DEVELOPMENT OF AN ARABIC HANDWRITING LEARNING EDUCATIONAL SYSTEM

Hala Bezine and Adel M. Alimi

REGIM: REsearch Group on Intelligent Machines, University of Sfax,
National School of Engineers (ENIS). BP 1173, Sfax, 3038, Tunisia
Phone: +2016-74-274088, Fax: +2016-74-275-595
hala.bezine@ieee.org, adel.alimi@ieee.org

ABSTRACT

In today many students produce a wrong and illegible handwriting and teachers need a lot of time to check the handwriting errors. The traditional approach for handwriting teaching needs a long hours of handwriting practice. Unfortunately, this is not advantageous in many cases. In this paper, we develop an automated educational system for Arabic handwriting learning and detection of errors such as: the stroke production errors, stroke sequence errors, stroke relationship errors and stroke interline errors. Then our system can check the handwriting mistakes and immediately provide a feedback to the students to correct themselves.

In this work, we used an attributed relational graph to locate the handwriting mistakes. The experimental results show that our proposed system can treat multiple cases of errors with satisfied accuracy.

KEYWORDS

Arabic handwriting learning, handwriting mistakes, handwriting teaching, attibuted relational graph, intelligent systems

1. INTRODUCTION

In the traditional handwriting teaching system, the teacher must write an Arabic character on the blackboard and the students should rewrite the handwritten character on their copy books. After that, the teacher tries to check the handwriting errors in the student's notebooks and provides a feedback in the next time, because it's impossible for a teacher to verify and check every student's handwriting in the limited time of the lesson. This system can be successfully acquired only though practice regularly and for long periods. So these techniques have many drawbacks which motivate the researchers to work in this field. In this context, Z. Hu et al. [12] define three drawbacks of the traditional education, such as time -consumption, faultiness, Teacher-oriented). In fact, the learning process becomes much more effective, if the handwritten character is checked just after the students have finished their handwriting. On the other hand, the students cannot learn without the teacher supervision and they can correct the committed errors and repeat the same exercise several times they hope to speed up the learning process. Arabic alphabet consists of 29 characters and eight diacritics [8, 17]. It is read and written from the right to left. Words are written with respect to horizontal lines. Moreover, each Arabic character has two up to four different forms depending on its position within a word: initial, medial, final or isolated as depicted in figure 1. Consequently, the number of patterns increases from 29 to about 60 patterns as shown in table 1.

DOI : 10.5121/ijsea.2013.4203

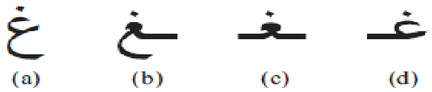


Figure 1: Different forms of the "GHYN" character.
(a) Isolated form, (b) Final form, (c) Medial form, (d) Initial form.

In addition, sixteen of the Arabic characters have a single dot, double, triple dots, or zigzag, called Hamza in Arabic [5].These diacritics are used to distinguish between characters having identical main parts. In this context, there are many groups of characters that have the same body, but they are distinguished by the number of dots (تب), the position of dots (خ,ج), whether they have dot or zigzag (ﻨ, ﺌ) [9]. An Arabic script is written by connecting the characters together. However, these six letters ر,ذ,د,ا,وز) cannot be joined to any other letter coming after one of them. So, an Arabic character might be connected to one side (either right or left), to both sides, or to neither one.

As agreed by the most researchers related to the document analysis and recognition field, the efficiency of a classifier and a handwriting generation model depends mainly on the features extraction step [4, 8, 22]. The same concept is depicted for the handwriting education and learning problem. In fact the writing style, speed, and the writing size have a great influence on the student's handwriting shape. The students spend long hours of handwriting practice and the teachers have to check the handwriting errors to provide the feedback in a next time.

This paper is about an automated tool for Arabic handwriting learning. This educational system is developed, using tablet PC and the attributed relational graph matching approach. Firstly, some animation is displayed for the handwriting template, and then the student should follow it to rewrite the handwriting input characters. Moreover, the system requires a detailed matching between the input and the template characters to check the handwriting errors. Finally, an immediate feedback is provided to the student about the location of the handwriting errors and how to correct them.

The organization of this paper is as follows: In section2, an overview on the educational learning systems is presented. Then, in section 3, we describe our proposed system accompanied by the handwriting errors detection and the correction procedure. Finally, section 4, is allocated to the conclusion and future works.

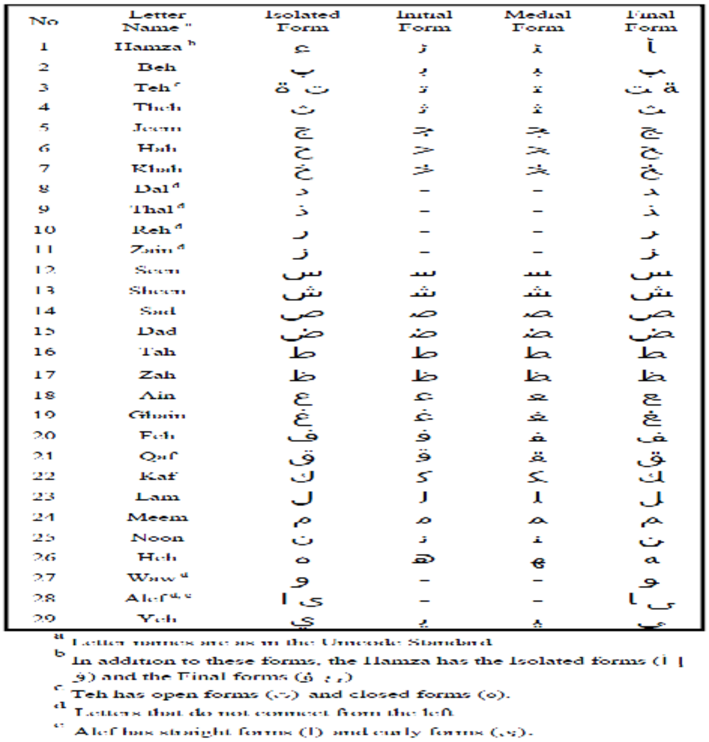
2. STATE OF THE ART OF THE HANDWRITING LEARNING EDUCATIONAL SYSTEMS

In the last few years, several research efforts have been focused on the developing of the learning handwriting systems for different languages such as: Chinese, Latin and Arabic. These products can be organized on three categories: read only systems, guided ones and systems with automatic errors detection.

