Handwritten Arabic (Indian) Numerals Recognition Using Fourier Descriptor and Structure Base Classifier

Shatha M. Noor*, Ihab A. Mohammed** and Loay E. George***
*Department of Computer Science, College of Science, University of Al-Nahrain, Baghdad-Iraq.
**Department of Computer Science, College of Science, University of Al-Nahrain, Baghdad-Iraq.
***Department of Computer Science, College of Science, University of Baghdad, Baghdad-Iraq.
E-mail: shatha_alhassany@yahoo.com, eihabmurjan@yahoo.com, loayedwar57@yahoo.com.

Abstract

In this paper a simple and accurate method is proposed to recognize Arabic (Indian) numerals using Fourier descriptors as the main classifier feature set, and to improve the recognition accuracy a simple structure based classifier is add as a supplementary classifier. The recognition system was tested on 450 samples collected from 5 students and the test results indicate that the recognition ratio is%89.6when only 5 Fourier descriptors are used as discriminating features set, and the ratio is raised to%98 when 4 Fourier descriptors are used in addition to the simple structure based classifier.

Keywords: Digit Recognition, Binarization, Thinning, Segmentation, Edge Connection, Chain Codes, Fourier Descriptor, Structured Classifier.

1. Introduction

The need to process documents on paper by computer has led to an area of research that may be referred to as document image understanding (DIU). The goal of a DIU system is to convert a raster image representation of a document, (e.g., a paper document scanned by a flatbed document scanner, into an appropriate symbolic form). DIU as a research endeavor consists of studying all processes involved in taking a document through various representations; i.e. from a scanned or facsimile multi-page document to high-level semantic descriptions of the document. Thus it involves many sub-disciplines of computer science including image processing, pattern recognition, natural language processing, artificial intelligence and database systems [1].

Digits recognition remains one of the most challenging problems in pattern recognition. It is used in ZIP code recognition, reading of bank checks, postal sorting, and automatic data entry. Several researchers have reported the recognition of Parisian (Farsi) and Arabic (Indian) handwritten digits [2-11], see Fig.(1), most of these researches are based on neural network. For historical review of OCR researches and development, Shunji [12], Khorsheed [13] and Lorigo[14] have discussed some of the offline Arabic character recognition methods. Ahmedand Al-Ohali [15] have reviewed the progress and challenges of Arabic character recognition. Trier et al [16] present a survey on feature extraction methods for character recognition, in this paper a simple and efficient method is proposed based on Fourier descriptors and structure based classifier.

Fig.(1) Arabic digits samples from digit one to digit nine.

2. The Proposed System

Fig.(2) shows the proposed recognition system. As first step, each digit image segment preprocessed; it is converted to gray image, and then to binary using thresholding method. The binary digit object image is narrowed using thinning algorithm, then its points are collected using contour follower algorithm. If the digit object contains defects then it is repaired using edge connection and morphological based methods. The digit object start pixel is allocated to create the chain code. The registered coordinates of the collected digit object chain code points are normalized. At the last step, the descriptive features for each segment are calculated using Fourier Descriptors with structure based classifier.

