Morphological Skeleton Traversal Algorithm for Person Recognition based on Handwritten Stroke Analysis

Gonzalo Peña, Marco Mora and Karen Hodges
gpena@litrp.cl,mora@spock.ucm.cl
Laboratory of Technological Research on Pattern Recognition
Department of Computer Science
Universidad Católica del Maule, Chile

***Abstract*—Traversing in the sense and direction of the writing the morphological skeleton of a grapheme, is a key element in the construction of a series of recent descriptors for person recognition based on the analysis of strokes of handwriting. In this paper, the traversal of the skeleton as a problem of traversal in graphs is formulated and an algorithm based on the Tables of Priority and Backtracking and Bound is presented. A wide base of data of more than 4500 skeletons of the grapheme "e" is constructed to carry out the experiments; it is shown that the theoretical formulation of the problem and the employed algorithmic strategy allows encompassing successfully complicated things of skeletons that possess crosses.**

I. Introduction

The handwriting is a controlled activity for the neuromotorapparatus of man, thus, the form of writing becomes an own characteristic of the individual, that responds to physiological and psychological aspects of the person [1]. The morphological characteristics of the strokes (static characteristics) are possible to imitate, however the information associated to the writing action (dynamic characteristics) could be used in effective ways of person recognition. Just as it is shown in [2],the formal aspect of the letters, its form and dimension, could be modified easily in conscious ways. But in the handwriting there is a hierarchy of symbols, and characteristics as the depth, intensity, stress, swiftness, direction, continuity that are possible to change.

One of the most studied dynamic characteristics is the stress by which a person writes. Currently, there are inexpensive devices that allow to do on-line estimations of this variable [3]. However, more two decades ago, there is concluding proof that the stress of the handwriting could be estimated properly from the image of the strokes in gray scale [4].The works that propose descriptors and simplified ways to represent the pattern of stress are uncountable.

One recent line of this type of descriptor is born in the presented study in [5], in which an approximation to person recognition is developed based on the analysis of individual letters o graphemes, and different descriptors are proposed based on the relative position of the points of greater stress or points of less value of gray in the image of the grapheme.

One of the mentioned descriptors is reported in [6] and corresponds to the Distance among the Points of Minimum Gray and the Morphological Skeleton of the Grapheme, the stroke is divided into segments inspiring in the zones of the studied graphemes in graphology[7], and the average proposed distance in each segment was calculated. In [8] preliminary results of the performance of descriptors belonging to the Distance among the Points of Minimum Gray ad the Morphological Skeleton of the Grapheme; a methodology of evaluation of the descriptor is proposed using the neuronal networks, experiments upon a small database are carried out, and it is concluded that the descriptor has a high rate of correctness if a classifier for linearly inseparable data is used.

In the previously mentioned works, the morphological skeleton of the grapheme is a key element, since the points of minimum gray are found on the perpendicular line upon it. For the appropriate compound of the descriptor, the resulting skeleton must not have ramifications, the pixels of the skeleton must have 2 neighbors, except in the extreme points where there is only one, and in the crossings where there are more than two neighbors. The previous characteristics are obtained through the application of robust algorithms for the computing of the skeleton [9], and an adequate processing of the binary image of the grapheme just as proposed in[10]. In order that the descriptor become more representative of the form in which it is written, the points of the minimum gray could be ordered in the sense and direction of the handwriting, and through this way to condense more dynamic information in the own descriptor. In zones where the pixels of the skeleton that has only two neighbors, ordering the pixels in the sense and direction of the handwriting that has an evident solution, thus there is only one way to traverse. The problem emerges when the involved graphemes having crossings and the resulting skeleton having in the zone of the crossing more than one pixel with bifurcations (case of the grapheme "e"). In [10] an initial solution is proponed for the previous problem, which consists in deciding the next pixel for checking based on a Table of Priorities. In this table the priorities of the pixels neighbors in the points of bifurcation are defined, depending on if the traversal is of toward or backward, among other criteria.

In this work, considering the idea of the Table of Priorities of [10], firstly the traversal of the skeleton as a problem of traversal in graphs is formulated, and an algorithm based on a strategy of Backtracking and Bound for traversing the crossings properly is proposed. It has generated a database of more than 4000 cases of skeletons with crossings of the grapheme "e" to carry out the experiments. The methodological conceptualization of the problem,and the employed algorithmic strategy have allowed obtaining an elevated performance of correctness of correct way, and traversing properly the skeleton of the grapheme in the sense and direction of the handwriting.

