Indian Journal of Science and Technology, *Vol 9(11), DOI: 10.17485/ijst/2016/v9i11/85176, March 2016*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Recognize Candidate Objects as Artificial Text in the Movie

Tawfiq A. Al-assadi, Israa Hadi Ali and Maryam H. Bahar

Information Technology College/ Babylon University, Iraq; tawfiqasadi63@yahoo.com, israa_hadi1968@yahoo.com, maryambahar_4@yahoo.com

Abstract

Objectives: Text characters which appear in a video sequences regarding a valued source of important information for content-based indexing and retrieval applications. **Methods/Analysis:** It's known that the text characters are difficult to be extracted and recognized due to their various sizes, grayscale values and complex backgrounds. This article will introduce a new hybrid methodology based on morphology and chain code for recognizing the identified candidate objects. **Findings:** Applying this morphology operation (thinning, miss-hit) on the candidate objects will enable the determining the number of terminal-points and 3-connection points to construct the feature code for each object. This can be used in comparison between the feature codes for each object with a dictionary that will be created by training sample of characters. If the feature code corresponds to isolated features code in the dictionary, this will enable the system to retrieve the character name, but if the feature code matches more than one or does not match any feature code in the dictionary; it will use the second filtration operation to recognize the remaining objects by using chain code for those objects. This leads to check the chain code of the remained objects, if there is correspondence between checked chain code and the isolated chain code in the dictionary then the character name will be retrieved. Otherwise, the system can be concluded that the object is not a character. Novelty/Improvement: The experimental result shows that the proposed system has better results when compared to previous works that depended on chain code only. Thus, it can greatly reduce the time complexity, as well as, extract and recognize all cases except in the case of an overlapping between objects in the movie.

Keywords: Chain Code, Feature Extraction, Morphology, Text Recognition, Video Tracking

Introduction

The extraction and recognition of the scene and the artificial text from unconstrained or general-purpose (or both) video is a challenging problem. The text extraction and recognition are extremely useful in providing text information about the video content as well as supporting queries via text video retrieval. [1, 2] have studied deeply the text recognition of video mainly focuses on recognizing the text in each single frame independently. Moreover, methods of Multiple Frame Integration (MFI) have been proposed which deal with the text information existing in the video helps to obtain the cleaner background, higher contrast and clearer text 3.

In [4] proposed an algorithm to recognize characters depending on optimal Wavelet Packet (WP), Zernike moments and fuzzy integral. Where fuzzy rules got optimal wavelet decomposition of the given character image, and through optimal wavelet packet put characters in the training set into n subspaces. Then, for each subspace, employed Zernike moments to get the mean membership grades of test characters with respect to each class in the training set, separately. Finally, declared the final class that test characters belonged according to aggregating the above membership grades and using fuzzy integral as fuzzy measures.

In [5], presented character reorganization by using zoning features and projection histogram for extract-

**Author for correspondence*

ing feature vectors with 69-dimensions. Support Vector Machines (SVM) with three linear kernels, Gaussian kernel and polynomial kernel have been used as the classifier in classification stage. He uses 8000 samples in learning stage and 600 samples in testing stage. By use every three kernels of support vector machine the results will be gotten and maximum accuracy achieved by using the Gaussian kernel with gamma equal to 0.16. In pre-processing stage, only image binarization and normalization at center with size 40× 40 are used on all the images of this dataset.

In [6] extend an existing end-to-end solution for text recognition in natural images to video. They present different methods for training local character models and also present methods to capitalize on the temporal redundancy of text in the video. Also, propose a new performance metric based on precision-recall curves to measure the performance of text recognition in videos.

In [7] proposed an off line handwritten character recognition framework utilizing feed forward neural network. A handwritten Sanskrit character is resized into 20x30 pixels and this character is used for training the neural network. After the training process, the same character is given as input to the neural network with different set of neurons in hidden layer and their recognition accuracy rate for different Sanskrit characters has been calculated and compared.