2.1. Read only systems

This kind of systems is not interactive; it is read-only because it cannot provide the practice of writing. It displays to the users how a character should be written with pictures and animations, as well as the pronunciation of the character. But the practice of writing will be done offline (paper, book, etc...) and the task of errors correction is allocated to the users, they should compare their handwriting on the paper with the pattern displayed by the system. In this context Muroya et al. 2001, have developed an educational software tool for teaching handwriting. This tool displays some animation to help the students to rewrite the handwriting pattern [18].

Table 1: The Arabic characters and their forms at different positions in an Arabic word.



No	Letter Name [a]	Isolated Form	Initial Form	Medial Form	Final Form
1	Hamza [b]	ء	ئـ	ـئـ	أ
2	Beh	ب	بـ	ـبـ	ـب
3	Teh [c]	ة ت	تـ	ـتـ	ـت ـة
4	Theh	ث	ثـ	ـثـ	ـث
5	Jeem	ج	جـ	ـجـ	ـج
6	Hah	ح	حـ	ـحـ	ـح
7	Khah	خ	خـ	ـخـ	ـخ
8	Dal [d]	د	–	–	ـد
9	Thal [d]	ذ	–	–	ـذ
10	Reh [d]	ر	–	–	ـر
11	Zain [d]	ز	–	–	ـز
12	Seen	س	سـ	ـسـ	ـس
13	Sheen	ش	شـ	ـشـ	ـش
14	Sad	ص	صـ	ـصـ	ـص
15	Dad	ض	ضـ	ـضـ	ـض
16	Tah	ط	طـ	ـطـ	ـط
17	Zah	ظ	ظـ	ـظـ	ـظ
18	Ain	ع	عـ	ـعـ	ـع
19	Ghain	غ	غـ	ـغـ	ـغ
20	Feh	ف	فـ	ـفـ	ـف
21	Qaf	ق	قـ	ـقـ	ـق
22	Kaf	ك	كـ	ـكـ	ـك
23	Lam	ل	لـ	ـلـ	ـل
24	Meem	م	مـ	ـمـ	ـم
25	Noon	ن	نـ	ـنـ	ـن
26	Heh	ه	هـ	ـهـ	ـه
27	Waw [d]	و	–	–	ـو
28	Alef [d, e]	ى ا	–	–	ـى ـا
29	Yeh	ي	يـ	ـيـ	ـي

[a] Letter names are as in the Unicode Standard

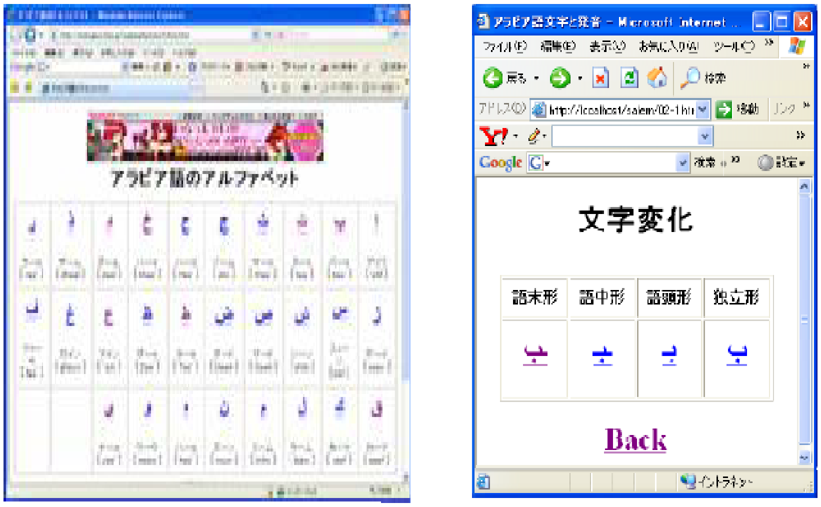
[b] In addition to these forms, the Hamza has the Isolated forms (أ إ ؤ) and the Final forms (ؤ ئ ء)

[c] Teh has open forms (ـت) and closed forms (ة).

[d] Letters that do not connect from the left.

[e] Alef has straight forms (ا) and curly forms (ى).

However, this tool is not interactive because the correction errors are made by the students to compare the handwriting on paper with the pattern displayed on the computer screen. On the other hand, Al-Neaimi S. et al. 2009 have developed a support system for Arabic language learning. This tool displays some animation and provides the pronunciation of Arabic characters [7]. Secondly, it contains efficient studying methods of learning, grammar, reading and writing. This tool is not interactive, because it is read-only system and users cannot practice handwriting through it. It displays a table of the Arabic Alphabets, in which each character is designated by text, animation and pronunciation. By clicking on one of these alphabets an explanation page will be displayed with the different possible forms of this character. Figure 2 shows an example of the proposed system interfaces.



(a): The Arabic alphabet table

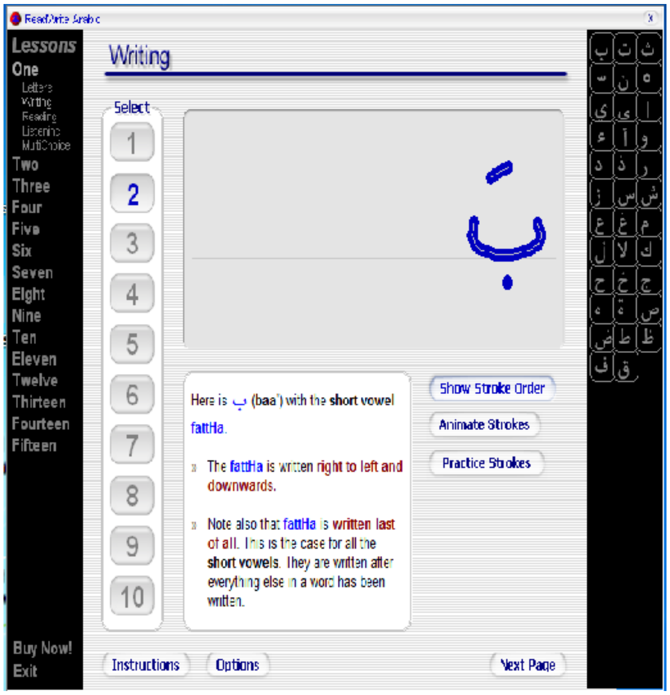
(b): The different forms of the letter « Bah »

Figure 2: An Example of interface of the Al Neaimi et al. tool [7].

