Arabic Character Extraction and Recognition using Traversing Approach

Abdul Khader Saudagar and Habeeb Mohammed
College of Computer and Information Sciences, Al Imam Mohammad Ibn Saud Islamic University, Saudi Arabia

Abstract: *The intention behind this research is to present an original work undertaken for Arabic character extraction and recognition for attaining higher percentage of recognition rate. Copious techniques for character, text extraction were proposed in earlier decades, but very few of them shed light on Arabic character set. From literature survey, it was found that 100% recognition rate is not attained by earlier proposed implementations. The proposed technique is novel and is based on traversing of the characters in a given text and marking their directions viz. North-South (NS), East-West (EW), North East-South West (NE-SW), North West-South East (NW-SE) etc., in an array and comparing them with the pre-defined codes of every character in the dataset. The experiments were conducted on Arabic news videos, documents taken from Arabic Printed Text Image (APTI) database and the results achieved from this research are very promising with a recognition rate of 98.1%. The proposed algorithm in this research work can replace the existing algorithms used in present Arabic Optical Character Recognition (AOCR) systems.*

Keywords: *Accuracy, arabic optical character recognition and text extraction.*

Received March 14, 2015; accepted August 16, 2015

1. Introduction

With the swift boom of television channels, media, WWW services, information turn out to be progressively obtainable and easily available. The preservation of records is done easier with computerization and hence results in easier documents access. On the other hand, when the number of documents become vital the digitalization process is not sufficient to guarantee a resourceful access. Indeed, there is a necessity to extract text which come into sight in video, which frequently mirror prospects semantic content which has received little interest in the past.

Text extraction from Arabic videos is a challenging problem when compared with scanned image. The splitting of the video into frames in this research work is carried out using Java programming, where the sample video file is split into frames using the java library with milliseconds of time span between them [21, 22]. The time between the frames is provided by the user to split the video into image frames. A technique for text extraction and recognition which uses pixel comparison with pre-defined dataset is presented in [20]. In this technique the sample images obtained from splitting the news video containing Arabic text is cut into small frames by altering the interval and compared the resulted frames for matching equal number of pixels with the frames in the dataset. Text detection approach based on inherent characteristics is proposed in [16] for Farsi characters. A robust approach for Arabic news video sequence is explained in [10] for text extraction and recognition process. The proposed system in [17] uses Smallest Univalue Segment Assimilating Nucleus (SUSAN) contour based algorithm with morphological function like dilation for eliminating unwanted non-text regions in images during text extraction process. A morphological dilation technique presented in [13] achieved a precision of 96.3 % in the process of text line extraction of Arabic script. A morphological analysis based temporal entity technique with finite state transducer is presented for Arabic language [27]. This technique achieves 84.2% precision. A two phase linguistic root extraction technique is proposed in [4] which involves removal of prefixes, suffixes and infixes depending on their morphological pattern.

Algorithms designed for extracting text from a video using java libraries and classes begins with removing the disturbances like superimposed lines over the text, discontinuity removal, dot removal and for localization, segmentation, recognition using neural network pattern matching technique is proposed in [9]. A recognition systems [7, 11] based on segmentation for handwritten Arabic words is developed where the segmented characters are classified based on their statistical, discrete and structural features. Neural networks are employed to calculate weights for all statistical features. A novel segmentation algorithm using “Naskh” font is proposed in [8] to extract the feature points from residual regions of word image and is successful in obtaining 86% of results. Feature vectors are calculated on each character and compared with templates of Arabic alphabet in terms of variations. In this work 93.5% recognition rate is achieved [3]. For text detection problem, a new unsupervised text detection algorithm to localize candidate parts by extracting closed boundaries and initialize the links by connecting two neighboring candidate parts based on the spatial relationship of characters is presented in [28]. An algorithm to detect/localize and segment Arabic static artificial texts embedded in video which consists of 3 steps firstly, detecting predetermined region of interest followed by applying some filtering rules, secondly temporal information are exploited in order to reinforce the recall and precision rates and at last thresholding-based method is used for separating text pixels from background pixels and produce a binary text image is proposed in [2].

A color histogram technique [25] which minimizing the numbers of video frames is proposed in order to detect video text regions which contain the score and player information in a sports video. Edge and color features are combined and verified by wavelet histogram for detecting texts in normal scene images is explained in [6]. Corner metric and Laplacian filtering techniques are used to detect the text appearing in video [12] independent of each other and the binarization of the localized text is done by the seed pixel determination of the text results for an efficient detection and localization. Another novel framework [19] by generating morphological binary map for calculating difference between the final image and the initial image of the video scene to detect and extract the text is proposed. Various hybrid approaches to detect and localize texts in innate scene images are proposed in [15, 18]. A Hidden Markov Model (HMM) model [26] proposed using sliding windows to extract the robust features in handwritten images. This model uses embedded training to locate the best word and the recognition rate is estimated to be 84.09%. A hybrid approach consists of linguistic filter which uses parts of speech tagger, chain of patterns in order to take out candidate Arabic Multi Word Term's and statistical filter for integrating the contextual information is proposed in [14]. A combination of two algorithms, run length smoothing and connected component labeling using Support Vector Machine (SVM) is used for the text and non-text classification. The results are based on the Anding and Oring operations on certain conditions are presented in [5]. To some extent this approach handles the intricacy of Arabic language fonts and characteristics. Various techniques which are used in extraction process of online recognition systems for Arabic handwriting scripts are explained with their strengths and weakness in [1]. Despite of lots of research undergone in archiving the text embedded in digital video, the problem of extraction of text that are of interest with maximum accuracy is still a limitation and, very few of them have handled Arabic script.