The structure of the work is the following: the section 2 presents the generation of the database of images, the section 3 explains the strategy of traversal of the skeleton based on Tables of Priorities, the section 4 details the proposed solution based on Tables of Priorities and Backtracking and Bound, in the section 5 the results of the experiments are presented, and finally in the section 6 the conclusions of the work are given in.

II. Procedure of Skeleton Obtaining

A to the instrument of handwriting, in this work a blue ink ball pen (the trademark is BIC) is incorporated. The database of experiment has more than 4000 images of grapheme "e" belonging to 36 people. .

To recollect the samples a sheet of white letter-size $75(gm^2)$ of density was used. In each paper a grill of $10\ times 10$ boxes was printed. In each one of the box in the grill the person wrote a sample of the grapheme. To obtain the digital image of the graphemes a conventional scanner of $1200(dpi)$ of resolution was used. The image of a grapheme has in average 850×900 pixels.

A. Binarization of the image in color

The image of the grapheme is originally in color. To binarize the image two steps are carried out: Firstly, the image in color is converted into an image in gray scale. For that, the image is transformed from the RGB model (original of the image) to the HSV model of color. This model is employed since the information of color in it is separated (Tone and Saturation) from the information of intensity of the image. As image in gray scale consider the V channel of the HSV model, due to this channel allows a representation of the intensity with low loss of visual information. Secondly, the image is binarized into gray scale considering, in this case, the known algorithm of Otsu [11].This algorithm finds automatically the threshold that is employed for the binarization.

The figure 1 shows the steps for binarizing the image. The figure 1(a) corresponds to the original image in color, the figure 1(b) the V channel of the HSV model, and the figure 1(c) the binary image. Just as observed in the figure 1(c), the grapheme has discontinuity due to the current imperfections in the image in color.These imperfections are due to reasons like: low concentration of ink because of the raising of the instrument of writing, paper features and the base of support [5].

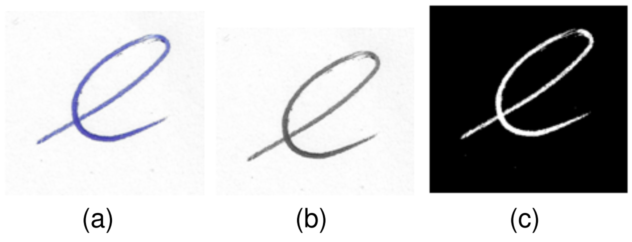


Figure 1: Binarized Image

B. Improvement of the binary image

The binary image of the grapheme must not have discontinuities so that its skeleton could be a continuous line of a pixel. Due to the former, it is necessary to implement a step of improvement of the binary image, which is divided into four steps:

1) To fill the gaps and obtain a solid grapheme a Morphological Dilatation is applied. This operation is considered thus simple, effective, and of low computational cost. Just as observed in the figure 2(a), the dilatation generate a solid grapheme but with a bigger size with respect to the original image.
2) The dilatation produces noise in the background of the image. To eliminate this noise, a filter of objects according to their size is implemented. For this, firstly it is calculated that the size of the object is much bigger, than in the binary image that corresponds to the grapheme. A threshold with regard to the size of the bigger object (typically 5%) is stablished. Finally all of the objects of the image that have a smaller size to the threshold are eliminated, such as observed in the figure 2(b).
3) To obtain smooth edged and representatives of the original grapheme, the binary image is smoothened by a filter of middle bi-dimensional with a big neighborhood (typically of 10×10). The effect of the smoothening of edged is observed in the figure 2(c).
4) Fort the growth of the object due to the dilatations, a Morphological Erosion is carried out to recover the original size of the binary stoke. The reduction of size with respect to the smoothened image is observed in the figure 2(d).

C. Computing of the skeleton

From the binary image without holes of the grapheme the morphological skeleton is computed, with the object of obtaining an estimation of the central line of the stroke. The proposed method is employed in [9] thus removes the pixels in such a way that an object without holes is reduced to a minimum concrete line.

The figure 3 shows the step of skeletization. The figure 3(a) shows the skeleton of the grapheme, which corresponds to a continuous line and of 1 pixel. The figure 3 (b) shows a zoom-in of the central part of the skeleton, observing that in the crossings points with bifurcations are produced.