2.2. Guided systems

These systems allow to the users to practice writing in a guided way and on-line. Similar to the traditional writing learning method, this kind of learning tools used the dotted pictures to make easier the writing practice, which is frequently used in the first lessons of the writing teaching in the primary schools. On the one hand, this tool displays a pattern of character with the pronunciation and animation and the correct process for the rewriting of this latter. The users have to rewrite the displayed character in order to acquire the basic skills of writing. For the Arabic language, an educational tool have been developed “ReadWrite Arabic” for the learning of reading/writing assistance (Figure 3).

The proposed system consists of fifteen lessons presenting the Arabic letters with some examples of use. These latter present how each character should to be written with different forms. Despite that each lesson is divided into five exercises namely (letters, writing, reading, listening, and multi-choice), the learning process becomes more and more difficult for young students because if there is just only one stroke which is drawn outside the defined zone by the transparent image, the system hangs, and the user should tries the writing process again [27].



Here is ب (baa') with the short vowel fattHa.

- The fattHa is written right to left and downwards.
- Note also that fattHa is written last of all. This is the case for all the short vowels. They are written after everything else in a word has been written.

Figure 3: Graphical interface of the Arabic writing learning [27].

2.3. Systems with automatic detection of errors

This kind of systems gives access to users to practice free writing mode, and provide immediately a feedback to indicate if there are any errors in the writing. The process can be performed as follows: Firstly, the system displays an animation of a pattern of character and the user should follows this latter. Then, a mapping between the displayed model and the input must takes place in order to detect the errors. Finally, a feedback is provided to the user indicating the committed mistakes and how to correct them.

For the Arabic language, Sherif A. et al, 2009 [3] proposed a teaching writing tool using the Tablet PC and handwriting recognition methodologies. The purpose of this tool is to help young children's to learning writing with correct movements to produce a good quality of writing in short periods of time. Firstly, the proposed system proceeds to recognize the entered writing by young children's and detects the different strokes comprised in the handwriting sample. Finally, it produces a feedback to help the young students to improve their writing.

The proposed system is composed of two modules: the guided writing mode and the free writing one. The architecture of the system is detailed in figure 4.

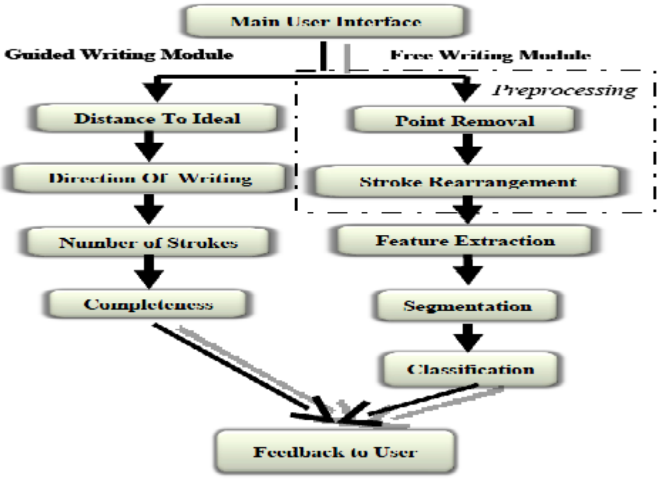
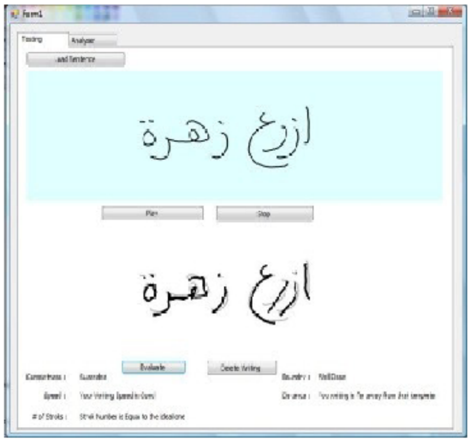


Figure 4: The architecture of the Arabic learning writing system with automatic detection errors.

The guided writing mode is one of the first levels of education designed for the children's who are in the early stages of learning. This tool displays an animation representing the writing process of a pattern of the letters (words, phrases) as well as a transparent image comprising this handwriting template. Then, the user is invited to follow this image to replicate the pattern of script. In addition, the authors define the control points over the transparent image by which the users should verify the correctness of their handwriting by comparing it with the script of reference registered on the transparent image.

The figure below (figure 5) shows an example of the guided handwriting mode. By this way, the user acquired the basic skills and he has the possibility to move to the second level of handwriting learning by using the free writing mode.

For this level, users can practice writing on a blank panel that contains a single line similar to the writing manuals used in the school. Indeed, the verification handwriting process comprises the classification stage which is not so effective for this case of study, because a detailed mapping between the plotted script in input and a stored template character is necessary.



Speed : Your Writing Speed is Good

# of Stroks : Stroke Number is Equal to the idealone

Boundary : WellDone

Distance : You writing is far away from that template

Figure 5: An example of the guided writing mode of Sherif Abdou et al, 2010 [2].

For the Latin language, we note the Evodia society founded by a team of researchers from the IRISA University of Rennes, is interested to the development of handwritten documents analysis software (Ahmed et al. 2010)[3]. This range of software named “Toutaki: I learn to write” provides an analysis of handwriting shape, the order of the traced form and the line spacing. Indeed, it is an interactive tool that allows a fun learning environment and a feedback about the detected errors but the location of the handwriting errors and how to correct them is not supported by this latter.

For the Chinese language, we retrieve multiple handwriting education systems have been suggested providing a feedback which indicate the handwriting errors, such that of K. T. Tang et al. (Tang et al. 2006) [24]. These authors have been proposed an approach for checking the stroke sequence errors. Others researchers have been focused on the stroke production errors such that; Tonouchi 1997(Tonouchi 1997) and Z. Hu 2007 (Hu et al. 2007) [14, 26]. The work proposed in (Hu et al. 2007) tries multiple types of errors e.g. the stroke production errors, stroke sequence errors and stroke relationship errors [14].

In this context, Z. Hu et al 2009, proposed a distant teaching application for Chinese handwriting learning. This latter allows an automatic detection of errors such us defined previously. After that, a feedback is returned and how to correct the committed mistakes [12]. In table 2, we summarized the proposed handwriting learning systems in the different languages with their different characteristics.

Table 2: A recapitulatif of the handwriting learning tools.