This paper is organized as follows. Section 2 briefly introduces the Arabic character set features and describes the approach used in the novel text extraction and recognition algorithm. Section 3 contains the proposed algorithm followed by analysis in section 4. Section 5 gives and discusses experimental results obtained by proposed algorithm. Conclusions are provided in section 6.

2. The Research Method

Arabic language is the widely spoken language in many countries.

Table 1. (a) Arabic character set (b) Arabic character set for proposed research work.

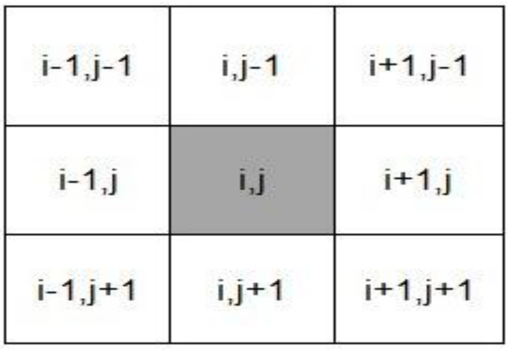
No. (a)	S	T	M	I
1	ا	ـا		
2	ب	ـب	ـبـ	بـ
3	ت	ـت	ـتـ	تـ
4	ث	ـث	ـثـ	ثـ
5	ج	ـج	ـجـ	جـ
6	ح	ـح	ـحـ	حـ
7	خ	ـخ	ـخـ	خـ
8	د	ـد		
9	ذ	ـذ		
10	ر	ـر		
11	ز	ـز		
12	س	ـس	ـسـ	سـ
13	ش	ـش	ـشـ	شـ
14	ص	ـص	ـصـ	صـ
15	ض	ـض	ـضـ	ضـ
16	ط	ـط	ـطـ	طـ
17	ظ	ـظ	ـظـ	ظـ
18	ع	ـع	ـعـ	عـ
19	غ	ـغ	ـغـ	غـ
20	ف	ـف	ـفـ	فـ
21	ق	ـق	ـقـ	قـ
22	ك	ـك	ـكـ	كـ
23	ل	ـل	ـلـ	لـ
24	م	ـم	ـمـ	مـ
25	ن	ـن	ـنـ	نـ
26	ه	ـه	ـهـ	هـ
27	و	ـو		
28	ي	ـي	ـيـ	يـ

No. (b)	S	T	M	I
1	ا	ـا		
2	ب			بـ
3	ت			تـ
4	ث			ثـ
5	ج			جـ
6	ح			حـ
7	خ			خـ
8	د			
9	ذ			
10	ر			
11	ز			
12	س			سـ
13	ش			شـ
14	ص			صـ
15	ض			ضـ
16	ط			
17	ظ			
18	ع	ـع	ـعـ	عـ
19	غ	ـغ	ـغـ	غـ
20	ف		ـفـ	فـ
21	ق		ـقـ	قـ
22	ك			كـ
23	ل			لـ
24	م			مـ
25	ن			نـ
26	ه	ـه	ـهـ	هـ
27	و			
28	ي	ـي		يـ

All most all the documents are in the Arabic language as it is the official language of these countries. Indeed there is a need to archive the available information for later use. Text can be extracted from these news videos and documents, pile up in text files, indexed and can be searched for necessary information in forthcoming years. Arabic language has a typical character set with twenty-eight characters, having four forms for each character such as Standalone (S), Initial (I), Medial (M) and Terminal

(T) as shown in Table 1-a. The empty spaces in Table 1-a exemplify the unavailable character forms. As there is similarity in the character set forms, few more forms are ignored from Table 1-a while traversing and marking the directions of the character. Table 1-b represents the Arabic character set for the research work after ignoring the alike character forms.

Preliminary research work is carried on images obtained from splitting of Arabic news videos with black color text on white background with Arial font and size 24, now called as input image. This research work uses the Zhang-Suen thinning algorithm for thinning the text on an open CV-based java platform and the obtained results of thinning the text are superior when compared to results obtained from other implementations [23]. The proposed traversing extraction algorithm is applied on the thinned image obtained from applying Zhang-Suen thinning algorithm on input image. From Figure 1, the current pixel position is i,j. From this current position the algorithm searches for black pixel in its neighbourhood, if it finds one black pixel in i,j+1 it move in that direction and makes that coordinate as the current pixel position. Upon occurrence of two black pixels, the algorithm uses the direction which has highest priority as shown in Table 2 and saves the other coordinate to be traversed in the following loop.



i-1,j-1	i,j-1	i+1,j-1
i-1,j	i,j	i+1,j
i-1,j+1	i,j+1	i+1,j+1

Figure 1. Pixel coordinates.

Table 2. Priority in traversing.

Priority	From	To
1 (High)	i , j	i,j+1
2	i , j	i-1,j
3	i , j	i-1,j-1
4	i , j	i-1,j+1
5	i , j	i+1,j
6	i , j	i+1,j+1
7	i , j	i+1,j-1
8 (Low)	i , j	i,j-1

3. The Proposed Algorithms

The following algorithms are used in the text extraction and recognition process.