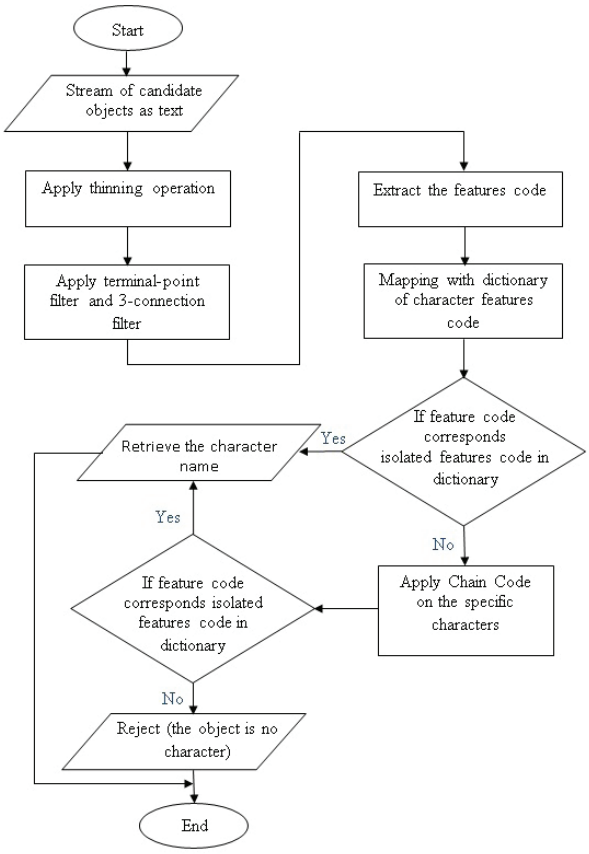

Figure 1. Flowchart of the proposed method for artificial text recognition.

2. Proposed Method

In [8], we proposed a new methodology for detection and extraction of candidate objects as texts appear in the movie based on some features such as (location, area, and texture) of the objects in a movie. The artificial text is characterized by its stability in the area, location and its high degree of homogeneity and similarity textures that used in the proposed method. The proposed text extraction method has several stages: first stage is splitting the input video into number of frames, the second stage is preprocessing (the noise removal, segmentation and object tracking) and features extracting step, the third stage is constructing of the features table for each object in the frame that contains location, area and texture of the objects and the fourth stage is determining the which are selected as a text by comparing the features of objects of the current frame with features of objects in previous frame, and removing the objects in current frame that are not matched with corresponding objects in previous frame. The objects which remain in the final frame are selected as a text.

Therefore, finding a new method for recognizing text embedded in a movie film is based on morphology operation and chain code. Here must take the stream of candidate objects as text from the previous work and finding the bounding box of the candidate objects to calculate the height and width for them to using in all recognition steps below that is shown in Figure 1.

2.1. Thinning Operation

Thinning is a data reduction process that erodes an object until it is one-pixel wide, producing a skeleton of the object. It is easier to recognize objects such as letters or silhouettes by looking at their bare bones. Figure 2 shows how thinning a rectangle produces a line of pixels. There are two basic techniques for producing the skeleton of an object, basic thinning and medial axis transforms.