Handwriting Learning Tool	Language	Category	Patterns	Shape analysis	Strokes order analysis	Direction analysis	Interline analysis
Zhi-Hui Hu et, al. (2009)	Chinese	Automatic detection of errors	Letters	Yes	Yes	No	No
Salem Al-Neaimi et al. (2009)	Arabic	Read only system	Sentences	No	No	No	No
Sherif Abdou et al. (2009)	Arabic	Automatic detection of errors	Sentences	Yes	Yes	No	No
Toutaki (2010)	Latin	Automatic detection of errors	Words	Yes	Yes	Yes	Yes
ReadWrite Arabic (2010)	Arabic	Guided system	Words	Yes	Yes	Yes	Yes

As depicted in table 2, different types of patterns are used for these learning systems such as: characters, words, and sentences.

Indeed, multiple Analysis have been done ranging from the shape analysis to the interline one.

3. THE PROPOSED HANDWRITING LEARNING SYSTEM

Face to the development of information technology such as the appearance of Tablet/PC and PDA (Personal Digital Assistant), some researchers are moving towards the development of interactive tools in order to facilitate the learning process and help students to learn their mother tongue or foreign ones [2, 16]. Most of these tools are limited to the learning of the alphabet of only one language. In addition to that and due to the complexity of the Arabic language, there are no systems which are dealing with the Arabic language with diacritics. In this context, we are focused on the development of a multilingual tool for the learning of handwriting with automatic errors detection. The general architecture of this system is presented in figure 6.

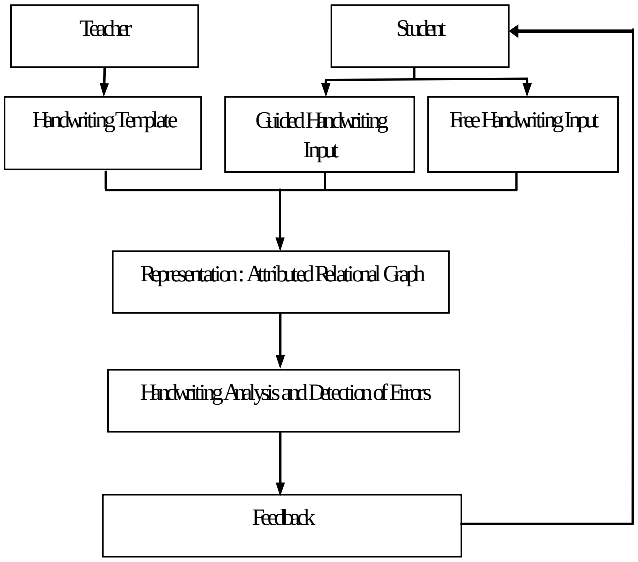

Teacher
Student
Handwriting Template
Guided Handwriting Input
Free Handwriting Input
Representation : Attributed Relational Graph
Handwriting Analysis and Detection of Errors
Feedback

Figure 6: The architecture of the proposed Arabic handwriting learning tool.

By using a tablet PC as a means of input and a transparent image, the user has the choice to practice the guided handwriting mode or the free one and such writing will be done on a blank area. Then the student has to select a template model in order to learn it. Then, by matching the handwriting template and the handwriting input, the performance of the entered path by the user will be carried out. Finally, an immediate feedback is provided to the student about the location of the errors, their type and how to correct them. The different steps of the previous architecture will be detailed in the following sections.

3.1. Approach description and the graph form representation of Arabic characters

In the literature, the graph technique is used in various fields of research, namely the processing and recognition of images, and classification and mapping of patterns [23]. For our proposed system, the Attributed Relational Graph (ARG) is used to model a handwriting trace. It should be noted that a trace represents one or a set of strokes which are organized according to a predefined structure (ARG) to form a word or a character [11]. This structure is defined by spatial relationships between the lines composing the handwriting shape. We have applied the graph matching algorithm to find the difference between the handwriting template model and the input handwriting pattern traced by the student. In the handwriting analysis step and in order to locate the stroke production errors (figure 7), we have applied the error tolerant graph matching to check the dissimilarity between both handwritten traces through the nodes and edges.

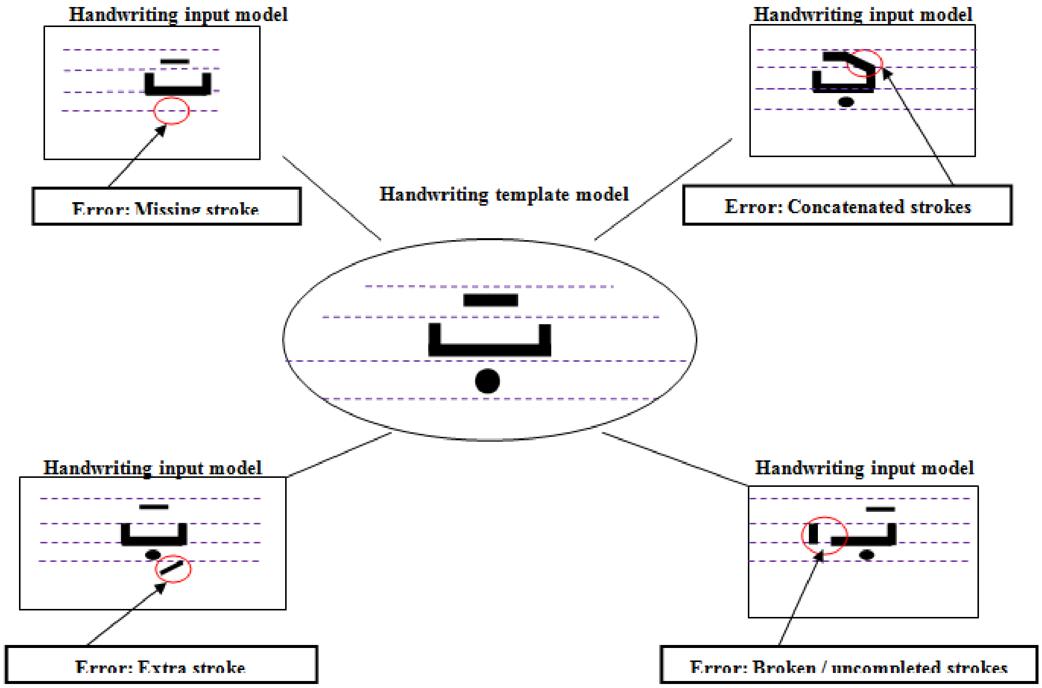


Figure 7: Examples of stroke production errors.