Algorithm 1: Individual Standalone Character Recognition

```
Input: Thinned image containing Arabic characters / strings
Output: Code words for every character
/* copyarray[][]: output traversed matrix */
/* anarray[]:variable for saving coordinate positions */
begin
while x2 < image_width and y2 < image_height do
                copyarray [x2][y2] ← 0;
    x ← image_width-1
    y ← 0
    anarray[0] ← 0
      while  x>=0 and y<image_height-1 do
        if RGB(x,y) <25 and anarray[0]==0 then
           i ← x, j ← y
          anarray[0] ← x;  anarray[1] ← y
          break
while  i>0  do
 repeat
if RGB(i,j+1) <25 and copyarray[i][j+1]!=1 then
               m=i
               n=j+1
               copyarray[m][n]=1
               ns=ns+1
elseif RGB(i-1,j)<25 and copyarray[i-1][j]!=1
                m=i-1
                n=j
               copyarray[m][n]=1
               ew=ew+1
elseif RGB(i-1,j-1)<25 and copyarray[i-1][j-1]!=1
                m=i-1
                n=j-1
               copyarray[m][n]=1
               senw=senw+1
elseif RGB(i-1,j+1)<25  and copyarray[i-1][j+1]!=1
               m=i-1
                n=j+1
               copyarray[m][n]=1
               nesw=nesw+1
elseif RGB(i+1,j)<25 and copyarray[i+1][j]!=1
               m=i+1
               n=j
               copyarray[m][n]=1
               we=we+1
elseif RGB(i+1,j+1)<25 and copyarray[i+1][j+1]!=1
                m=i+1
                n=j+1
                copyarray[m][n]=1
               nwse=nwse+1
elseif RGB(i+1,j-1)<25 and copyarray[i+1][j-1]!=1
               m=i+1
               n=j-1
               copyarray[m][n]=1
               swne=swne+1
elseif RGB(i,j-1)<25 and copyarray[i][j-1]!=1
               m=i
               n=j-1
               copyarray[m][n]=1
               sn=sn+1;
end if
          i=m
         j=n
until (RGB(i,j+1)<25 OR RGB(i-1,j)<25  OR RGB(i-1,j-1)<25
OR RGB(i-1,j+1)<25) == true

return for saved pixel position which are not traced and if
exists repeat the process
   end.
```

Algorithm 2: Connected Second Character Start Position

Input: Thinned image containing Arabic characters/strings
Output: Start position of second character

```
/* i,j : current pixel position */
begin
while  i>=0 and j<image_height-1 do
repeat
begin
        if RGB (i-1,j) <25 then
                if RGB (i,j-1) <25
                        if RGB (i+1,j) <25
                                charsplit←true;

        secondcharacter_startposition←i,j
                        end if
                end if
        end if
end
AND
begin
        if RGB (i,j+1) <25 then
                if RGB (i,j-1) <25
                        if RGB (i+1,j) <25
                                charsplit←true;

        secondcharacter_startposition←i,j
                        end if
                end if
        end if
end
until i <=10
end
```

Algorithm 3: Second string recognition in sentence

```
Input: Thinned image containing Arabic characters/strings
Output: Start position of the first character of second string
/* copyarray[][]: output traversed matrix */
/* i,j : current pixel position */
begin
while  i>=0 and j<image_height-1 do
repeat
  begin
        if ((RGB (i,j+1) >25) and (RGB (i-1,j+1) >25) and
(RGB (i-1,j) >25) and (RGB (i-1,j-1) >25)  and copyarray[i][j-
1]= =1) then
        secondstring←true;
        secondstring_startposition: i←i-2 and j←0
        end if
end
OR
begin
        if ((RGB (i,j-1) >25) and (RGB (i-1,j+1) >25) and
(RGB (i-1,j) >25) and (RGB (i-1,j-1) >25) and
copyarray[i][j+1]==1) then
        secondstring←true;
        secondstring_startposition:i←i-2 and j←0
        end if
end
OR
begin
        if ((RGB (i,j-1) >25) and (RGB (i-1,j+1) >25) and
(RGB (i-1,j) >25) and (RGB (i-1,j-1) >25) and (RGB (i,j+1)
>25) and copyarray[i+1][j]==1) then
        secondstring←true;
        secondstring_startposition: i←i-2 and j←0
        end if
end
until i <=10
end
```

Algorithm 4: Concatenation of code generated and recognition of character

```
Input: Thinned image containing Arabic characters/strings
Output: Recognized character
/* codearray[]: output code matrix of size p */
/* i,j : current pixel position */
begin
q←0
while  i>=0 and j<image_height-1 do
repeat
  begin
        codearray[q] ← generated code
        compare codearray with dataset

        if true
                return character recognized
        else
                q=q+1
        end if
  end
until i <=10
end
```

4. Analysis

Start searching the thinned image for the first occurrence of black pixel from (x-1,0) position where x is the width of the image, upon finding the black pixel start traversing the text in the direction of the occurrence of the black pixel as shown in Figure 2. If there is any discontinuity then repeat the process for the next occurrence of the black pixel. Consider the following Figure 3 for marking the directions in the array. The Arabic characters are traversing in the directions as shown in Figure 4 with the baseline passing through center of the character.

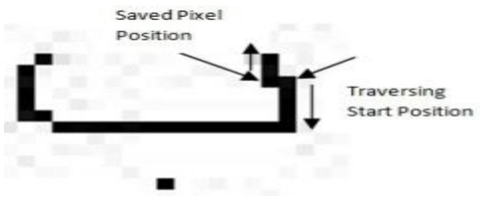


Figure 2. Traversing and saving position.