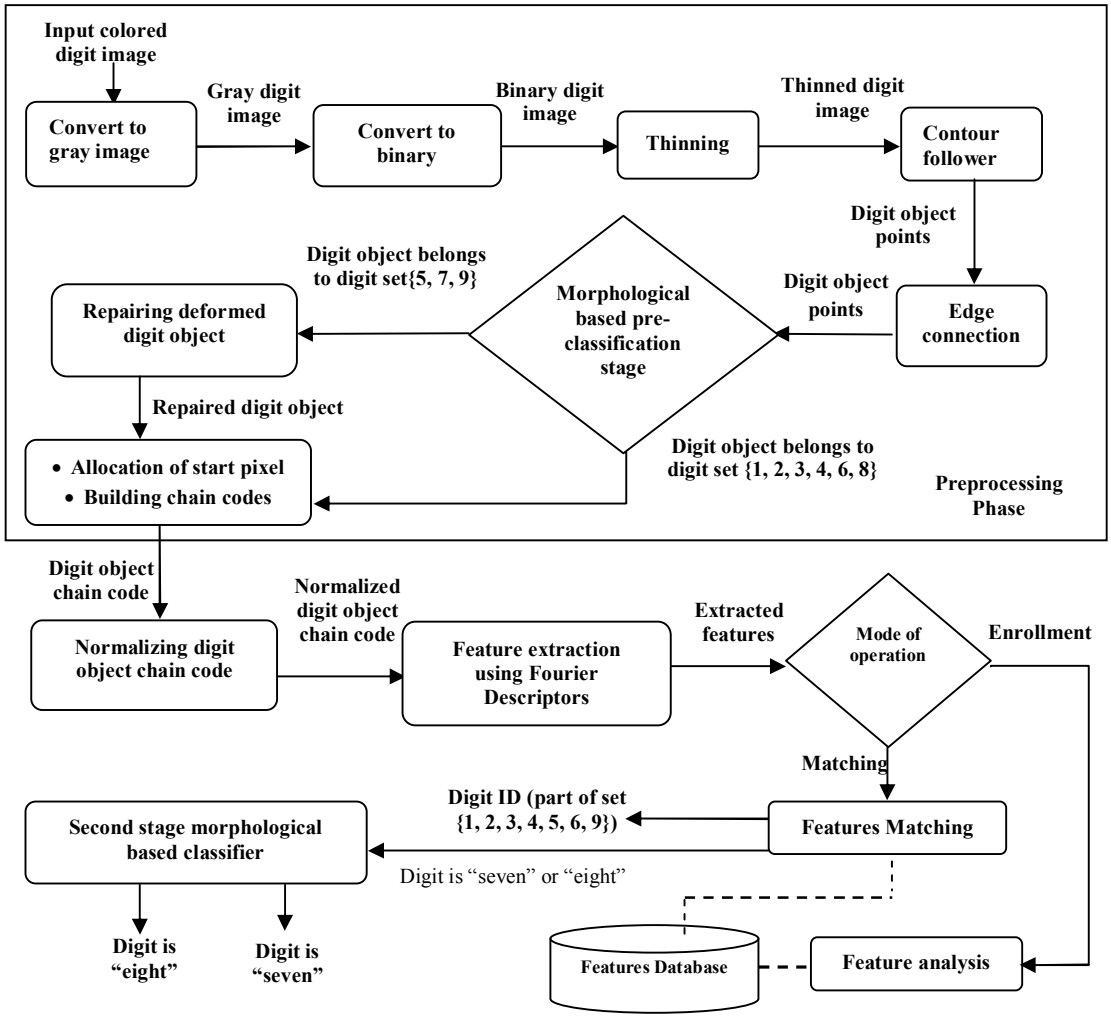


Fig. (2) The proposed system.

2.1 Conversion to Binary

The digit image is converted to grayscale using the following equation

$$Gry_{xy} = \frac{R_{xy} + G_{xy} + B_{xy}}{3}$$

At the next step, the gray image is converted to binary image using thresholding method [17], the test trails found the threshold value of 190 leads to best separation between the background and the digits object. The image was thinned using the Hilditch algorithm [18].

2.2 Segmentation

The digit object's pixels are extracted by scanning the image pixel from top to bottom and from left to right looking for object pixel (with gray value = 0).Once, an object point is met then the contour follower algorithm is activated. The code list shown below illustrates the implemented steps of contour follower algorithm which is used to extract all the connected black pixels as a one segment and then assign the terminal pixels of the collected segment.

```
Algorithm: Contour Follower
Input: x and y coordinates of a black pixel.
Description: collect segment pixels by changing all
connected pixels with gray level 0 to gray level 200
starting from the given pixel.

segment(0).x = given x
segment(0).y = given y
image(x, y) = 200
cp = 0
lp = 0
While cp<= lp do
  x = segment(cp).x
  y = segment(cp).y
  tco = 0
  For i = -1  to  1  do
    For  k = -1  to  1  do
      If Image(x+i,y+k) = 0 then
        Image(x+i,y+k) = 200
        lp = lp + 1
        segment(lp).x = x+i
        segment(lp).y = y+k
        tco = tco + 1
      Else If Image(x+i,y+k) = 200 then tco = tco + 1
      End If
    End For
  End For
  If tco = 1 then set the current pixel as a terminal
  cp = cp + 1
Repeat
```

