hamdaoui, A. Sakly and A. Mtibaa, 2012. Real time implementation of detection of bacteria in microscopic images using system generator. J. Biosensors Bioelectron., Vol. 3.

Lempitsky, V., P. Kohli, C. Rother and T. Sharp, 2009. Image segmentation with a bounding box prior. Proceedings of the IEEE 12th International Conference on Computer Vision, September 29-October 2, 2009, Kyoto, Japan.

Mailloux, J.G., 2008. Prototypage rapide de la commande vectorielle sur fpga a laide des outils simulink-system generator.[Rapid prototyping of vector control on FPGA using Simulink-System Generator]. Master's Thesis, University of Quebec, Quebec, Canada.

Maulik, U. and S. Bandyopadhyay, 2000. Genetic algorithm-based clustering technique. Pattern Recogn., 33: 1455-1465.

Reynolds, C., 1987. Flocks, herds and schools: A distributed behavioral model. Comput. Graph., 21: 25-34.

XSG, 1998. Xilinx system generator v2.1 basic tutorial. Printed in USA, http://bwrcs.eecs.berkeley.edu/Classes/cs152/handouts/Tutorials_book.pdf.

Zhang, H., Y. Wang, B. Wang and X. Wu, 2007. Evolutionary random sequence generators based on LFSR. Wuhan Univ. J. Nat. Sci., 12: 75-78.