FPGA Implementation of Alfa-Beta Associative Memories

Mario Aldape-Pérez, Cornelio Yáñez-Márquez, and
Amadeo José Argüelles-Cruz

Center for Computing Research, CIC
National Polytechnic Institute, IPN
Mexico City, Mexico
mario@aldape.org.mx; cyanez@cic.ipn.mx; jamadeo@cic.ipn.mx
WWW home page: http://www.aldape.org.mx

Abstract. Associative memories have a number of properties, including a rapid, compute efficient best-match and intrinsic noise tolerance, that make them ideal for many applications. However, a significant bottleneck to the use of associative memories in real-time systems is the amount of data that requires processing. Notwithstanding, Alfa-Beta Associative Memories have been widely used for color matching in industrial processes [1], text translation [2] and image retrieval applications [3]. The aim of this paper is to present the work that produced a dedicated hardware design, implemented on a field programmable gate array (FPGA) that applies the Alfa-Beta Associative Memories model for pattern recognition tasks. Along the experimental phase, performance of the proposed associative memory architecture is measured by learning large sequences of symbols and recalling them successfully. As a result, a simple but efficient embedded processing architecture that overcomes various challenges involved in pattern recognition tasks is implemented on a Xilinx Spartan3 FPGA.

Keywords: Associative Memories, FPGA, Pattern Recognition, Reconfigurable Logic.

1 Introduction

An associative memory M is a system that relates input patterns and output patterns as follows: $x \longrightarrow \boxed{M} \longrightarrow y$ with x and y, respectively, the input and output pattern vectors. Each input vector forms an association with its corresponding output vector. For each k integer and positive, the corresponding association will be denoted as: (x^k, y^k). Associative memory M is represented by a matrix whose ij-th component is m_{ij} [4]. Memory M is generated from an *a priori* finite set of known associations, called the fundamental set of associations. If μ is an index, the fundamental set is represented as: $\{(x^\mu, y^\mu) \mid \mu = 1, 2, ..., p\}$ with p as the cardinality of the set. The patterns that form the fundamental set are called fundamental patterns. If it holds that $x^\mu = y^\mu \; \forall \mu \in \{1, 2, ..., p\}$ M is

Table 1. Alfa and Beta Operators.

$\alpha : A \times A \longrightarrow B$

x	y	α(x,y)
0	0	01
0	1	00
1	0	10
1	1	01

$\beta : B \times A \longrightarrow A$

x	y	β(x,y)
00	0	0
00	1	0
01	0	0
01	1	1
10	0	1
10	1	1

auto-associative, otherwise it is heteroassociative; in this case, it is possible to establish that $\exists \mu \in \{1, 2, ..., p\}$ for which $x^\mu \neq y^\mu$. If we consider the fundamental set of patterns $\{(x^\mu, y^\mu) \mid \mu = 1, 2, ..., p\}$ where n and m are the dimensions of the input patterns and output patterns, respectively, it is said that $x^\mu \in A^n$, $A = \{0, 1\}$ and $y^\mu \in A^m$. Then the j-th component of an input pattern is $x_j^\mu \in A$. Analogously, the j-th component of an output pattern is represented as $y_j^\mu \in A$. A distorted version of a pattern x^k to be recuperated will be denoted as $\tilde{x}^k$. If when feeding an unknown input pattern x^ω with $\omega \in \{1, 2, ..., k, ..., p\}$ to an associative memory M, it happens that the output corresponds exactly to the associated pattern y^ω, it is said that recuperation is perfect.

2 Alfa-Beta Associative Memories

Alfa-Beta Associative Memories mathematical foundations are based on two binary operators: α and β. Alfa operator is used during the learning phase while Beta operator is used during the recalling phase. The mathematical properties within these operators, allow the $\alpha\beta$ associative memories to exhibit similar characteristics to the binary version of the morphological associative memories, in the sense of: learning capacity, type and amount of noise against which the memory is robust, and the sufficient conditions for perfect recall [5]. First, we define set $A = \{0, 1\}$ and set $B = \{00, 01, 10\}$, so α and β operators can be defined as in Table 1.