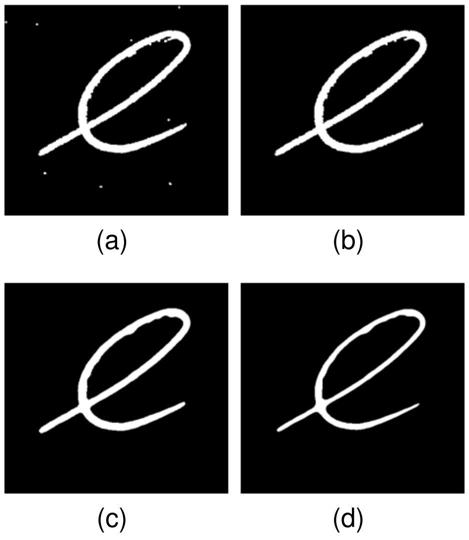


Figure 2: Original Binarized Image

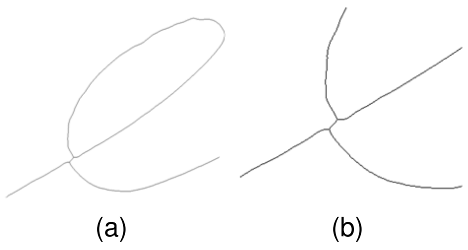


Figure 3: Skeleton of binary image

III. VORACIOUS ALGORITHM FOR TRAVERSAL OF THE SKELETON WITH CROSSINGS BASED ON THE TABLES OF PRIORITY

The method of traversal of the skeleton based on the Table of Priority proposed in [10]. In the former work shows an important evaluation of the descriptor for person recognition based on the analysis of handwriting strokes of [6].

Considering that the pixels of the skeleton of the grapheme have 2 neighbors, and that in the crossings the pixels could have more than 2 neighbors, the ordered traversal of the skeleton is carried out in the following way:

- Initially the beginning and ending pixels of the skeleton should be determined.
- Coming from the initial pixel, the neighboring pixel is located is in a neighborhood of 3×3, and is going Forwards to the mentioned pixel.
- For the rest of the pixels of the skeleton, since the pixels have 2 neighbors, it is considered as neighbor of that has not been visited. For the case of the points of the crossing, which have more than 2 neighbors, to pick the proper neighbor, the direction of the traversal of the skeleton should be, furthermore, considered.
- Each visited pixel is saved in an ordered vector according to what the visits are going to produce.

The solution to the problem consider 8 directions of the neighbors to one pixel of the crossing. North (N), Northern-East (NE), Southern-East (SE), South (S), Southern-West (SW), West (W), Northern-West (NW). These directions are presented based on arrows that point to these directions $(N = \uparrow, NE = \nearrow, E = \rightarrow, SE = \searrow, S = \downarrow, SO = \swarrow, O = \leftarrow, NO = \nwarrow)$.

Another element that consider this proposal is that, being in a pixel of crossing, it is necessary to determine if it is traversing the skeleton of Forwards and backwards (this happens in the skeleton of the grapheme “e”). To give solution to the former, it is assumed as direction of traversal that is calculated between 2 points $p1$ and $p2$ that have been visited, and are distanced respectively of $d1$ and $d2$ from the point of crossing.

In most the of the obtained skeletons there are 2 points of crossing, as a matter of fact, being in a pixel of the crossing there are two ways to follow the traversal. These ways could be differentiated through their directions. To choose between these options, a List of Priority is created for the 8 neighboring pixels. With all the previous, being in a point of the crossing, it is progressed according to the following steps:

- It is determined if the traversal of the skeleton goes forwards or backwards.
- It is determined if it is in the first or second crossing.
- The directions of the neighbors are determined.
- The neighbor is picked according to a Table of Priorities.

To construct a Table of Priority it is necessary to consider 2 elements: if the traversal goes forwards or backwards, and if that it is traversing is the first or the second crossing. The former implies that there are 4 different cases, which generate 4 Tables of Priority:

- Case 1: The traversal goes forwards, and it is in the first crossing, just as shown in the figure 4(a). The priorities for picking the next pixel are shown in the column case 1 of the table I.
- Case 2: The traversal goes forwards, and it is in the second crossing, just as shown in the figure 4(b). The priorities for picking the next pixel are shown in the column case 2 of the table I.
- Case 3: The traversal goes backwards, and it is in the first crossing, just as shown in the figure 4(c). The priorities for picking the next pixel are shown in the column case 3 of the table I.
- Case 4: The traversal goes backwards, and it is in the second crossing, just as shown in the figure 4 (d). The priorities for picking the next pixel are shown in the column case 4 of the table I.