2.3 Edge Connection

During the binarization process, some pieces of the digit object may lost which causes the break of digit object into pieces. To hand this problem an edge connection algorithm to connect the terminals of disconnected pieces of the same digit segment is introduced. It is used with the segmentation algorithm. As an example Fig.(3-a) shows a segment disconnected into two pieces, they must be connected in order to capture the whole body of digit "six" object, Fig.(3-b) shows the terminal point (x_2, y_2) and a fourth point connected to this one (x_1, y_1), from these two points the algorithm decide which direction to search in by computing the slope of the line between these two points as illustrated in Fig.(3-d), (3-e), (3-f), and (3-g). Fig. (3-c) shows the found terminal point and connects it with the point at (x_2, y_2) to restore digit "six". Some researchers like Rajput et al [20] have used dilate and erode morphological operation for connecting edges and filling gaps, but this method fails in many samples because it can't connect terminals with long distance between them and it may connect terminal pixels of the same segment which causes a distortion in the segment itself.

```
Algorithm: Edge Connection
Input: x and y coordinates of the current digit
segment terminal pixel.
Output:  segment including other connected nearby
segments.

Set x2 = given x
Set y2 = given y
Step backward and find (x1, y1) of the fourth pixel
connected to this given terminal pixel.
Set dx = x2 – x1
Set dy = y2 – y1
If dx = 0 then
   dir = vertical
   If y2< y1 then inc = -1
   Else inc = 1
   End If
Else
   slope = |dy/dx|
   If slope > 1 then
      dir = vertical
      If y2< y1then inc = -1
      Else inc = 1
      End If
   Else
      dir = horizontal
      If x2< x1 then inc = -1
      Else inc = 1
      End If
   End If
End If
If dir = vertical then
   For y = 1 to 10 do
      For x = -y to y do
         If Image(x2+y, y2+inc*y) = 200 and is a
            terminal then
               connect (x2, y2) with (x2+y, y2+inc*y)
               using Bresenham algorithm
         End If
      End For
   End For
Else
   For x = 1 to 10 do
      For y = -x to x do
         If Image(x2+inc*x, y2+y) = 200 and is a
            terminal then
              connect (x2, y2) with (x2+inc*x, y2+y)
              using Bresenham algorithm
         End If
      End For
   End For
End If
```

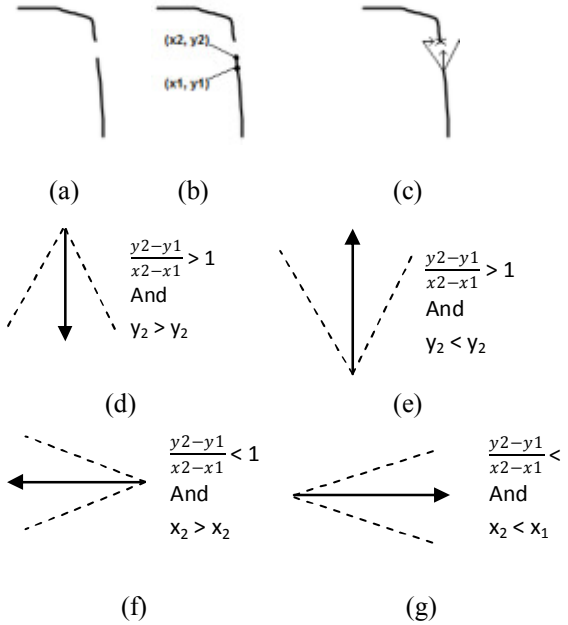


Fig.(3) The connection of edges of digit "six": (a) two segments, (b) terminal point and the selected fourth point,(c) a terminal point found in the upper area, (d) search the area below the terminal point, (e) search the area above the terminal point, (f) search the area to the left of the terminal point, (g) search the area to the right of the terminal point.

2.4 Repairing Deformed Samples

It have been found that some deformed digit samples produce different Fourier descriptors features than the expected ones, and these samples correspond to digits "five", "seven", and "nine". Some samples of digit "five" had an extra line as shown in Fig. (4-a) or had an open circle as shown in Fig. (4-b), while some samples from digit "seven" had an extra line in the lower part of the digit as shown in Fig.(4-c) which might be selected as the start pixel. While, some samples of digit "seven" may have the left section longer than the right one, and in such case its left upper terminal may be selected as the start pixel. Also, some samples of digit "nine" may have extra extension as shown in Fig.(4-d), or unconnected terminal pixel as shown in Fig.(4-e).