These two binary operators along with maximum ($\vee$) and minimum ($\wedge$) operators establish the mathematical tools around the Alfa-Beta model. The definitions of α and β exposed in Table 1, imply that: α is increasing by the left and decreasing by the right, β is increasing by the left and right, β is the left inverse of α. According to the type of operator that is used during the learning phase, two kinds of Alfa-Beta Associative Memories are obtained. If maximum operator ($\vee$) is used, Alfa-Beta Associative Memory of type MAX will be obtained, denoted as M; analogously, if minimum operator ($\wedge$) is used, Alfa-Beta Associative Memory of type min will be obtained, denoted as W [6]. In any case, the fundamental input and output patterns are represented as follows:

$$x^\mu = \begin{pmatrix} x_1^\mu \\ x_2^\mu \\ \vdots \\ x_n^\mu \end{pmatrix} \in A^n \qquad y^\mu = \begin{pmatrix} y_1^\mu \\ y_2^\mu \\ \vdots \\ y_m^\mu \end{pmatrix} \in A^m$$

In order to understand how the learning and recalling phases are carried out, some matrix operations definitions are required.

α max Operation: $P_{mxr} \nabla_\alpha Q_{rxn} = \left[f_{ij}^\alpha\right]_{mxn}$, where $f_{ij}^\alpha = \vee_{k=1}^r \alpha(p_{ik}, q_{kj})$
β max Operation: $P_{mxr} \nabla_\beta Q_{rxn} = \left[f_{ij}^\beta\right]_{mxn}$, where $f_{ij}^\beta = \vee_{k=1}^r \beta(p_{ik}, q_{kj})$
α min Operation: $P_{mxr} \Delta_\alpha Q_{rxn} = \left[f_{ij}^\alpha\right]_{mxn}$, where $f_{ij}^\alpha = \wedge_{k=1}^r \alpha(p_{ik}, q_{kj})$
β min Operation: $P_{mxr} \Delta_\beta Q_{rxn} = \left[f_{ij}^\beta\right]_{mxn}$, where $f_{ij}^\beta = \wedge_{k=1}^r \beta(p_{ik}, q_{kj})$

Whenever a column vector of dimension m is operated with a row vector of dimension n, both operations ∇_α and Δ_α, are represented by $\oplus$; consequently, the following expression is valid:

$$y \nabla_\alpha x^t = y \oplus x^t = y \Delta_\alpha x^t.$$

If we consider the fundamental set of patterns $\{(x^\mu, y^\mu) \mid \mu = 1, 2, ..., p\}$ then the ij-th entry of the matrix $y^\mu \oplus (x^\mu)^t$ is expressed as follows:

$$\left[y^\mu \oplus (x^\mu)^t\right]_{ij} = \alpha(y_i^\mu, x_j^\mu).$$

2.1 Learning Phase

Find the adequate operators and a way to generate a matrix M that will store the p associations of the fundamental set $\{(x^1, y^1), (x^2, y^2), ..., (x^p, y^p)\}$, where $x^\mu \in A^n$ and $y^\mu \in A^m$ $\forall \mu \in \{1, 2, ..., p\}$.

Step 1. For each fundamental pattern association $\{(x^\mu, y^\mu) \mid \mu = 1, 2, ..., p\}$, generate p matrices according to the following rule:

$$\left[y^\mu \oplus (x^\mu)^t\right]_{mxn}$$

Step 2. In order to obtain an Alfa-Beta Associative Memory of type MAX, apply the binary MAX operator ($\vee$) according to the following rule:

$$M = \vee_{\mu=1}^p \left[y^\mu \oplus (x^\mu)^t\right]$$

Step 3. In order to obtain an Alfa-Beta Associative Memory of type min, apply the binary min operator ($\wedge$) according to the following rule:

$$W = \wedge_{\mu=1}^p \left[y^\mu \oplus (x^\mu)^t\right]$$

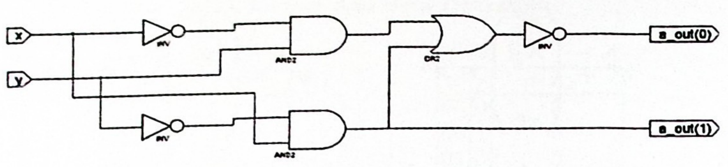

Fig. 1. Alfa Unit

Consequently, the ij-th entry of an Alfa-Beta Associative Memory of type MAX is given by the following expression:

$$\nu_{ij} = \vee_{\mu=1}^{p} \alpha(y_i^{\mu}, x_j^{\mu})$$

Analogously, the ij-th entry of an Alfa-Beta Associative Memory of type min is given by the following expression:

$$\psi_{ij} = \wedge_{\mu=1}^{p} \alpha(y_i^{\mu}, x_j^{\mu}).$$

2.2 Recalling Phase

Find the adequate operators and sufficient conditions to obtain the fundamental output pattern y^{μ}, when either the memory M or the memory W is operated with the fundamental input pattern x^{μ}.