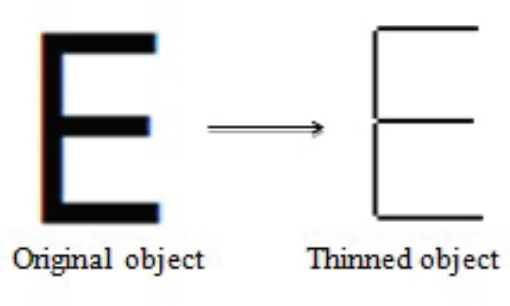

Figure 2. Thinning an object until it is One Pixel Wide.

```
Algorithm Name: Thinning operation
Input: Object (Segment) // foreground represent as black and background represent
as white
Output: Thinned object

Begin
Repeat
For i=1 To Object _pixel_ Width Do
    For j=1 To Object _pixel_ heigth Do // From top to down
            If Object _pixel (i,j)color is black AND have at least one black neighbor then
            Change color of Object _pixel (i,j) to white
        Else If Object _pixel (i,j)color is black AND have eight black neighbors then
            Stay color of Object _pixel (i,j) black
        End if
    End for
End for

For i=1 To Object _pixel_ Width Do
    For j= Object _pixel_heigth DownTo 1 // From down to top
            If Object _pixel (i,j)color is black AND have at least one black neighbor then
            Change color of Object _pixel (i,j) to white
        Else If Object _pixel (i,j)color is black AND have eight black neighbors then
            Stay color of Object _pixel (i,j) black
        End if
    End for
End for

For i=1 To Object _pixel_heigth Do
    For j= 1 To Object _pixel_Width Do // From left to right
            If Object _pixel (i,j)color is black AND have at least one black neighbor then
            Change color of Object _pixel (i,j) to white
        Else If Object _pixel (i,j)color is black AND have eight black neighbors then
            Stay color of Object _pixel (i,j) black
        End if
    End for
End for

For i=1 To Object _pixel_heigth Do
    For j= Object _pixel_ Width DownTo 1 // From right to left
            If Object _pixel (i,j)color is black AND have at least one black neighbor then
            Change color of Object _pixel (i,j) to white
        Else If Object _pixel (i,j)color is black AND have eight black neighbors then
            Stay color of Object _pixel (i,j) black
        End if
    End for
End for
  Until one pixel wide
END // End Algorithm
```

Figure 3. Thinning algorithm.

Thinning erodes an object over and over again (without breaking it) until it is one-pixel wide[9]. Figure 3 illustrates the algorithm of the thinning operation.

2.2. Terminal-point and 3-connections Filter

This step is applying two filters on the objects after the thinning process to create the features code for each object. The first filter consists of 3×3 element, in order to find the entire terminal pixel in a binary image we need to run the hit-and-miss transform with the filter that have just one neighbour and can be called Terminal-point filter, and the second filter that used for finding pixels that have three connections consist of three neighbours called 3-connections filter. Figure 4 show the two samples of filters used in this operation. Figure 5 shows the effect of these two filters on a thinned object.

Figure 6 illustrates the algorithm of the Terminal-point and 3-connections filters.

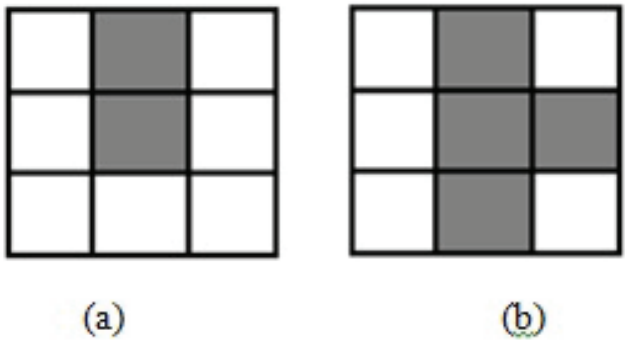

Figure 4. (a) Terminal-point filter, (b) 3-connections filter.

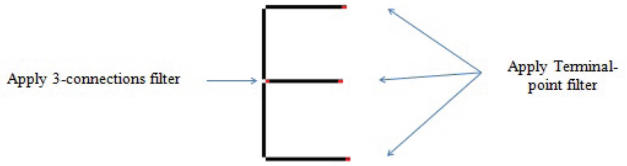

Figure 5. Effect of two filters on a thinned object.

```
Algorithm Name: Terminal-point and 3-connections filters
Input: Thinned object
Output: The number of Terminal-points and 3-connections points
Begin
  Initialize Terminal_point_counter=0 and 3_connections_counter=0
      For i=1 to object_pixel_heigth Do
      For j=1 to object _pixel_Width Do
      If object _pixel (i, j) have one black neighbor then
        Increment Terminal_point_counter by 1
      Else if object _pixel (i, j) have three black neighbors then
        Increment 3_connections_counter by 1
      End if
    End for
  End for
END // End Algorithm
```

Figure 6. Terminal-point and 3-connections filters algorithm.