To solve this problem, a morphological based criterion was added as a pre-classification stage. It depends on the distance between the digit contour parts located at right hand sides of the digit body. These distances are computed to three vertical levels of the digit (up, mid, low), shown in Fig.(5).It had been found that samples of digits "one", "two", "four", and "six" have no gaps and therefore zero distance values are computed. Where, the remaining digits (i.e., "three", "five", "seven", "eight", and "nine") have distance values. In order to distinguish among them, it is found that all samples of digit "five" have a middle distance larger than its both upper and lower distances as shown in Fig.(5-a), while all samples of digit "seven" have an upper distance larger than its middle distance, and its middle distance is larger than its lower distance as shown in Fig.(5-b).Where, the samples of digit "eight" have upper distance less than its middle distance, and its middle distance is less than its lower distance as shown in Fig.(5-c).

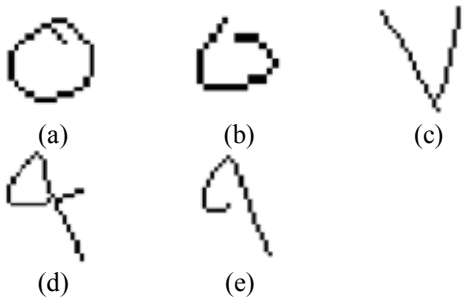


Fig.(4) Deformed samples: (a) digit "five" sample with extra line and one terminal pixel, (b) digit "five" sample with two terminal pixels and incomplete open circle, (c) digit "seven" sample with extra line, (d) digit "nine" sample with extra line, (e) digit "nine" sample with open circle in the upper part.

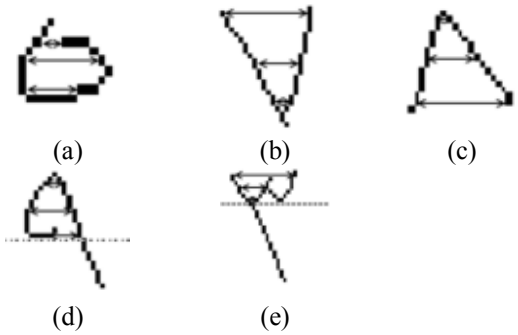


Fig.(5) Locations of upper, middle, and lower distance: (a) digit "five" distance locations, (b) digit "seven" distance locations, (c) digit "eight" distance locations, (d) digit "nine" distance locations, (e) digit "three" distance locations.

Both digit "three" and "nine" have non zero distances at the upper part of the digit segment, above the horizontal center of the digit segment as shown in Fig.(5-d) and Fig. (5-e), this remark is utilized to distinguish them from digits "five", "seven", and "eight". In order to distinguish between digit "three" and digit "nine", the number of terminal pixels were used, all samples of digit "three" have at least 3 terminal pixels and usually have 4 terminal pixels, while all samples of digit "nine" have only 1 or 2 terminal pixels.

After distinguishing one of the digits set member {5, 7, 8, 9} digits; if the digit is "five" and have only one terminal pixel then it may have an extra line and the extra line eraser algorithm is used to remove that line while if it has two terminal pixels then the Bresenham algorithm is used to connect the two terminal and close the circle. In the case of digit "seven", if it have three terminal pixels then the lowest terminal pixel is the start point of an extra line and will be passed to algorithm extra line eraser. Finally in the case of digit "nine", if it have two terminal pixels then the upper terminal pixel can either represent a start point of an extra line or of a missing line and in both cases the algorithm of repairing digit nine will be used to have the right decision. Fig.(6) shows samples of the repaired digits (shown in Fig. (4)).

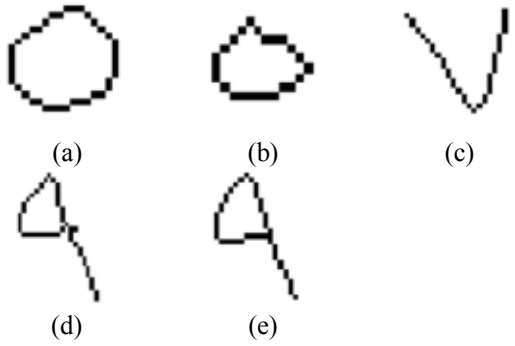


Fig.(6) Repaired deformed samples: (a) digit "five" after removing the extra line, (b) digit "five" after connecting the two terminal pixels and closing the circle, (c) digit "seven" after removing the extra line in the lower part, (d) digit "nine" after removing the extra line, (e) digit "nine" after connecting the upper terminal pixel and closing the head of digit "nine".