Indeed, the nodes in the graph are used to represent the different strokes in an appropriate character. The edges denote the relationship between two consecutive strokes. Let *V* the set of nodes and *E* the set of the edges. Then, the graph can be representing as $G = (V, E, \alpha, \beta)$.where $\alpha: V \rightarrow L_V$ is the node labeling function, and $\beta: E \rightarrow L_E$ is the edge labeling function. L_V and L_E are the sets of nodes and edges labels respectively.

Each node in the ARG is used to represent the strokes of the Arabic character. This latter stores the *x* and *y* coordinates of an appropriate stroke, i.e., the attributed of the node *a* is $a= (xi, yi)$; where i = 1, 2, 3,…, n. And *n* is the number of points which belong to the stroke.

The node labeling function $\alpha: V \rightarrow L_V$ returns (θ, γ), where θ is the *n* data points for each stroke and γ is the relationship between each stroke and interlines. Each edge stores the relationship of both nodes which are connected by the means of this edge. The edge labeling function $\beta: E \rightarrow L_E$ returns (μ, λ), where μ is the interval relationship along the x-axis and λ is the interval relationship along the y-axis [13, 20, 23]. The set of edges *E* represents the spatial relationships between the strokes of the handwriting trace. The spatial relationships are defined according to these proposed by Allen 1983, [6] as depicted in figure 8.

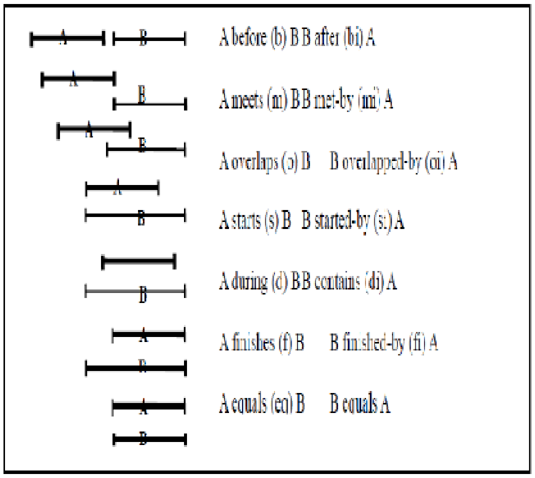

A
B
A before (b) B B after (bi) A
A meets (m) B B met-by (mi) A
A overlaps (o) B B overlapped-by (oi) A
A starts (s) B B started-by (si) A
A during (d) B B contains (di) A
A finishes (f) B B finished-by (fi) A
A equals (eq) B B equals A

Figure 8: The Allen's spatial intervals relationships. (Allen, 1983).

The set of nodes V is used to describe the strokes and the position of each stroke according to the baseline (writing line) as defined in table 3.

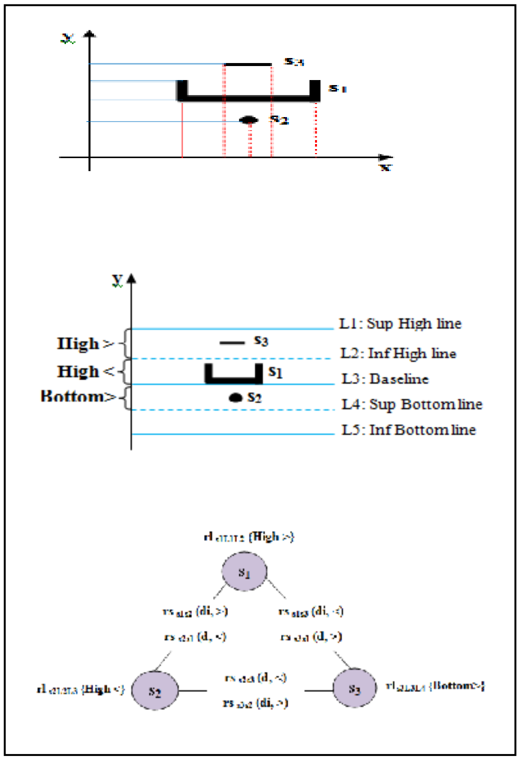
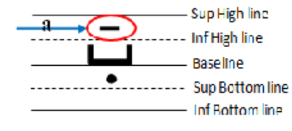
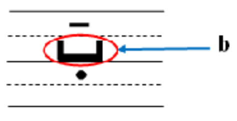
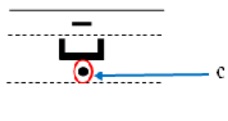
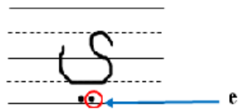
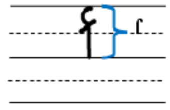
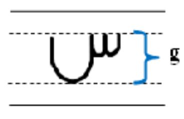
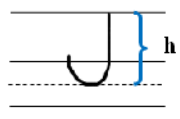


Figure 9: The modeling of the Arabic character ("Bah") by graph representation: (a) spatial relationship representation between different strokes. (b) Spatial relationship representation between each stroke and interlines. (c) The corresponding ARG.

As an example, we have the character "Bah"; the different relationships between strokes are shown in Figure 9(a). It's composed of three strokes designated respectively S_1, S_2 and S_3.The relationship between each stroke and the interlines is presented in figure 9(b). The Attributed Relational Graph (ARG) representation of this latter character is depicted in figure 9(c). The strokes S_1, S_2 and S_3 in the character "Bah" are equivalent to the nodes S_1, S_2 and S_3 in the ARG.

Table 3: Different intervals relationships between the strokes and interlines for few examples of Arabic characters.

Stroke	Relation	Symbol	Example
a	[Sup High line , Inf High line]	high >	 a Sup High line Inf High line Baseline Sup Bottom line Inf Bottom line
b	[Inf high line , baseline]	High <	
c	[baseline , Sup bottom line]	Bottom >	
e	[Sup bottom line , Inf bottom line]	Bottom <	
f	[Sup high line , baseline]	{High>, High <}	
g	[Inf high line, Sup bottom line]	{high <, bottom>}	
h	[Sup high line , Sup bottom line]	{high>, high <, bottom>}	

The variable rs_{sisj} is the relationship between strokes Si and Sj, and i, j =1, 2, 3, S_i, S_j (S_1, S_2, S_3), and $S_i \neq S_j$. The relationship rs_{s1s3} is represented by (di, <), rs_{s1s2} is denoted by (di, >), and rs_{s2s3} is noted by (d, >). Note that the rs_{s2s1} is formed by applying the inverse of each component of the relationship used to represent rs_{s2s1} and noted by (d, <). The variable rls_{iLxLy} is the relationship between the stroke Si and the interlines L_x and L_y. Similarly to S_i, L_x and L_y {L_1, L_2, L_3, L_4, L_5} (L_1: High line max, L_2: High line min, L_3: Baseline, L_4: Bottom line max and L_5: Bottom line min), and $L_x \neq L_y$.