2.3. Construct Features Code

In this stage, we find the frequency of each filter in the object, and mapping them with the dictionary of the English letters features code. If the feature code corresponds isolated row in the dictionary we can retrieve the character name, otherwise we Pre-processing to specification the character name by applying the chain code in the next step. Table 1 illustrates the structure of feature code.

2.4. Create Dictionary

After applying the two filters on the thinned objects, we built the dictionary table (by training the set of objects) that shows the number of occurrences of each of these filters in the object. The following Table 2 shows the dictionary of printed Alphabet English letters.

Table 2 shows that the five letters were properly recognized, because they have a unique feature code, but for the rest of the characters the system has the following cases:

- The characters that have three terminal points and one 3-connections point (E, F, T, Y), if the x-coordinate of the three terminal points is equal the object is "E", if the x-coordinate of just two terminal points is equal the object is "F", if the y-coordinate of the two terminal points is equal to the y-coordinate of 3-connections point, the object is "T", else the object is "Y".
- The characters that have four terminal points and two 3-connections points (H, K, X), if the x-coordinate of the two pair of terminal points is equal and the y-coordinate of the two pair of terminal points is also equal and the y-coordinate of the two 3-connections points is equal the object is "H", and if the x-coordinate of the just two terminal points is equal the object is "K", else the object is "X".
- The characters that have two terminal points and four 3-connections points (R, Q), if the y-coordinate of the two terminal points is equal the object is "R", else the object is "Q".

Table 1. Structure of features code

Object-ID	Terminal-point	3-connections	Character Name

Table 2. Dictionary of printed Alphabet English letters

Character Name	Terminal-point	3-connections
A	2	3
B	0	4
C	2	0
D	0	0
E	3	1
F	3	1
G	2	0
H	4	2
I	2	0
J	2	0
K	4	2
L	2	0
M	4	4
N	2	0
O	0	0
P	1	1
Q	2	4
R	2	4
S	2	0
T	3	1
U	2	0
V	2	0
W	5	5
X	4	2
Y	3	1
Z	2	0

- The characters that have no terminal points and no 3-connections points (O, D) are exceptional cases, which will be processed specially, if the object has corner point it will be character "D", else the object is "O".
- The characters that have two terminal points and no 3-connections points (C, G, I, J, L, N, S, U, V, Z), in this

case use the chain code that can be explained briefly in the next step.

2.5 Chain Code

In the previous filtration, there were nine letters (C, G, I, J, L, N, S, V, U, V, Z) have the same feature code which containing two terminal points without 3-contactions points that cannot be recognized properly. This leads to use chain code algorithm with 8-connectivity by using terminal points filter, which implies putting the first terminal points founded as a starting point for the chain code algorithm and breaking the chain code algorithm when the second terminal points are reached.

Chain codes are used to represent the boundary of a shape composed of pixels of regular cells by a connected sequence of straight-line segments of specified length and direction. The shape is traversed in a clockwise manner.