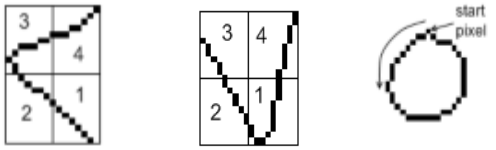
```
Algorithm: Extra line eraser
Input: x and y coordinates of a terminal pixel.
Output: delete the extra line pixels starting from
the terminal pixel.
x = terminal x : y = terminal y
flag = false
While flag is false
 co = 0
 For i = -1 to 1 do
  For k = -1 to 1 do
   If i<> 0 and k <> 0 and image[x+i, y+k] <> 255
   then
     nx = x+i : ny = y+k
     co = co + 1
   End If
  End For
 End For
 If co = 1 then
    image[x, y] = 255
    delete the pixel from the segment
    x = nx : y = ny
 End If
 Else flag = true
Repeat
```

```
Algorithm: Repairing digit nine
Input: x and y coordinates of the upper terminal
pixel.
Output: either delete the extra line pixels or insert
the missing line pixels.
Set x = given x + 1
Set y = given y
Set flag = false
While x <= segment maximum x and flag = false
do
  If image[x, y] <> 255 then
    flag = true
  Else
    x = x + 1
  End If
Repeat
If flag = false then
  delete the extra line by calling extra line eraser
  algorithm with the given x and y
Else
  For i = given x to current x do
    image[i, y] = 200
    add this pixel to the segment
  End for
End If
```

2.5 Allocation of Start Pixel

In order to achieve high recognition rates, the Fourier Descriptors requires to trace pixels in the same direction for each digit segment, either clock-wise or counter clock-wise, this direction depends on which terminal pixel is selected as the start pixel. For example, digit

two in Fig.(7-a) have two candidate start pixels, an upper terminal pixel and a lower terminal pixel; choosing each one will lead to different set of Fourier features values. In order to solve this problem, the digit segment space was divided into 4 quarters as shown in Fig.(7-a), the start pixel can be found by ordering the terminals pixels according to their occurrences in the 4 quarters starting with quarter 1 and ending in quarter 4. Then the first found terminal is adopted as the start pixel. In Fig.(7-a) the start pixel will be the terminal pixel found in quarter number 1, which means that all samples of digit "two" will have a start pixel in quarter 1, while in Fig. (7-b) the start pixel will be the terminal in quarter number 3 for all samples of digit "seven". As for digit "five" which have no terminal pixels, the pixel with the highest y coordinate will be selected as the start pixel and pixels are traced counter clock-wise as depicted in Fig.(7-c).

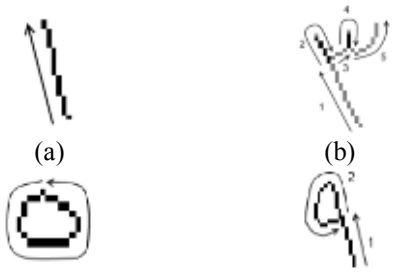


(a) (b) (c)

Fig.(7) Selection of start pixel: (a) digit "two" sample with the 4 quarters, (b) digit "seven" sample with the 4 quarters, (c) digit "five" with the pixel having the highest y coordinate as the start pixel and the left arrow showing the direction in which the pixels are collected.

2.6 Building Chain Codes

A novel algorithm was developed to create the chain code for any thinned digit segment. It is based on tracking the segment pixels sequentially. Many problems have been handled while developing this algorithm; like when the track reaches a cross point (T-type node) which has many leaves (as shown in Fig.(8-b), the digit "three" sample contains 2 cross points with three different paths for each); the algorithm must choose which path to go first and must return back to the cross point to follow a different path. To solve this problem the developed chain code is designed to classify the paths either as a terminated path (as shown in Fig.(8-a)), forked path (as shown in Fig.(8-b)), or circular path (as shown in Fig.(8-c) and Fig.(8-d)). When the tracing reaches a cross point of many paths, it analyzes each path to know its type and the number of pixels it contains. Then, it sorts the traced paths starting first with the circular paths, then the terminated paths, and the forked paths at last. The circular and terminated paths are sorted according to the number of pixels the path contains, forked paths are sorted according to the number of pixels the path contain to the cross point of other paths.