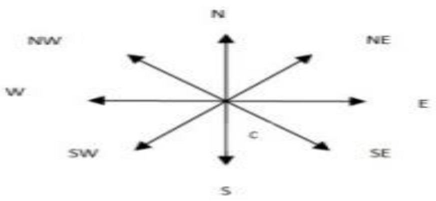


Figure 3. Direction codes.

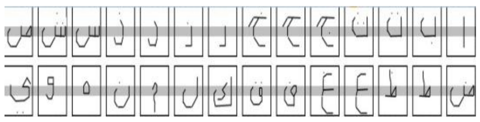


Figure 4. Arabic characters in traversing mode.

Count the pixels while traversing the character in the specific direction as shown in Figure 3 and upon reaching a certain limit in the one direction, mark the direction code as shown in Table 3 in the first position of array whose size is k. The limits of counter values are decided by the researcher. If there is one black pixel surrounded by white (discontinuous) pixels, check whether the dot is above, below or on the baseline. If the dot is above the base line, mark it as Upper Dot (UD). If it is below the baseline, mark it as Lower Dot (LD) otherwise Center Dot (CD) as a code in the next position of the array. At some coordinates the availability of number of black pixels are more than 3, that coordinate is the commencement of the second character or join with another character, at that point stop traversing the text and compare the codes in the array with pre-defined dataset.

Table 3. Description of character codes.

S. No	Character Code	Description
1	NS	NorthSouth
2	SN	SouthNorth
3	EW	EastWest
4	WE	WestEast
5	NWSE	NorthWestSouthEast
6	NESW	NorthEastSouthWest
7	SENW	SouthEastNorthWest
8	SWNE	SouthWestNorthEast
9	UD	UpperDot
10	CD	CenterDot
11	LD	LowerDot

Few character forms as shown in Table 4 cannot be traversed completely in a single loop and generate the necessary code. These characters need to be traversed in multiple loops and the code generated in each loop need to be concatenated using algorithm 4 to form the unique code for that character as shown in the given example.

code (code 1) [code 2] /* individual codes in different loops

{code code 1 code 2} /* unique code

If the code generated while traversing is NESW or UD or EW alone, then these codes are ignored and not considered in concatenation.

Table 4. Character forms require multiple loops for code generation.

Index Value	Standalone	Medial	Initial
5	ج		
12	س		سـ
13	ش		شـ
14			صـ
15			ضـ
16	ط		
17	ظ		
18	ع		
19	غ		
20		ـفـ	
21		ـقـ	
26		ـهـ	هـ

The character forms as shown in Table 5 generate more than one code due to variance in thinning algorithm. So all the codes generated by the character need to be included in the dataset for character recognition.

Table 5. Character forms generate more than one code.

Index Value	Standalone	Terminal	Medial	Initial
1	ا			
2	ب			بـ
3	ت			
5				جـ
6				حـ
7				خـ
8	د			
10	ر			
12				سـ
13	ش			
14				صـ
15				ضـ
18			ـعـ	عـ
21			ـقـ	
22	ك			
23				لـ
24				مـ
25	ن			
26		ـه		هـ
28				يـ

Upon traversing few characters as shown in Table 6 generate the similar code, so special codes are introduced in order to differentiate between them.

Table 6. Character forms with special codes.

Index Value	Standalone	Medial	Initial
8	د		
9	ذ		
12	س		سـ
18		ـعـ	
19		ـغـ	غـ
21		ـقـ	

Generating code for the characters which are connected with 'Alif' for example با as a terminal character is difficult task. So a single code is generated separated by '0' which indicates the first part of code is related with the specific character which is connected with 'Alif' and the code after '0' is for the 'Alif'. Few characters such as ع and غ generate similar codes where the accuracy in the recognition of the text descends. The pre-defined dataset direction codes of every character in all forms are as shown in Table 7.

Table 7. Character codes.