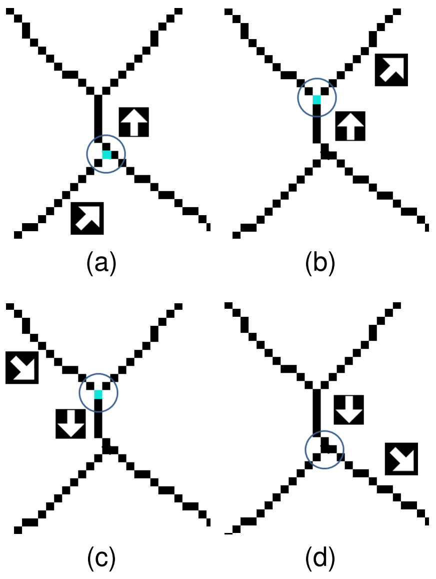


Figure 4: Points of crossings and Directions

-	Traversal Forwards		Traversal Backward	
Priority	**Case 1**	**case 2**	**Case 3**	**Case 4**
1	↗	↗	↘	↘
2	↑	→	↓	↓
3	↖	↑	↙	→
4	→	↖	→	↙
5	←	←	←	←
6	↘	↘	↗	↗
7	↙	↙	↖	↖
8	↓	↓	↑	↑

Table I: Table of Priority

IV. TRAVERSAL OF THE SKELETON BASED ON BACKTRACKING AND BOUND AND TABLES OF PRIORITY

A. Formulation of the traversal of the skeleton as Traversal in related graphs

The pixels of the skeleton correspond to the nodes of the graph. Due to that the traversal of the skeleton is in the sense and direction of the handwriting, the graph is conducted, e.g., a di-graph. The priority assigned to a neighboring pixel could be considered as the value assigned to the edges of the graph, for this, the skeleton could be considered as a related etiquette di-graph.

From the conceptual point of view, the strategy of priorities of [10] implements a voracious type of algorithm to solve the problem of traversal in the crossings. The voracious algorithms decide for the solution that is considered the most promising one, and the followed act proves the feasibility of the solution, without the possibility to analyze afterwards the goodness of the chosen solution. The advantage of this type of algorithms is that they have a low computational complexity and are easy to implement.

B. Strategy based on Tables of Priorities and Backtracking and Bound

In this problem in particular, it is not possible to determine if the most promising solution is the correct solution. What actually happens is that, in a pixel of crossing the neighboring pixel of priority interest does not necessarily conduct a way in the sense and direction of the handwriting. For this in this work a way is proposed to maintain the idea of the priorities, but a strategy of Backtracking and Bound for the traversal is implemented. This allows via a validation afterwards that a solution that is not the most promising but the correct one to be chosen. In particular, initially a search is carried out in depth, e.g., considering all the feasible ways.

Starting from the neighboring pixel of major priority, the way is traversed and afterwards it is validated if this conducts to a correct solution. If the chosen solution is correct, it can trim and eliminate the rest of the solutions. If it is not, the way determined by the pixel that follows in priority in the table will start to be traversed.

A first strategy of validation of the way that is being traversed consists in identifying if the traversal goes forwards and backwards. The algorithm has an adjustable parameter that indicates how many pixels after the last visited crossing must be traversed to carry out the validation. Being in a crossing and the traversal goes forwards, the pixel determined by the parameter must have direction NE respect to the crossing. Being in a crossing and the traversal goes backwards the pixel determined by the parameter must have direction SE respect to the crossing. The figure 5 shows the behavior of the previous strategy and the correct traversal of the skeleton in the crossings. In the figure 5(a) the Case 1 is shown, where the traversal goes forwards, and the first point of crossing is faced. How the strategy of validation detects an incorrect way is shown, and it goes back to the point of crossing. In analogous way, the figures 5(b) and 5(c) show the appropriate performance of the algorithm for the Case 2 and Case 4 respectively. Moreover, a validation is carried out to avoid cyclic traversals produced in the zones of the crossing. It is correct to visit a point of crossing per second, which is a cycle, every time when the characteristic link of the grapheme has already traversed. The figure 6(a) shows the cycle of the grapheme “e”. The figure 6(b) shows the point of crossing of the beginning and of the end of the cycle. Finally the figure 6(c) shows a cycle produced in the zone of crossing. The strategy of validation for the cycles consists in considering invalid cycles with a small number of visited points.