The relationships rl_{S1L1L2} is noted by {High>}, rl_{S2L2L3} is designed by {High<}, and rl_{S3L3L4} is denoted by {Bottom>}.

3.2. Handwriting analysis

The analysis is the main phase in our handwriting learning system. It consists of the correspondence between the reference model and the pattern inputted by the user. Subsequently, the errors should be detected and classified according to their types. This process consists of three steps: 1) Graph matching space relations between the strokes, 2) errors detection, and 3) Feedback.

3.2.1. Graph matching and space relations between strokes

Firstly, the template and the input handwriting patterns are both represented by an ARG, in such a way the graph $g_1 = (V_1, E_1, \alpha_1, \beta_1)$ represents the input pattern and the graph $g_2 = (V_2, E_2, \alpha_2, \beta_2)$ represents the template one. Secondly, we have to check the dissimilarity between both ARGs (g_1, g_2), by the means of the error-tolerant graph matching and the function f: $\widehat{V_1} \rightarrow \widehat{V_2}$, $\hat{E}_1 \rightarrow \hat{E}_2$, where $\widehat{V_1} \subseteq V_1$, $\widehat{V_2} \subseteq V_2$, $\hat{E}_1 \subseteq E_1$ and $\hat{E}_2 \subseteq E_2$ [17].

In addition, we have applied a transformation, denoted by function f, from the input graph to the template graph. This function consists of many edit operations available for both nodes and edges. In this context, H. C. Lam and al. 2001 have defined five edit operations nodes and three edit operations edges as follows [16]:

- Edit operations nodes: 1) node substitution implying that the input stroke is correct, 2) node merging implying that the input strokes are broken ones, 3) node splitting implying that the input stroke is a concatenated stroke, 4) node deletion implying that the input stroke is an extra one, 5) node insertion reflects a missing stroke.
- Edit operations Edges: 1) edge substitution implying that both nodes sharing the edge are correct, 2) edge deletion implying that one of the nodes sharing this edge is an extra or broken stroke, 3) edge insertion implying that one of the nodes sharing the edge is a missing or concatenated stroke.

To compute the spatial relationship distance between the set of edges R_t in the template input and the corresponding one R_s in the handwriting input model, we need to determine the cost of the edge substitution such that defined below [19, 21]

Edge substitution cost: is the matching cost between an edge in the input character and another one in the template character. We use *Rt* to denote the set of edges in the template pattern and Rs as the set of edges in the input one. The *ith* template edge Rt_i can be designed by $(\mu t_i, \lambda t_i)$ and Rs_j; the *jth* input edge can be denoted by $(\mu s_j, \lambda s_j)$. The dissimilarity between $(\mu t_i, \lambda t_i)$ and $(\mu s_j, \lambda s_j)$ is defined by $D(Rt_i, Rs_j)$ which is inspired from the interval neighborhood graph concept [6, 10] and depicted in figure 10.

Two intervals relationships are neighbors, if they can be transformed into another one by continuous deformation such as shortening, lengthening, and moving [10].

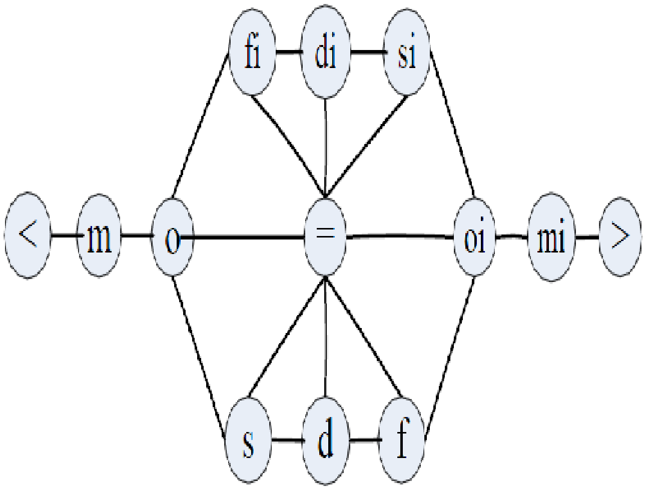


Figure 10: The neighborhood graph structure.

The distance between two intervals relationships μt_i and μs_j or the distance between λt_i and λs_j is defined as the topological distance between both relationships, i.e., the length of the shortest path from μt_i to μs_j or from λt_i to λs_j in the neighborhood graph interval. The final spatial relationship distance D (Rt_i, Rs_j) is defined as shown in (equation 1).

$$D(Rt_i, Rs_j) = \sqrt{D(\mu t_i, \mu s_j)^2 + D(\lambda t_i, \lambda s_j)^2} \qquad (1)$$

Edge deletion: It is used in conjunction with the node deletion operation. If some extra nodes need to be removed by using the node deletion fact, then the corresponding edges should be deleted also by the edge deletion operation.

Edge insertion: Similar to the edge deletion operation, but this latter is used in conjunction with the node insertion operation. If some missing nodes need to be added by using the node insertion fact, then the corresponding edges should also to be inserted by the edge insertion fact.

The graph edit distance defines the overall cost of transforming from the ARG$_{g1}$ to ARG$_{g2}$ [14] using the function f as defined in (equation 2).

$$cost(f, g_1, g_2) = C_{node}(f, g_1, g_2)C_{edge}(f, g_1, g_2) \qquad (2)$$

Where C_{node} and C_{edge} are respectively the node and edge edit distance. These terms are defined in (equation 3).

$$C_{node}(f, g_1, g_2) = C^n_{sub} + C^n_{mer} + C^n_{spl} + C^n_{del}C^n_{ins} \qquad (3)$$