Index Value	Character	Character Name	Character Code
1	ا	Alif-Standalone	NS NS / NS NS NS / NS NS NS NESW
2	بـ	Baa-Initial	NS EW LD / NS EW NESW LD / NS EW SENW LD
2	ب	Baa-Standalone	NS EW EW SENW SWNE SN LD / NS EW EW SENW SWNE SN
3	ثـ	Thaa-Initial	NS EW UD UD UD
3	ث	Thaa-Standalone	NS EW EW SENW SWNE SN UD UD / NS EW EW SENW NESW SWNE SN SN UD UD
4	ٹـ	Saa-Initial	NS EW UD UD UD
4	ٹ	Saa-Standalone	NS EW EW SENW SWNE SN UD UD UD
5	جـ	Geem-Initial	NS EW EW SENW NESW LD / NS EW SENW NESW LD
5	ج	Geem-Standalone	EW NESW LD LD (NESW LD LD) {EW NESW LD LD NESW LD LD}
6	حـ	Haa-Initial	NS EW EW SENW NESW / NS EW SENW UD UD
6	ح	Haa-Standalone	EW NESW LD LD LD
7	خـ	Qaa-Initial	NS EW EW SENW NESW UD / NS EW SENW UD / NS EW SENW NESW UD
7	خ	Qaa-Standalone	EW NESW UD
8	د	Daal-Standalone	NS EW DA / NS EW NESW
9	ذ	Zaal-Standalone	NS EW UD ZA
10	ر	Raa-Standalone	NS NS EW SENW NESW / NS NS EW SENW NESW NWSE
11	ز	Zaa-Standalone	NS NS EW SENW NESW UD
12	سـ	Seen-Initial	NS (NS EW) [NS] {NS NS EW NS}/ NS NESW (NS EW GM)[NS EW] {NS NESW NS EW GM NS EW} / NS (NS EW GM) [NS EW]{NS NS EW GM NS EW} / NS NESW NWSE (NS EW GM) [NS EW GM] { NS NESW NWSE NS EW GM NS EW GM} NS NESW (NS EW GM)[NS EW GM] { NS NESW NS EW GM NS EW GM}
12	س	Seen-Standalone	NS (NS EW GM)[EW SENW NESW SWNE SN SN SN] {NS NS EW GM EW SENW NESW SWNE SN SN SN }
13	شـ	Sheen-Initial	NS UD (NS EW UD UD)[NS] {NS UD NS EW UD UD NS}
13	ش	Sheen-Standalone	NS UD (NESW SN)[NS NS EW SENW NESW SWNE SN SN] {NS UD NESW SN NS NS EW SENW NESW SWNE SN SN} / NS UD (NESW SN)[NS NS EW SENW NESW SWNE SN SN UD] {NS UD NESW SN NS NS EW SENW NESW SWNE SN SN UD}
14	صـ	Saad-Initial	NS EW EW EW NESW / NS EW (EW) [NS] { NS EW EW NS}
14	ص	Saad-Standalone	NS NS NS EW EW EW SENW NESW SWNE SN
15	ضـ	Zaad-Initial	NS EW EW EW NESW UD / NS EW (EW EW UD){ NS EW EW EW UD}
15	ض	Zaad-Standalone	NS NS NS EW EW EW SENW NESW SWNE SN UD
16	ط	Thoe-Standalone	NS EW (NS NS NS EW NESW NWSE) {NS EW NS NS NS EW NESW NWSE }
17	ظ	Zoe-Standalone	NS EW UD (NS NS NS EW NESW NWSE) { NS EW UD NS NS NS EW NESW NWSE}
18	عـ	Eain-Initial	NS EW SENW NESW / NS EW SENW NESW LD LD
18	ـعـ	Eain-Medial	NS EW EM/ NS EW SENW
18	ـع	Eain-Terminal	NS NS EW SENW NESW WE NWSE
18	ع	Eain-Standalone	NS NS EW NESW WE NWSE UD UD (NS) { NS NS EW NESW WE NWSE UD UD NS}
19	غـ	Gain-Initial	NS EW SENW NESW UD GS
19	ـغـ	Gain-Medial	NS EW UD GM
19	ـغ	Gain-Terminal	NS NS EW SENW NESW WE NWSE UD
19	غ	Gain-Standalone	NS NS EW NESW WE NWSE UD UD (NS) { NS NS EW NESW WE NWSE UD UD NS}
20	فـ	Faa-Initial	NS EW NWSE SN UD
20	ـفـ	Faa-Medial	EW UD (NS EW UD) { EW UD NS EW UD }
20	ف	Faa-Standalone	NS NS EW EW EW SENW NESW NWSE SWNE SN UD
21	قـ	Qaaf-Initial	NS EW NWSE SN UD UD
21	ـقـ	Qaaf-Medial	EW UD (NS EW UD UD) { EW UD NS EW UD UD } / NS EW UD UD QM
21	ق	Qaaf-Standalone	NS NS EW EW SENW NESW SWNE SN SN UD UD
22	كـ	Kaaf-Initial	NS NS EW EW NESW NWSE
22	ك	Kaaf-Standalone	NS NS EW EW SENW SWNE SN SN UD / NS NS EW EW SENW SWNE SN SN UD UD
23	لـ	Laam-Initial	NS NS EW SENW SN SN / NS EW NESW UD UD UD / NS NS EW UD UD UD
23	ل	Laam-Standalone	NS NS EW SENW NESW SWNE SN SN
24	مـ	Meem-Initial	EW SENW NESW / EW EW SENW NESW
24	م	Meem-Standalone	NS NS EW NESW NWSE
25	نـ	Noon-Initial	NS EW UD
25	ن	Noon-Standalone	NS NS EW SENW NESW SWNE SN UD / EW SENW NESW SWNE SN UD / EW SENW NESW SWNE SN SN SN UD
26	هـ	Ha-Initial	NS NS EW EW SENW NESW / NS EW NESW (NESW) { NS EW NESW NESW}
26	ـهـ	Ha-Medial	EW UD (NS NS SENW NESW SWNE SN) { EW UD NS NS SENW NESW SWNE SN}
26	ـه	Ha-Terminal	NS EW SWNE UD UD / NS NESW SWNE UD UD
26	ه	Ha-Standalone	NS EW SENW SWNE SN
27	و	Waw-Standalone	NS NS NS EW SENW NESW
28	يـ	Yaa-Initial	NS EW LD LD / NS EW NESW LD LD
28	ـي	Yaa-Terminal	NS EW EW SENW NESW WE SWNE LD LD
P28	ي	Yaa-Standalone	NS EW EW SENW NESW NWSE SWNE SN LD LD
	لا	LaamAlif	NS NS NS NESW (SENW NESW UD) { NS NS NS NESW SENW NESW UD }
	لإ	LaamAlif1	NS NS NESW (SENW NESW UD)[LD LD LAH] { NS NS NESW SENW NESW UD LD LD LAH}
	ﻼ	LaamAlif2	NS NS NS EW SENW NESW LA / NS EW SENW NESW SWNE SN SN UD UD UD
		Ignore	NESW
		Ignore	UD
		Ignore	EW

Upon matching the code in the predefined dataset, with its corresponding index value the character is recognized. This process is repeated for all characters in the string. If the difference between the last traversed pixel position in the first string/word and the first black pixel found after the last black pixel of the first string/word is greater than 9 pixels in their x- axis coordinates indicates the commencement of the second string/word in a sentence denoted by $. Once all the characters are extracted and recognized, they are placed in an array as individual characters and concatenated with the help of the Arabic Unicode format, as shown in Figure 5, for suitable word formation and, therefore, sentence formation [24].