Step 1. A pattern x^{ω}, with $\omega \in \{1, 2, ..., p\}$, is presented to the Alfa-Beta Associative Memory, so x^{ω} is recalled according to one of the following rules.

Alfa-Beta Associative Memory of type MAX:

$$M \Delta_{\beta} x^{\omega} = \wedge_{j=1}^{n} \beta(\nu_{ij}, x_j^{\omega}) = \wedge_{j=1}^{n} \left\{ \left[\vee_{\mu=1}^{p} \alpha(y_i^{\mu}, x_j^{\mu}) \right], x_j^{\omega} \right\}$$

Alfa-Beta Associative Memory of type min:

$$W \nabla_{\beta} x^{\omega} = \vee_{j=1}^{n} \beta(\psi_{ij}, x_j^{\omega}) = \vee_{j=1}^{n} \left\{ \left[\wedge_{\mu=1}^{p} \alpha(y_i^{\mu}, x_j^{\mu}) \right], x_j^{\omega} \right\}$$

Without dependence on the Alfa-Beta Associative Memory type used throughout the recalling phase, a column vector of dimension n will be obtained.

3 Implementation Details

As previously mentioned, the main goal of this paper is to derive an efficient implementation of the Alfa-Beta Associative Memories targeted towards FPGAs.

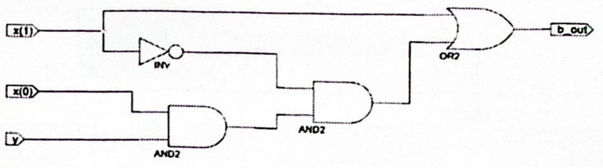

Fig. 2. Beta Unit

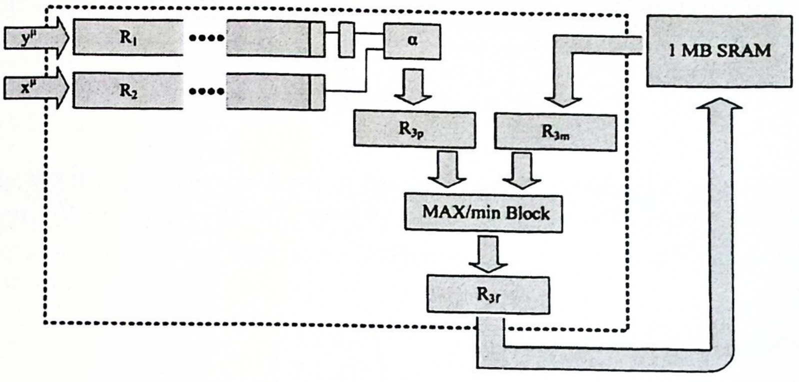

Fig. 3. Learning Phase Architecture

The Alfa operator implementation is shown in Figure 1, while the Beta operator implementation is shown in Figure 2.

The proposed architecture works with a 50 MHz master clock, which implies a 20ns period. As is it shown in Figure 3, the learning phase is implemented with 5 registers, 1 Alfa block, 1 MAX/min block and 2 external 10ns SRAM chips (mounted on the same board), that allow 1MB of data storage. There are two remarkable topics to be taken into consideration. The former concerns about the amount of logic resources that are needed to implement the two binary operators (Alfa and Beta). The latter results from the fact that most of the components that constitute the learning phase are combinatorial circuits. Hence, it is possible to read data from the external SRAM memory at the same time that a new bit is shifted to the Alfa block. Therefore, it is possible to write back the result of the MAX/min block to the external SRAM memory, during the same clock period.