Respect of the complexity of the algorithm the following could be said: Since the grapheme “e” has no more than 10 points of crossing, and each one of them has more than 3 bifurcations, the algorithmic complexity does not drastically increase when there are many crossings. Other than the previous, when in the crossings all of the correct ways are chosen, the algorithm function identically based on Tables of Priority with linear complexity.

With the purpose of providing a more pedagogical expli-

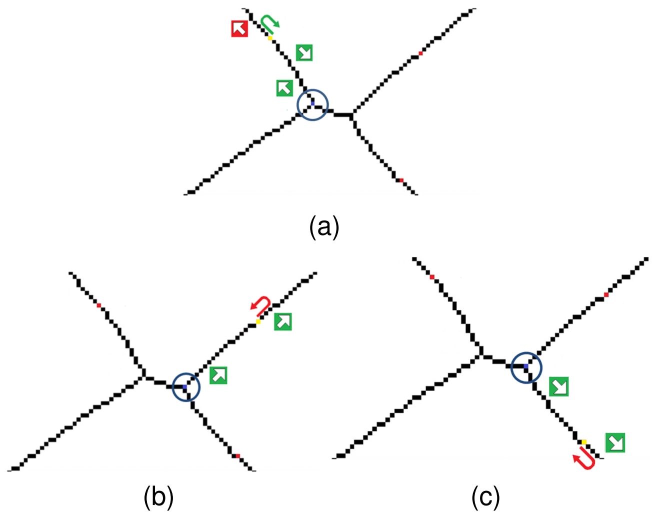


Figure 5: Tracking of algorithm

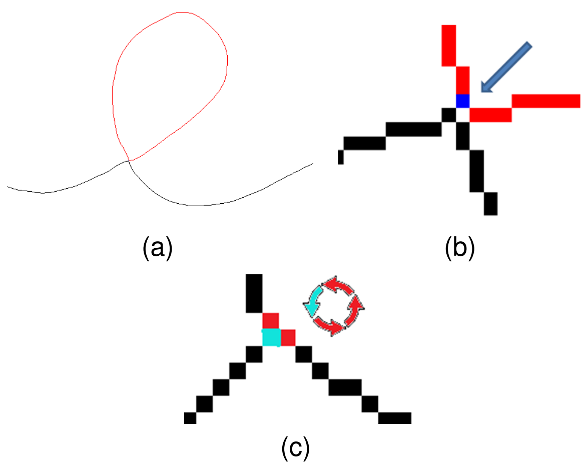
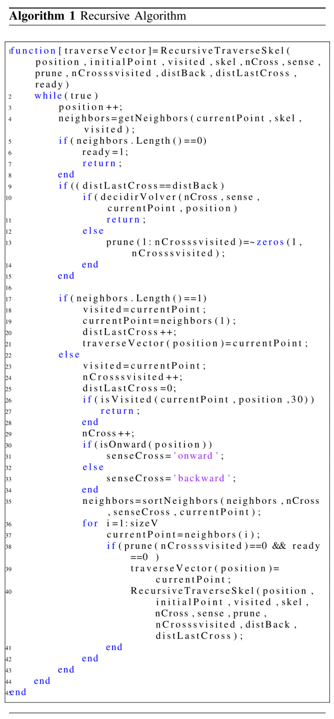


Figure 6: Cycles

cation, next a pseudo-code of the algorithm is presented. The involved variables and functions are the following:

- Variables and vectors:
  - initialPoint: Variable that stores to coordinates (x, y) of the starting point.
  - position: Variable that indicates the position of the current point in the vector of the traversal.
  - currentPoint: Variable that indicates the coordinates (x, y) of the current point.
  - visited: Variable that stores temporarily the previously visited point.
  - senseCross: Variable that indicates if it is in the way goes forwards or backwards.
  - nCross: Variable that indicates the number of the crossing that is being interfered.
  - prune: : Vector that indicates in each crossing if it is necessary to prune the way. It will start with 0s to indicate that initially there is no pruning.
  - nVisitedCross : Variable that indicates the number of visited crossings.
  - distBack: Parameter that stores the distance from the last visited crossing to the point of validation.
  - distLastCross: variable that stores the distance between the current point and the last visited crossing.
  - skel: Variable that stores the skel of binary image.
  - neighbors: Variable that stores neighbors of currentPoint.
  - ready: Pointer or global variable that it indicate the solution was found. It is initialized in 0.
  - traverseVector: Vector that stores the ordered visited pixels .
- Functions:
  - getNeighbors: Function that returns a vector with the neighbors of a point of the different from the last visited.
  - zeros: Function that creates a vector of 0s.
  - isVisited : Function that indicates if currentPoint has been visited distVisit points go backwards.
  - decideBack : Function that decides if it is necesary to continue through a way.
  - isOnward : Function that it indicates if current sense traverse is onward.
  - sortNeighbors : Sorting according priority table.
- Methods:
  - vector.Length:Method that indicates length of vector.



Algorithm 1 Recursive Algorithm

```
function[traverseVector]=RecursiveTraverseSkel(
    position ,initialPoint ,visited ,skel ,nCross ,sense ,
    prune ,nCrosssvisited ,distBack ,distLastCross ,
    ready)
    while(true)
        position++;
        neighbors=getNeighbors(currentPoint ,skel ,
            visited);
        if(neighbors.Length()==0)
            ready=1;
            return;
        end
        if((distLastCross==distBack)
            if(decidirVolver(nCross ,sense ,
                currentPoint ,position)
                return;
            else
                prune(1:nCrosssvisited)=~zeros(1,
                    nCrosssvisited);
            end
        end

        if(neighbors.Length()==1)
            visited=currentPoint;
            currentPoint=neighbors(1);
            distLastCross++;
            traverseVector(position)=currentPoint;
        else
            visited=currentPoint;
            nCrosssvisited++;
            distLastCross=0;
            if(isVisited(currentPoint ,position ,30))
               return;
            end
            nCross++;
            if(isOnward(position))
                senseCross='onward';
            else
                senseCross='backward';
            end
            neighbors=sortNeighbors(neighbors ,nCross
                ,senseCross ,currentPoint);
            for  i=1:sizeV
                currentPoint=neighbors(i);
                if(prune(nCrosssvisited)==0 && ready
                    ==0  )
                    traverseVector(position)=
                        currentPoint;
                    RecursiveTraverseSkel(position ,
                        initialPoint ,visited ,skel ,
                        nCross ,sense ,prune ,
                        nCrosssvisited ,distBack ,
                        distLastCross);
                end
            end
        end
    end
end
```

(a)

Figure 7: Pseudo-code

V. Results

It is this session that the results of the traversal of the skeleton produced by the voracious algorithm and the one based on Backtracking and Bound are shown. These are considered for a total of 4860 images of the graph “e”. It is important to say that in order that both algorithms have a correct functioning, the determined characteristics of the Morphological Skeleton (Ideal Skeleton) such as: being a single line of a pixel, and not having ramifications must be accomplished. The previous only achieves if the binary image of the grapheme has no gaps.

The table II shows the results of both algorithms about the total of the skeleton. It is understood as correctness when an algorithm traverses the skeleton in the sense of the handwriting. It is observed that the algorithm based on Backtracking and Bound has a substantially better performance than the one based on the voracious one. The errors of the both proposals in big part are due to the skeletons contain ideal characteristics. The figures 7 and 8 show imagesthat produce ideal skeletons, and faulty skeleton.

Table II: Results on total samples

Algorithm	**% correctness**	**% Errors**
Tables of Priority	72%	28%
Backtracking and Bound	93%	7%

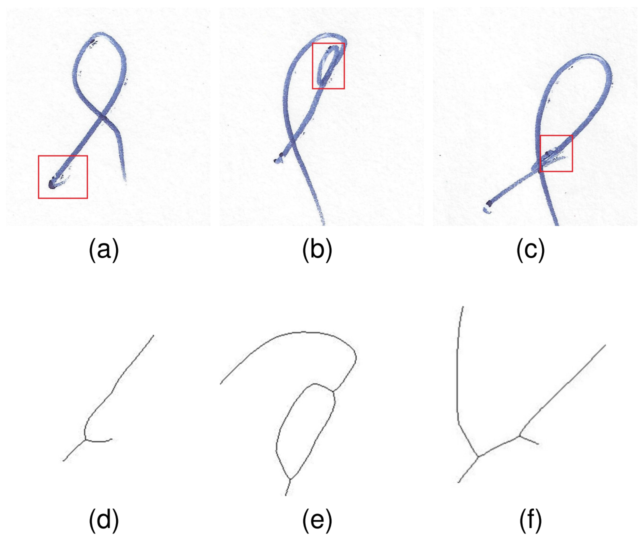


Figure 8: Not ideals Skeletons

The table III shows the results of the experiments on ideal skeletons. It is observed with clarity that the proposed algorithm traverses corrctly almost all the skeletons.