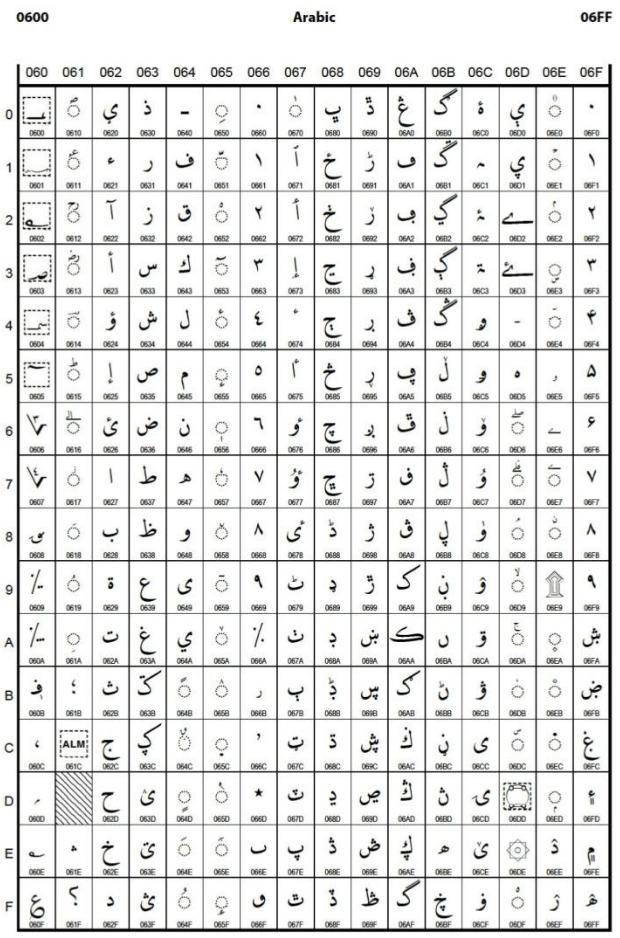

0600
Arabic
06FF

Figure 5. Arabic unicode chart.

5. Results

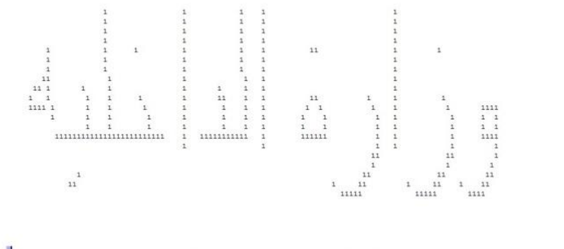
The implementation is tested on Arabic news video clips taken from Aljazeera television and 55 sample images taken from Arabic Printed Text Image database using Java 1.7 on Netbeans 8.0.1 IDE. The system configuration used for coding and testing is Intel i5 ×64 processor with 16 GB RAM and 512 GB hard disk. The accuracy of the experimental results is calculated using the formula as shown below.

$$precision = \frac{Number of\ true\ recognized\ strings}{Total\ number\ of\ strings} \tag{1}$$

Very promising results with a recognition rate of 98.1% are achieved by using this algorithm. A video sample of length 6 minutes and 22 seconds is taken from Aljazeera television containing scrolling text from left to right direction. Figure 6 shows the samples of output frames after splitting the video with a time-stamp of 10 sec. The time-stamp is decided based on the scrolling rate of the Arabic text in a video so that the scrolling text never disregard from the frames. Figure 7 shows the traversed string of the Arabic text in the video frame and code generated.

a)frame # 15. b)frame # 21. c)frame # 32.

Figure 6. Sample frames from video.



```
$
NSNSNSEWSENWNESWnullNSNSEWSENWNESWUDnullNSNSNSNESWnullNSNSEWSENWNESWnullEWSENWNESWSWNEUDUDnull$
NSNSNSNESWnullNSEWNESWUDUDUDnullNSEWDAnullNSNSNSNESWnullNESWnullNSEWSENWUDnullNSNSEWUDUDUDnullNSEWLDLDnullNSNESWSWNEUDUDnull
$
NSNSNSEWSENWNESW   NSNSEWSENWNESWUD   NSNSNSNESW   NSNSEWSENWNESW   EWSENWNESWSWNEUDUD   $
NSNSNSNESW   NSEWNESWUDUDUD   NSEWDA   NSNSNSNESW   NESW   NSEWSENWUD   NSNSEWUDUDUD   NSEWLDLD   NSNESWSWNEUDUD
BUILD SUCCESSFUL (total time: 3 seconds)
```

Figure 7. Traversed string and code generated.

Another test case of image containing Arabic text in Figure 8 and its corresponding code generated is as shown in Figure 9.