As it is shown in Figure 4, the recalling phase is implemented with 4 registers, 1 Beta block, 1 min/MAX block and the same 2 external 10ns SRAM chips that were used to store the fundamental associations during the learning phase. The recalling phase is executed as follows. Firstly, R_{3m} receives one data word from

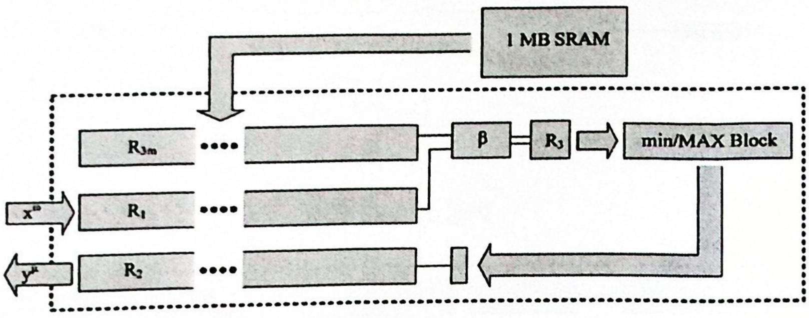

Fig. 4. Recalling Phase Architecture

the Alfa-Beta Associative Memory (stored in the 2 external 10ns SRAM chips). Then, R_1 receives the unknown input pattern. Finally, R_2 stores the recalled output pattern.

4 Numerical Results

Example 4.1. Let $p = 5$, $n = 4$, $m = 4$. Given the fundamental patterns $\{(x^\mu, y^\mu) \mid \mu = 1, 2, ..., p\}$, obtain an Alfa-Beta Associative Memory. The fundamental associations will be denoted as: $\{(x^1, y^1), (x^2, y^2), ..., (x^5, y^5)\}$.

$$x^1 = \begin{pmatrix}1\\1\\0\\1\end{pmatrix} \quad x^2 = \begin{pmatrix}0\\0\\0\\1\end{pmatrix} \quad x^3 = \begin{pmatrix}0\\1\\1\\1\end{pmatrix} \quad x^4 = \begin{pmatrix}0\\1\\0\\0\end{pmatrix} \quad x^5 = \begin{pmatrix}1\\0\\1\\1\end{pmatrix}$$

$$y^1 = \begin{pmatrix}1\\1\\0\\1\end{pmatrix} \quad y^2 = \begin{pmatrix}1\\0\\0\\1\end{pmatrix} \quad y^3 = \begin{pmatrix}1\\1\\1\\1\end{pmatrix} \quad y^4 = \begin{pmatrix}1\\1\\0\\0\end{pmatrix} \quad y^5 = \begin{pmatrix}1\\0\\1\\1\end{pmatrix}$$

Learning phase. Obtain the corresponding matrices $M_1, M_2, \ldots, M_5$, according to step 1, indicated in section 2.1.

$$y^1 \oplus (x^1)^t = \begin{pmatrix}1\\1\\0\\1\end{pmatrix} \oplus (1\,1\,0\,1) = \begin{pmatrix}1\,1\,2\,1\\1\,1\,2\,1\\0\,0\,1\,0\\1\,1\,2\,1\end{pmatrix}$$

$$y^2 \oplus (x^2)^t = \begin{pmatrix}1\\0\\0\\1\end{pmatrix} \oplus (0\,0\,0\,1) = \begin{pmatrix}2\,2\,2\,1\\1\,1\,1\,0\\1\,1\,1\,0\\2\,2\,2\,1\end{pmatrix}$$

$$\vdots$$

$$y^5 \oplus (x^5)^t = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} \oplus (1\,0\,1\,1) = \begin{pmatrix} 1\,2\,1\,1 \\ 0\,1\,0\,0 \\ 1\,2\,1\,1 \\ 1\,2\,1\,1 \end{pmatrix}$$

According to step 2, an Alfa-Beta Associative Memory of type MAX denoted by M, is obtained. Analogously, according to step 3, an Alfa-Beta Associative Memory of type min denoted by W, is obtained.

$$M = \begin{pmatrix} 2\,2\,2\,2 \\ 2\,1\,2\,2 \\ 2\,2\,1\,1 \\ 2\,2\,2\,1 \end{pmatrix} \quad ; \quad W = \begin{pmatrix} 1\,1\,1\,1 \\ 0\,1\,0\,0 \\ 0\,0\,1\,0 \\ 1\,0\,1\,1 \end{pmatrix}$$

Recalling phase. Obtain the corresponding output patterns, by performing the operations $M \,\Delta_\beta\, x^\mu$, $\forall \mu \in \{1, 2, ..., p\}$ as stated in section 2.2. Due to paper space limitations, only the Alfa-Beta MAX type recalling phase results are shown.