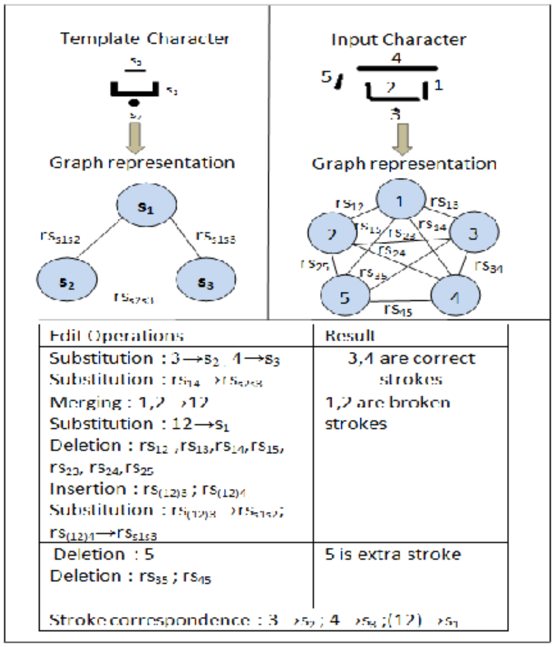
C^n_{sub}, C^n_{mer} , C^n_{spl} , C^n_{del}, C^n_{ins}, are respectively the costs of node substitution, merging, splitting, deletion and insertion. The edge edit distance C_{edge} is defined in (equation 4), such that C^e_{sub}, C^e_{del}, C^e_{ins} are the costs of the edge substitution.

$$C_{edge}(f, g_1, g_2) = C^e_{sub} + C^e_{del} + C^e_{ins} \qquad (4)$$

As an example of the graph matching between both ARGs and using the edit operations, we have the example of the character “bah”, as depicted in figure 11. Note that each edit operation has the appropriate meaning and the user can identify the handwriting errors easier.

The proposed approach tries to check the optimal ARG matching by generating a path that can transform the input graph g_1 to the template graph g_2. At the search level n, the evaluation function $h(n)$ (equation 5) , such that $h(n)$ is equal to the summation of respectively the actual matching cost $h_1(n)$ between the initial search level to the current search level n and the actual of the matching cost $h_2(n)$ from the current search level n to the goal.

$$h(n) = h_1(n) + h_2(n) \qquad (5)$$

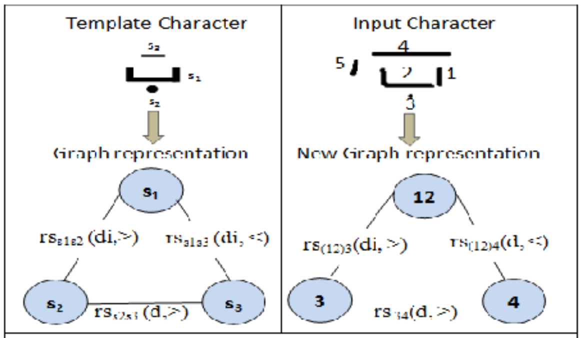


Edit Operations	Result
Substitution : $3 \rightarrow s_2$, $4 \rightarrow s_3$ Substitution : $rs_{14} \rightarrow rs_{s2s3}$ Merging : 1,2 →12 Substitution : $12 \rightarrow s_1$ Deletion : rs_{12} ,rs_{13},rs_{14},rs_{15}, rs_{23}, rs_{24},rs_{25} Insertion : $rs_{(12)3}$; $rs_{(12)4}$ Substitution : $rs_{(12)3} \rightarrow rs_{s1s2}$; $rs_{(12)4} \rightarrow rs_{s1s3}$	3,4 are correct strokes 1,2 are broken strokes
Deletion : 5 Deletion : rs_{35} ; rs_{45}	5 is extra stroke
Stroke correspondence : 3 → s_2 ; 4 → s_3 ;(12) → s_1	

Figure 11: Attributed relational graph matching procedure.

3.2.2. Errors detection

To detect the difference between two patterns of characters we apply the node and edges operations as defined previously. As the example in (figure 11), after applying the edit operations, we can get the stroke correspondence between the input and the template character. The stroke correspondence is: (1, 2) → s_1; 3 → s_2; 4 → s_3 (5 is an extra stroke , 1 and 2 are broken strokes, 3 and 4 correct strokes in the input character). A new graph is then obtained as shown in (figure 12). The edges in this latter describe the spatial relations between the strokes. Referring to (figure 12), there is a difference between the spatial relationships $rs_{s1s3}$and $rs_{(12)4}$. For this purpose, the spatial relationship along the x-axis for r_{ss1s3} is *di* and the spatial relationship along the x-axis for $rs_{(12)4}$ is *d*. We can conclude that the spatial relationship between the strokes (12) and (4) is incorrect compared to the corresponding template strokes s_1 and s_3.



	Relationship Correspondence		
	$rs_{s1s2} \rightarrow rs_{(12)3}$	$rs_{s2s3} \rightarrow rs_{34}$	$rs_{s1s3} \rightarrow rs_{(12)4}$
Spatial relationship along the x-axis	di – di	d – d	di $\neq$ d
Spatial relationship along the y axis	>> – >>	>> – >>	<< – <<

Figure 12: An example of the spatial relationship matching between strokes and the relationship errors process.

3.2.3. Feedback

After the character matching, we can check the difference between the template and input characters, and detect the different writing errors. By the means of the stroke correspondence, we can find the different types of errors such that: stroke interlines errors, stroke sequence errors, and the stroke production errors. The spatial relationship between the strokes is used to locate the relationship errors as defined in (figure 12). Then our system can check the handwriting mistakes and immediately provide a feedback to the users in order to correct them.

4. CONCLUSION AND FUTUR WORKS

In this paper, we have proposed an Arabic handwriting learning system with automatic detection of errors. We have investigated the Attributed Relation Graph (ARG) to represent the Arabic handwriting characters incorporating the spatial relationship information between strokes, and the spatial relationship information between the strokes and interlines.

We have applied the graph matching algorithm to find the difference between the handwriting template model and the input handwriting pattern. The used algorithm tries to find the optimal graph matching.

Our system can automatically check whether there is one or multiple errors in the handwriting input pattern, such as: the stroke production errors, stroke sequence errors, stroke relationship errors.

Finally, we will continue to handle with this system to make it easier for use and in order to improve the student's educational level. Extended applications for other languages may be also taken into account, and the possibility of integration on mobile phones enabled with handwriting inputs.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the financial support of this work by grants from General Direction of Scientific Research (DGRST), Tunisia, under the ARUB 01/UR/11/02 program.

REFERENCES

[1] Abdou S., Fahmy A., (2009) “Artificial Tutor for Arabic Handwriting Training,” , Proceedings of the Int. Conf. on Arabic Language Resources and Tools, Egypt-Cairo, pp. 24-31.