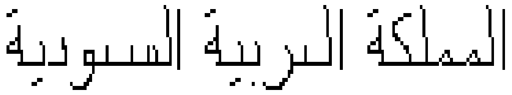
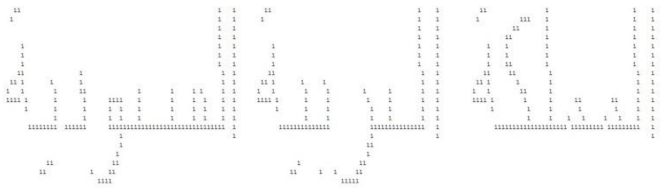


Figure 8. Input image.



```
NSNSNSnullNSEWNESWUDUDUDnullEWSENWNESWnullEWSENWNESWnullNSNSEWUDUDUDnullNSNSEWEWNESWNWSEnullN
SNESWSWNEUDUDnullNESWnull
$NSNSNSNESWnullNSEWNESWUDUDUDnullNSEWGMnullNSNSEWSENWNESWNWSEnullNSEWNESWLDnullNSEWLDLDnullNSN
ESWSWNEUDUDnullNESWnull
$NSNSNSNESWnullNSEWNESWUDUDUDnullNSnullNSEWGMnullNSEWnullNSEWGMnullNSNSNSEWSENWNESWnullNSEWNESW
nullNSEWNESWLDLDnullNSNESWSWNEUDUDnullNESWnull
NSNSNS   NSEWNESWUDUDUD   EWSENWNESW   EWSENWNESW   NSNSEWUDUDUD   NSNSEWEWNESWNWSE
NSNESWSWNEUDUD   NESW   $NSNSNSNESW   NSEWNESWUDUDUD   NSEWGM   NSNSEWSENWNESWNWSE
NSEWNESWLD   NSEWLDLD   NSNESWSWNEUDUD   NESW   $NSNSNSNESW   NSEWNESWUDUDUD   NS   NSEWGM
NSEW   NSEWGM   NSNSNSEWSENWNESW   NSEWNESW   NSEWNESWLDLD   NSNESWSWNEUDUD   NESW
BUILD SUCCESSFUL (total time: 2 seconds)
```

Figure 9. Traversed string and code generated.

6. Conclusions

The proposed extraction and recognition algorithm gives superior results when compared to the approaches, techniques discussed in introduction section. This algorithm is for Arabic language which is written from right to left direction. With minor modifications in coding, revising characters codes, this algorithm can also used be for languages whose script is written from left to right, top to down direction such as Chinese, Japanese, Hindi etc., As the calligraphy of the character varies from one font, size to another, the proposed algorithm faces limitation of font used. This can be surmounted by slight modification in character codes. The experimental results certify that this extraction algorithm is more efficient, accurate and can be used in Arabic Optical Character Recognition (AOCR) systems.

Acknowledgment

This research is supported by King Abdulaziz City for Science and Technology (KACST), Saudi Arabia, vide grant no. AT-32-87

References

[1] Abuzaraida M., Zeki A., and Zeki A., “Feature Extraction Techniques of Online Handwriting Arabic Text Recognition,” *in Proceedings of 5th International Conference on Information and Communication Technology for the Muslim World*, Rabat, pp. 1-7, 2013.

[2] Alasadi A. and Subber T., “Arabic-Text Extraction from Video Images,” *Journal of Basrah Researches (Sciences)*, vol. 39, no. 4, pp. 120-136, 2013.

[3] Aljarrah I., Al-Khaleel O., Mhaidat K., Alrefai M., Alzu'bi A., and Rabab'ah M., “Automated System for Arabic Optical Character Recognition with Lookup Dictionary,” *Journal of Emerging Technologies in Web Intelligence*, vol. 4, no. 4, pp. 362-370, 2012.

[4] Alsaad A. and Abbod M., “Arabic Text Root Extraction Via Morphological Analysis and Linguistic Constraints,” *in Proceedings of 16th International Conference on Computer Modeling and Simulation*, Cambridge, pp. 125-130, 2014.

[5] Alshameri A., Abdou S., and Mostafa K., “A Combined Algorithm for Layout Analysis of Arabic Document Images and Text Lines Extraction,” *International Journal of Computer Applications*, vol. 49, no. 23, pp. 30-37, 2012.

[6] Darab M. and Rahmati M., “A Hybrid Approach to Localize Farsi Text in Natural Scene Images,” *Procedia Computer Science*, vol. 13, pp. 171-184, 2012.

[7] Dinges L., Al-Hamadi A., Elzobi M., Al-Aghbari Z., and Mustafa H., “Offline Automatic Segmentation based Recognition of Handwritten Arabic Words,” *International Journal of Signal Processing, Image Processing and Pattern Recognition*, vol. 4, no. 4, pp. 131-144, 2011.

[8] Elnagar A. and Bentrcia R., “A Multi-Agent Approach to Arabic Handwritten Text Segmentation,” *Journal of Intelligent Learning Systems and Applications*, vol. 4, pp. 207-215, 2012.

[9] Ghorpade J., Palvankar R., Patankar A. and Rathi S., “Extracting Text from Video,” *Signal and Image Processing: an International Journal*, vol. 2, no. 2, pp. 103-112, 2011.

[10] Halima M., Karray H., and Alimi A., “Arabic Text Recognition in Video Sequences,” *in Proceedings of International Conference on Informatics, Cybernetics and Computer Applications*, Bangalore, pp. 603-608, 2010.

[11] Haraty R. and Ghaddar C., “Arabic Text Recognition,” *The International Arab Journal of Information Technology*, vol. 1, no. 2, pp. 156-163, 2004.