$$M \,\Delta_\beta\, x^1 = \begin{pmatrix} 2\,2\,2\,2 \\ 2\,1\,2\,2 \\ 2\,2\,1\,1 \\ 2\,2\,2\,1 \end{pmatrix} \Delta_\beta \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} = y^1$$

$$M \,\Delta_\beta\, x^2 = \begin{pmatrix} 2\,2\,2\,2 \\ 2\,1\,2\,2 \\ 2\,2\,1\,1 \\ 2\,2\,2\,1 \end{pmatrix} \Delta_\beta \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} = y^2$$

$$M \,\Delta_\beta\, x^3 = \begin{pmatrix} 2\,2\,2\,2 \\ 2\,1\,2\,2 \\ 2\,2\,1\,1 \\ 2\,2\,2\,1 \end{pmatrix} \Delta_\beta \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = y^3$$

$$M \,\Delta_\beta\, x^4 = \begin{pmatrix} 2\,2\,2\,2 \\ 2\,1\,2\,2 \\ 2\,2\,1\,1 \\ 2\,2\,2\,1 \end{pmatrix} \Delta_\beta \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} = y^4$$

$$M \,\Delta_\beta\, x^5 = \begin{pmatrix} 2\,2\,2\,2 \\ 2\,1\,2\,2 \\ 2\,2\,1\,1 \\ 2\,2\,2\,1 \end{pmatrix} \Delta_\beta \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} = y^5$$

The reader can easily verify that the Alfa-Beta *min* type recalling phase also recalls the whole fundamental set of patterns perfectly.

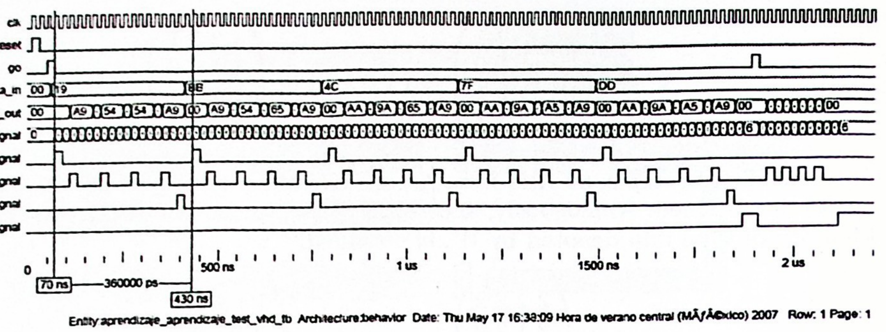

Fig. 5. Learning Phase

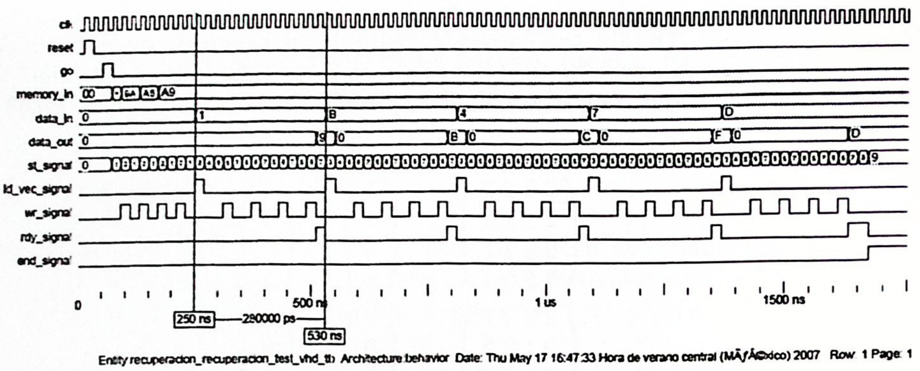

Fig. 6. Recalling Phase

5 Experimental Results

The experimental phase was carried out in two stages. In the first one, the same fundamental set of patterns that was presented in section 4, was downloaded to the proposed architecture. The performance results are shown in Figure 5 and Figure 6. The learning phase is executed in 2.5 μs and the recalling phase is executed in 2 μs. As expected, the entire fundamental set of patterns was perfectly recalled.