Table III: Results on ideals skeletons

Algorithm	**% crrcteness**	**% Errors**
Voraz	77%	23%
Backtracking	99%	1%

VI. Conclusion

The Morphological Skeleton of the grapheme is a key element for the construction of a recent one of descriptors for people recognition based on analysis of handwriting text.

In this work an algorithm has been presented for traversing in the sense and direction of the handwriting and morphological skeletons of graphemes with crossings. The proposal of the algorithm was based on the formulation of the traversal of the skeleton as a problem of traversal in graphs, and an algorithmic strategy based on Tables of Priorities and Backtracking and Bound.

It is important to emphasize that the algorithms of skeletization are a problem of investigation itself, that is not the focus of this work. Nevertheless the previous, based on the employed method of skeletization, the proposed step of processing for the binary image of the grapheme, and the proper theoretical formulation of the problem have allowed obtaining an elevated rate of correctness of traversal.

Acknowledgment

The authors of this paper are grateful to Verónica Aubin, Universidad Nacional de La Matanza, Argentina. The aforementioned colleague generously shared with us her experience in the field of People Recognition based on Handwritten Strokes. Her valuable collaboration has allowed to develop this research. Special appreciations to professor ChuanChih, Hsu who had collaborated in the language usage in the English version of this article. ChuanChih, Hsu chuan@ucm.cl Teacher of English Area of Languages, Institute of StudiumGenerale Católica del Maule University, Chile

References

[1] F. Vinals and M. Puente, *Pericia Caligrafica Judicial, Practica, Casos y Modelos*, 2nd ed. Herder, 2006.

[2] J. Crepieux-Jamin, *A B C de la Graphologie*, 2nd ed. Presses Universitaires de France, 1950.

[3] D. Dimitrova and G. Gluhchev, “Pressure Evaluation in On-Line and Off-Line Signatures,” in *Biometric ID Management and Multimodal Communication*, ser. Lecture Notes in Computer Science, vol. 5707, 2009, pp. 207–211.

[4] M. Ammar, Y. Yoshida, and T. Fukumura, “A new effective approach for off-line verification of signatures by using pressure features,” in *Proc. 8th ICPR*, 1986, pp. 566–569.

[5] V. Aubin, “Estimacion de parametros identificarorios en trazos manuscritos mediante procesamiento de imagenes,” Master’s thesis, Escuela de Posgrado de la Universidad Nacional de la Matanza, Florencio Varela 1903, San Justo, Buenos Aires, Argentina, 2013.

[6] V. Aubin, J. Doorn, and G. Kaplan, “Nuevos Descriptores para la Identificacion de Personas basados en la Simetria del Trazo,” in *XIX Congreso Argentino de Ciencias de la Computacion CACIC 2013*, October 2013.

[7] A. Vels, *Grafología de la A a la Z*, 1st ed. Herder, 2000.

[8] V. Aubin and M. Mora, “A Descriptor for Handwritten Strokes Off-Line Analysis: A Preliminary Study,” in *Proceedings of V Chilean Workshop on Pattern Recognition, ISBN 978-956-353-484-9*, November 2013, available On-Line on www.litrp.cl/cwpr2013.html.

[9] L. Lam, S. Lee, and C. Suen, “Thinning methodologies-a comprehensive survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 9, pp. 869–885, 1992.

[10] K. Hodges and G. Pena, "Computo de un Nuevo Descriptor para el Analisis Off-line de Trazos Manuscritos," 2014, Tesis para obtener el Titulo de Ingeniero en Computacion, Escuela de Ingenieria en Computacion, Universidad Catolica del Maule. Dr. Marco Mora UCM-Chile (guia), M. Inf. Veronica Aubin UNLAM-Argentina (revisor), Ing. Alex Zuniga UCM-Chile (revisor). Disponible en http://www.litrp.cl/jm/es/documentos.html.

[11] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.