(a) (b)

(c) (d)

Fig.(8) Chain-Code algorithm and paths types: (a) digit "one" sample has a terminated path, (b) digit "three" sample has 2 cross points and the black pixels represents the paths selected first and recorded twice while returning back, (c) digit "five" sample has a circular path.

Algorithm: Chain-Code
Input: x and y coordinates of the start pixel.
Output: arrange the pixels as a sequential chain code.

1- Start with the x and y of the given start pixel.
2- Trace pixels and insert them in the segment array until reaching a cross point of multiple paths or the end of the segment.
3- If multiple paths found then for each path do step 4, and if end of segment is reached go to step 11.
4- Trace the path pixel by pixel until reaching a cross point or end of segment recording the number of pixels found and the type of the path.
5- Insert any found circular paths.
6- Sort paths with terminated paths, then forked paths.
7- Sort terminated paths with the path containing the minimum number of pixels first.
8- Sort forked paths with the path containing the minimum number of pixels from the start pixel to the forked point first.
9- For each path of the sorted paths except the last one do steps 1 to 8 and record the pixels found in reverse order (must return back to the cross point to continue with another path).
10- Record the last path pixels.
11- End.

2.7 Normalizing Pixels Coordinates

After the establishment of the chain code for each digit segment, then the coordinates of all collected pixels are normalized to be invariant to translation and scaling. Equations (2 - 4) have been used for mapping x and y coordinates to their corresponding normalized values.

$$x_n = \frac{x - x_{min}}{D} \quad \text{.....(2)}$$

$$y_n = \frac{y - y_{min}}{D} \quad \text{.....(3)}$$

$$D = \max\{x_{max} - x_{min}, y_{max} - y_{min}\} \quad \text{.....(4)}$$

Where n is the number of pixels of the traced digit segment, x_n and y_n are the normalized values, x and y are the old pixel coordinates, x_{min} is the smallest registered x coordinate value, x_{max} is the largest registered x coordinate value, y_{min} is the smallest registered y coordinate value, y_{max} is the largest registered y coordinate value.

2.8 Extraction Features

Fourier descriptors are used to describe the objects shape in terms of its spatial frequency content. In this research they used as the main recognition criterion to produce a set (i.e., feature vector) consist of 17 features. Equations (5 - 8) are used to determine the 17 features. For each digit segment its created chain-code consists of the pixels normalized coordinates pairs (x_n, y_n) traversed in counter-clockwise direction.

$$F(n) = F_R(n) + j\,F_I(n) \quad \text{.....(5)}$$

$$F_R(n) = \sum_{i=0}^{M} x(i)\cos\left(\frac{2\pi in}{M}\right) + y(i)\sin\left(\frac{2\pi in}{M}\right) \quad \text{...(6)}$$

$$F_I(n) = \sum_{i=0}^{M} -y(i)\cos\left(\frac{2\pi in}{M}\right) + x(i)\sin\left(\frac{2\pi in}{M}\right) \quad \text{...(7)}$$

$$|F(n)| = \sqrt[2]{F_R(n)^2 + F_I(n)^2} \quad \text{.....(8)}$$

Where, n is the number of features (n=1, 2 … 16), F(n) is the Fourier descriptor or feature, $F_R(n)$ is the real part of Fourier descriptor, $F_I(n)$ is the imaginary part of Fourier descriptor, M is the number of pixels in a digit segment, x(i) and y(i) are the x and y coordinates of pixel number i in a digit segment.

The determined features values were normalized, to be in the range between 0 and 1, using equation (9).

$$NFD = \begin{cases} 0 \; if \; FD \leq min_c \\ \frac{FD - min_c}{max_c - min_c} \; if \; FD > min_c \; and \; FD < max_c \\ 1 \; if \; FD \geq max_c \end{cases} \quad \text{.....(9)}$$

Where min_c and max_c are computed by equations (10) and (11) respectively

$$min_c = mean_c - 2.5 * std_c \quad \text{.....(10)}$$

$$max_c = mean_c + 2.5 * std_c \quad \text{.....(11)}$$

Where $mean_c$ is the mean of all collected feature values belong to the same digit class, std_c is the corresponding standard deviation value. After determination of Fourier descriptor values, and before determining their normalized value, each feature value, F(i) for i=1 … 16, is divided by zero-order descriptor value, i.e. F(0). This division will let the new 16 features become scale invariant. Rotation invariant of the features is achieved by ignoring the phase information and taking only the magnitude values of the Fourier features. It is found that the Fourier descriptors produced for digit "seven" and digit "eight" are similar, and to handle this problem the second classification stage classifier is used which is based on the pre-determined morphological criterion given in section (2.4).