In order to estimate how the Alfa-Beta Associative Memory model performs with high dimensional data, 20 binary images (Figure 7 and Figure 8) were

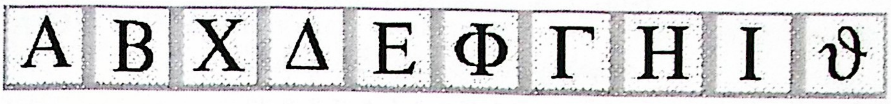

Fig. 7. Fundamental Input Patterns

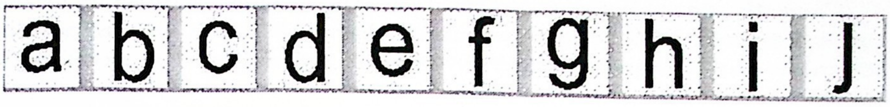

Fig. 8. Fundamental Output Patterns

used as fundamental patterns. Each one of these images is 40 by 40 pixels, which produces a 1600 bits pattern; accordingly, each pattern association results in a 640 Kbytes matrix. The experimental phase was carried out as follows: after the register initialization process was concluded, the first association was learned and recalled. Subsequently, the first and second associations were learned and recalled; after that, the same procedure continued in a consecutive manner until the fundamental set of patterns was completely learned and recalled. The above mentioned procedure was executed 100 times, each time changing the fundamental associations randomly. The averaged recalling results are shown in Table 2. A relevant thing to mention about the recalling criterion that was used along the experimental phase is that, in this case, perfect recall means that all of the 1600 bits were exactly recovered. Particularly, outstanding results were achieved by using the Alfa-Beta min type recall (the whole fundamental set of patterns was perfectly recalled).

6 Conclusions and Ongoing Research

In this paper, we introduced a simple but efficient implementation of the Alfa-Beta Associative Memories targeted towards FPGAs that overcomes a serious challenge in pattern recognition tasks (bottle-neck problems due to high dimensional data). A relevant thing to mention is that after a fundamental pattern is downloaded to the proposed architecture, each bit is learned in 90 ns, which fulfills one of the main purposes of this paper. Moreover, if the learning rate is known, it is possible to estimate the learning phase duration even with high dimensional fundamental patterns. Usually, this situation takes place when the fundamental patterns are RGB images. It is worth to mention that the proposed architecture can be easily adapted to work as an Alfa-Beta bi-directional associative memory.

Table 2. Fundamental set recalling results.

	1	2	3	4	5	6	7	8	9	10
$\alpha\beta$ *MAX*	1	2	3	4	4	4	4	3	1	1
$\alpha\beta$ *min*	1	2	3	4	5	6	7	8	9	10

Currently, we are investigating how to use the proposed architecture for feature selection in RGB images and mixed noise variants. We are also working towards a parallel implementation of the learning phase, based on recent mathematical results.

Acknowledgments The authors of the present paper would like to thank the following institutions for their support: National Polytechnic Institute, Mexico (CIC, CGPI, PIFI, COFAA) , CONACyT and SNI.

References

[1] Yáñez-Márquez, C., Felipe-Riverón, E. M., López-Yáñez, I., & Flores-Carapia, R. (2006). A Novel Approach to Automatic Color Matching. Lecture Notes in Computer Science (LNCS), 4225, 529-538.
[2] Acevedo-Mosqueda, M. E., Yáñez-Márquez, C., & López-Yáñez, I. (2006). Alpha-Beta Bidirectional Associative Memories Based Translator. International Journal of Computer Science and Network Security (IJCSNS), 6, 190-194.
[3] Yáñez-Márquez, C., Sánchez-Fernández, L. P., & López-Yáñez, I. (2006). Alpha-Beta Associative Memories for Gray Level Patterns. Lecture Notes in Computer Science (LNCS), 3971, 818-823.
[4] Kohonen, T. (1972). Correlation Matrix Memories. IEEE Transactions on Computers, 21, 353-359.
[5] Acevedo-Mosqueda, M. E., Yáñez-Márquez, C., & López-Yáñez, I. (2006). A New Model of BAM: Alpha-Beta Bidirectional Associative Memories. Lecture Notes in Computer Science (LNCS), 4263, 286-295.
[6] Acevedo-Mosqueda, M. E., Yáñez-Márquez, C., & López-Yáñez, I. (2007). Alpha–Beta bidirectional associative memories: theory and applications. Neural Processing Letters, 26, 1-40.