[12] Kaushik K. and Suresha D., “Automatic Text Extraction in Video Based on the Combined Corner Metric and Laplacian Filtering Technique,” *International Journal of Advanced Research in Computer Engineering and Technology*, vol. 2, no. 6, pp. 2119-2124, 2013.

[13] Khayyat M., Lam L., Suen C., Yin F., and Liu C., “Arabic Handwritten Text Line Extraction by Applying an Adaptive Mask to Morphological Dilation,” *in Proceedings of 10th IAPR International Workshop on Document Analysis Systems*, Gold Cost, pp. 100-104, 2012.

[14] Meryem H., Ouatik S., and Lachkar A., “A Novel Method for Arabic Multi-Word Term Extraction,” *International Journal of Database Management Systems*, vol. 6, no. 3, pp. 53-67, 2014.

[15] Mohanabharathi R., Surender K., and Selvi C., “Detecting and Localizing Color Text in Natural Scene Images using Region Based and Connected Component Method,” *International Journal of Modern Engineering Research*, vol. 3, no. 1, pp. 331-335, 2013.

[16] Moradi M., Mozaffari S., and Orouji A., “Farsi/Arabic Text Extraction from Video Images by Corner Detection,” *in Proceedings of 6th Iranian Conference on Machine Vision and Image Processing*, Isfahan, pp. 1-6, 2010.

[17] Murthy K. and Kumaraswamy Y., “Robust Model for Text Extraction from Complex Video Inputs Based on SUSAN Contour Detection and Fuzzy C Means Clustering,” *International Journal of Computer Science Issues*, vol. 8, no. 3, pp. 225-234, 2011.

[18] Pan Y., Hou X., and Liu C., “A Hybrid Approach to Detect and Localize Texts in Natural Scene Images,” *IEEE Transactions on Image Processing*, vol. 20, no. 3, pp. 800-813, 2011.

[19] Pratheeba T., Kavitha V., and Rajeswari S., "Morphology Based Text Detection and Extraction from Complex Video Scene," *International Journal of Engineering and Technology*, vol. 2, no. 3, pp. 200-206, 2010.

[20] Saudagar A., Mohammed H., Iqbal K., and Gyani Y., "Efficient Arabic Text Extraction and Recognition Using Thinning and Dataset Comparison Technique," *in Proceedings of International Conference on Communication Information and Computing Technology*, Mumbai, pp. 1-5, 2015.

[21] Saudagar A., Syed A., Al-Tameem A., Al-Otaibi M., and Mohammed H., "Efficient Video Splitting Technique for Scrolling Arabic Text Extraction: a Comparative Study," *in Proceedings of the 2nd International Conference on Applied Information and Communications Technology*, Muscat, pp. 702-707, 2014.

[22] Saudagar A. and Mohammed H., "A comparative Study of Video Splitting Techniques," *in Proceedings of the 23rd International Conference on Systems Engineering*, Las Vegas, pp. 783-788, 2014.

[23] Saudagar A. and Mohammed H., "Opencv Based Implementation of Zhang-Suen Thinning Algorithm Using Java for Arabic Text Recognition," *in Proceedings of the 3rd International Conference on Information System Design and Intelligent Applications*, Visakhapatnam, pp. 265-271, 2016.

[24] Saudagar A. and Mohammed H., "Concatenation Technique for Extracted Arabic Characters for Efficient Content Based Indexing and Searching," *in Proceedings of the 2nd International Conference on Computer and Communication Technologies*, Hyderabad, pp. 567-575, 2015.

[25] Vijayakumar V. and Nedunchezhian R., "Novel Method for Super Imposed Text Extraction in a Sports Video," *International Journal of Computer Applications*, vol. 15, no. 1, pp. 1-6, 2011.

[26] Xiang D., Yan H., Chen X., and Cheng Y., "Offline Arabic Handwriting Recognition System Based on HMM," *in Proceedings of 3rd International Conference on Computer Science and Information Technology*, Chengdu, pp. 526-529, 2010.

[27] Zaraket F. and Makhlouta J., "Arabic Temporal Entity Extraction using Morphological Analysis," *International Journal Computer Linguistics and Applications*, vol. 3, no. 1, pp. 121-136, 2012.

[28] Zhang J., Extraction of Text Objects in Image and Video Documents, Thesis, University of South Florida, 2012.

Abdul Khader Saudagar received his Bachelor of Engineering B.E, Master of Technology M. Tech and Doctor of Philosophy PhD in Computer Science & Engineering in 2001, 2006 and 2010 respectively. His areas of interests are: Artificial Image Processing, E-Commerce, Information Technology, Databases, Web and Mobile Application Development. He has 6 years of teaching experience at both undergraduate (UG) and postgraduate (PG) level and presently working as Assistant Professor in Department of Information Systems, College of Computer & Information Sciences, Al Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, The Kingdom of Saudi Arabia. Dr. Saudagar has published a number of research papers in National, International Conferences and International Journals. He is associated as member with various professional bodies like IACSIT, IAENG, ISTE etc., and working as Editorial Board member, Reviewer for many international Journals.

Habeeb Mohammed working as a lecturer in Department of Computer Science, Al Imam Mohammad Ibn Saud Islamic University , Riyadh, The Kingdom of Saudi Arabia. He completed M.C.A (Master of Computer Applications) in 1999. He has 15 years of teaching experience and a certified Java Professional.