3. Results

In this research work, a form have been designed with a yellow boarder as shown in Fig.(9-a) to acquire digits samples. Five forms have been filled out by five persons and scanned using a flatbed scanner at a resolution of 100 dbi and stored as a colored images then each form was separated into 9 image stripes containing 10 different samples of the same digit as shown in Fig.(9-b) and (9-c). All images stripes containing the same digit were moved to a separate folder which results in a 9 folders each containing samples of a specific digit. it was noticed that the boarder pixels on all sprites have red component greater than 220 and green component greater than 210 and blue component between 161 and 209 and these values are differ from the color components values of the background and the digits, so all boarder pixels components were changed to 255 which eliminates the boarders.

Digit zero wasn't part of the input data because it is easily recognized as a small set of pixels ordered in the shape of a filled circle.

The system is tested using 450 samples collected from 5 students, first the system was tested depending only on the Fourier descriptors features and a best recognition ratio was acquired from a combination of 5 Fourier descriptors FD1, FD2, FD6, FD7, and FD15, with a recognition ratio of %89.6 (as shown in Table (1)). When the morphological based classifier was used as an additional classifier (beside to the Fourier descriptors) the recognition ratio increased to %98 using a combination of the 4 Fourier descriptors FD1, FD2, FD10, and FD14 as shown in Table (2).

(a) (b) (c)

Fig.(9) Input form: (a) Input form sample , (b) input form divided into stripes, (c) image of one stripe.

Table (1)
Tested result of using the Fourier descriptors combinations as the only classification method.

Number of Fourier Descriptors used	*Best Fourier Descriptors Combinations Found*	*Number of recognized samples*	*Recognition Ratio*
1	1	251	55.8
2	1, 4	341	75.8
3	1, 2, 6	393	87.3
4	1, 2, 6, 15	399	88.7
5	1, 2, 6, 7, 15	403	89.6
6	1, 2, 4, 5, 6, 15	403	89.6
7	1, 2, 3, 4, 5, 6, 14	401	89.1

Table (2)
Tested result of using the Fourier descriptors combinations as the main classifier and the structured based classifier as the second classifier.

Number of Fourier Descriptors used	*Best Fourier Descriptors Combinations Found*	*Number of recognized samples*	*Recognition Ratio*
1	1	411	91.3
2	1, 2	434	96.4
3	1, 2, 5	439	97.6
4	1, 2, 10, 14	441	98
5	1, 2, 3, 6, 11	439	97.6
6	1, 2, 3, 4, 9, 14	438	97.3
7	1, 2, 3, 4, 5, 6, 14	436	96.9

3. Conclusions

The test results indicated that the Fourier descriptors alone is not enough for recognizing Arabic digits; so a morphological based criterion is needs to improve the recognition accuracy. The digits "seven" and "eight" have very similar former set of values, and to solve the problem of discrimination between these two digits as additional structure based classification step based on morphological criterion were used.

Reference

[1] Rumiana Krasteva, Ani Boneva, Ditchko Butchvarov, and Veselin Geortchev, 2001, "Basic components in Optical Character Recognition Systems. Experimentally Analysis on Old Bulgarian Character Recognition", *Academic Open Internet Journal*, Vol. 5, 2001.

[2] Habib Mir Mohammad Hosseini and Abdesselam Bouzerdoum, "A Combined Method for Persian and Arabic Handwritten Digit Recognition", Australian New Zealand Conf. on Intelligent Information Systems, 1996.

[3] A. Harifi and A. Aghagolzadeh, "A New Pattern for Handwritten Persian/Arabic Digit Recognition", proceedings of world academy of science, engineering and technology, Vol. 3, pp. 249-252, January 2005.

[4] Ahmad T. Al-Taani, "An Efficient Feature Extraction Algorithm for the Recognition of Handwritten Arabic Digits", International Journal of Engineering and Mathematical Sciences, 2006, pp. 107-111.