[2] Abdou S., Fahmy A. and Elgammal A., (2010) “A Tool for Arabic Handwriting Training”. The Fifth Conference of Learning International Networks Consortium (LINC), MIT, Cambridge, Massachusetts, USA, pp. 23-26.

[3] Ahmad T. Al-Taani, Al-Haj S., (2010) “Recognition of On-line Arabic Handwritten Characters using Structural Features”, Int. J. of Pattern Recognition Research 1, pp. 23-37.

[4] Alimi A., (1997) “An Evolutionary Neuro-Fuzzy Approach to Recognize on Line Arabic Handwriting”, Proc. Int. Conf. on Document Analysis and Recognition, ICDAR’97, pp. 382-386.

[5] Al-Jawfi. R., (2009) “Handwriting Arabic Character Recognition LeNet Using Neural Network”,The Int. Arab Journal of Information Technology, vol. 6, n°. 3, pp. 568-580.

[6] Allen J. F., (1983) “Maintaining Knowledge about Temporal Iinterval”, Communication of the ACM, vol. 26, n°. 11, pp.832-843.

[7] AL-Neaimi S., Harada N., Sasaki H., and Mizuno K., (2009) “Development of an Arabic Language Learning Support System”, Proceedings of the 17th Int. Conf. on Computers in Education. Hong Kong, pp. 486-489.

[8] Amin A., (1997) “Arabic Character Recognition, in Handbook of Character Recognition and Document Image Analysis”, H. Bunke and P. Wang, Eds. World Scientific, pp. 397-420.

[9] Gheith A. Abandah, Younis S. K, (2008) “Handwritten Arabic Characterusing Multiple Classifiers based on Letter Form” ,In Proc. 5th IASTED Int‟l Conf. on Signal Processing, Pattern Recognition & Applications SPPRA’08, Austria, pp. 128-133.

[10] Freksa C., (1992) “Temporal Reasoning Based on Semi-Intervals”, Artificial Intelligence, vol. 54, pp.199-227.

[11] Hammadi M., Bezine H., Njah S. and Alimi A. M., (2012) “Towards an Educational Tool for Arabic Handwriting Learning”, Int. Conf. on Education and E-Learning Innovations, Tunisia.

[12] Hu Z., Xu Y. , Huang L., Leung H., (2009) “A Chinese Handwriting System with Automatic Error Detection”, Int. J. of Software, Special Issue of Advanced Distance Learning Technologies, vol. 4, Issue 2, pp 101-107.

[13] Hu Z., Leung H. Xu Y., (2008) “Automated Chinese Handwriting Error Detection using Attributed Relational Graph Matching”, ICWL’2008, pp. 344-355.

[14] Hu Z. H., Leung H. and Xu Y., (2007) “Stroke Correspondence Based on Graph Matching for Detecting Stroke Production Errors in Chinese Character Handwriting”, Lecture Notes on Computer Science, vol. 4810 pp. 734-743.

[15] Klassen J. Tim , Malcolm Heywood I., (2002) “Towards the On-line Recognition of ArabicCharacters,”. IJCNN ‟02. Proc. of the Int. Joint Conf. on.Neural Networks, pp. 320-326.

[16] Lam H. C., Ki W. W., Law, Chung A. L. S., Ko P. Y., Ho A. H. S. and Pun S. W., (2001) “Designing CALL for Learning Chinese Characters”, J. of Computer Assisted Learning, vol. 17, n°. 1, pp. 115-128.

[17] Liu J., Cham W. K. and Chang M. M. Y., (1996)“Online Chinese character recognition using Attributed Graph Matching”. IEE Proc.-Vis. Image Signal Process, vol.143, pp.125-131.

[18] Muroya A., (2001) “Pen based Compute Handwriting Instruction” U. S. Patent n°. 5, pp. 222-227.

[19] Manjula R., Vaideeswaran J., (2011) “A New framework for Measuring the quality of Engineering Education System using SEI-CMM approach – (E2 – CMM), vol 2, n° 1, pp. 245-250.

[20] Nabil M., Ngu A. H. H. and Shepherd J., (1996) “Picture Similarity Retrieval using the 2D Projection Interval Representation”. IEEE Trans. Knowledge and Data Engineering, vol. 8, n°. 4, pp.533-539.

[21] Noma A., Graciano A., Cesar R., Consularo L., and Bloch I., (2011) "Interactive Image Segmentation by Matching Attributed Relational Graphs", Int. J. of Pattern Recognition, vol. 45, n°21, pp26-59.

[22] Plamondon R., Lopresti D., Shomaker L. et Srihari R., (1999)"On Line Handwriting Recognition", Enceclopidia of Electrical and Electronics, New York, Wiley, vol 15, pp. 123-146.

[23] Saleh, S., (1998)"An On-Line Automatic ArabicDocument Reader", MSc Thesis, University of Basrah, Iraq.

[24] Tang K. T., and Leung H., (2006) "Reconstructing the Correct Writing Sequence from a Set of Chinese Character Strokes", Proc. of the 21st Int. Conf. on Computer Processing of Oriental Languages ICCPOL, pp. 333-344.

[25] Tang K. T., Li K. K. and Leung H., (2006) " A Web Based Chinese Handwriting Education System with Automatic Feedback and Analysis", Proc. of the Int. Conf. ICWL'06, pp. 176-188.

[26] Tonouchi Y. Kawamura A., (1997) "An On line Japenese Character Recognition Method using Length-Based Stroke Correspondence Algorithm", Proc. Int. Conf on Document Analysis and Recognition, ICDAR'1997, pp. 633-636.

[27] http://www.declan-software.com/index.htm.

Authors

Hala Bezine is an IEEE member. She received Engineering Diploma from the National School of Engineering of Sfax in 1996, the Diplôme d'Etudes approffondies in Automatics in 1998 and a Ph.D. in Electrical Engineering in 2006. Currently, she is an associate professor of Electrical and Computer Engineering at the University of Sfax. Her interests include cursive handwriting modeling and analysis, pattern recognition, robotic systems and control of complex human movements.

Adel M. Alimi is an IEEE senior member of IAPR, INNS. He graduated in Electrical Engineering in 1990, obtained a Ph.D. and then an HDR both in Electrical Engineering in 1995 and 2000, respectively. He is now an associate professor in Electrical and Computer Engineering at the University of Sfax. His research interest includes of intelligent methods to pattern recognition, robotic systems, vision systems, intelligent pattern recognition and industrial processes. He is the associate editor of the international journal Pattern Recognition Letters.