[5] Nibaran Das, AyatullahFarukMollah, SudipSaha, Syed SahidulHaque, "Handwritten Arabic Numeral Recognition using a Multi Layer Perceptron", National Conference on Recent Trends in Information Systems, 2006.

[6] Sabri A. Mahmoud, Marwan H. Abu-Amara, "Recognition of Handwritten Arabic (Indian) Numerals using Radon-Fourier-based Features", recent advances in signal processing, robotics and automation, pp. 158-163.

[7] Amir Mowlaei, KarimFaez, "Recognition of Isolated Handwritten Persian/Arabic Characters and Numerals using Support Vector Machines", IEEE XIII Workshop on Neural Networks for Signal Processing recognition, 2003, pp. 547-554.

[8] Sabri A. Mahmoud and Sameh M. Awaida, "Recognition of off-line Handwritten Arabic (Indian) Numerals using Multi-Scale Features and Support Vector Machines vs. Hidden Markov Models", The Arabian Journal for Science and Engineering, Vol. 34, No. 2B, 2009, pp. 429-444.

[9] YousefAjamiAlotaibi, "Comparative Study of ANN and HMM to Arabic Digits Recognition Systems", JKAU: Eng. Sci., Vol. 19, No. 1, 2008, pp: 43-60.

[10] Amir Mowlaei, KarimFaez and Abolfazl T. Haghighat, "Feature Extraction with Wavelet Transform for Recognition of Isolated Handwritten Farsi/Arabic Characters and Numerals", DSP, 2002, pp. 923-926.

الخلاصة

في هذا البحث، تم اقتراح طريقة بسيطة، سريعة، ودقيقة لتمييز الارقام العربية (الهندية) باستخدام واصفات فورير كمصنف اساسي، ولتحسين الكفاءة تم اضافة مصنف يعتمد البنية كمصنف تكميلي. بعد اجراء الاختبارات على 450 عينة تم جمعها من قبل 5 طلاب، بينت النتائج ان نسبة التمييز هي 89.6% عند استخدام 5 واصفات فورير كمصنف اساسي، و ارتفعت نسبة التمييز الى 98% عند استخدام 4 واصفات فورير والمصنف المعتمد على البنية.

[11] F. A. Al-Omari and O. Al-Jarrah, "Handwritten Indian Numerals Recognition System Using Probabilistic Neural Networks", Advanced Engineering Informatics, 2004, pp. 9–16.

[12] Shunji Mori, Member, IEEE, Ching Y. Suen, Fellow, IEEE, and Kazuhiko Yamamoto, Member, IEEE, "Historical Review of OCR Research and Development", Proceedings of the IEEE, Vol. 80, No. 7, July 1992, pp. 1029-1058.

[13] M. S. Khorsheed, "Off-Line Arabic Character Recognition – A Review", Pattern Analysis & Applications, 2002, pp. 31–45.

[14] Liana M. Lorigo, Member, IEEE Computer Society, and VenuGovindaraju, Member, IEEE Computer Society, "Offline Arabic Handwriting Recognition: A Survey", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 28, No. 5, May 2006, pp. 712-724.

[15] Pervez Ahmed and Yousef Al-Ohali, "Arabic Character Recognition: Progress and Challenges", J. King Saud Univ., Vol. 12, pp. 85-116, 2000.

[16] Qivind Due Trier, Anil K. Jain and TorfinnTaxt, "Feature Extraction Methods for Character Recognition – A Survey", Pattern Recognition, Vol. 29, No. 4, pp. 641-662, 1996.

[17] John C. Russ, 2007 Fifth Edition, "The Image Processing Handbook", CRC.

[18] Louisa Lam, Seong-WhanLee, Member, IEEE, and Ching Y. Suen, Fellow, IEEE, "Thinning Methodologies–A Comprehensive Survey", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14, No. 9, September 1992.

[19] Robert Whitrow, 2008, "OpenGL Graphics Through Applications", Springer.

[20] G. G. Rajput, RajeswariHorakeri, and SidramappaChandrakant, "Printed and Handwritten Kannada Numeral Recognition Using Crack Codes and Fourier Descriptors Plate", IJCA special issue on "Recent Trends in Image Processing and Pattern Recognition", RTIPPR, 2010.