There are two types to define the neighbourhood of a given image pixel in digital image processing: 4- and 8-connectivity, as depicted in Figure 7 The 4-connectivity allows only vertical and horizontal movements between adjacent pixels, whereas the 8-connectivity also uses diagonal connections, the example as depicted in

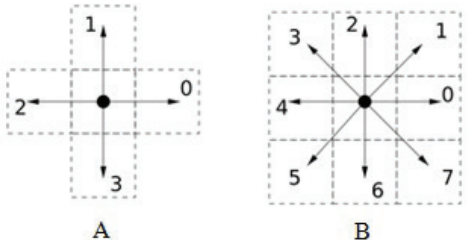

Figure 7. (A) chain code 4-connectivity, (B) chain code 8-connectivity

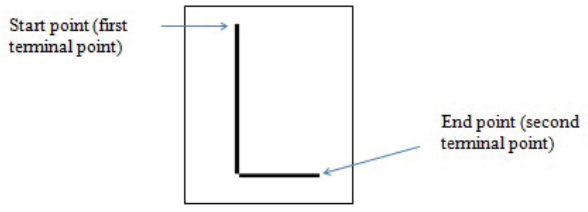

Figure 8. Freeman chain code (Clockwise – start from one), A: Chain code 4-connectivity = 4n 1n, B: Chain code 8-connectivity = 7n 8n 1n.

Figure 8 [10], n is the number of repeat number that preceded, and the number of iterations is not taken into consideration in this work. Table 3 show the dictionary of the printed alphabet English letters with their chain codes.

The recognition of the characters by the chain code has the following cases:

- If the x-coordinate of the two terminal points is equal and the chain code does not equal "7n" the object can be recognized as "C".
- If chain code begins with "1n8n7n" the object can be recognized as "G".
- If chain code is "7n", the object can be recognized as "I".
- If chain code begins with "7n" and ending with "3n", and the y-coordinate of the two terminal points is not equal, then the object can be recognized as "J".
- If chain code "7n8n1n" the object can be recognized as "L".
- If chain code begins with "3n" and ending with "3n", the object can be recognized as "N".
- If chain code begins with "7n8n" and ending with "7n8n" or chain code begins with "1n2n" and ending with "2n1n", the object can be recognized as "S".
- If chain code begins with "7n" and ending with "3n", and the y-coordinate of the two terminal points is equal, then the object can be recognized as "U".
- If chain code begins with "7n8n" and ending with "3n2n" or chain code begins with "8n7n" and ending with "2n3n" and the y-coordinate of the two terminal points is equal, then the object can be recognized as "V".
- If chain code begins with "1n8n" and ending with "8n1n" the object can be recognized as "Z"

2.6. Algorithm

The proposed text recognizing algorithm explained as follows in Figure 9.

3. Experimental Result

In the recognition of the stream candidate objects extracted from the movie shown in Figure 10 and Figure 11, the system process one object at each time by applying thinning operation on this object. Table 4 illustrate the candidate objects after thinning operation in the second

Table 3. Dictionary of the printed alphabet English letters with their chain codes

Character Name	Terminal-point	3-connections	Chain Code
C	2	0	2n3n4n3n4n5n4n5n4n5n6n5n6n5n6n7n6n 7n6n7n6n7n8n7n8n7n8n7n8n7n8n1n8n1n 8n1n2n1n2n3n2n3n2n3n
G	2	0	1n8n7n6n5n6n5n6n5n6n5n6n5n4n5n4n5n4n 3n4n3n4n3n4n3n4n3n4n3n2n3n2n3n2n3n2n 3n2n1n2n1n2n1n8n1n8n1n8n7n8n7n6n
I	2	0	7n
J	2	0	7n8n7n8n7n8n1n8n1n2n1n2n3n2n3n2n3n
L	2	0	7n8n1n
N	2	0	3n7n8n7n8n7n8n7n8n7n8n7n8n 7n8n7n8n7n8n7n8n7n8n7n8n7n8n3n
S	2	0	7n8n7n8n1n8n1n2n1n2n3n2n3n2n3n4n3n4n5n 4n5n4n5n4n5n4n5n4n5n4n5n4n3n4n3n2n3n2n 1n2n1n2n1n8n1n8n1n8n7n8n
U	2	0	7n8n7n8n7n8n7n8n1n8n1n2n1n 2n1n2n3n2n3n2n3n2n3n
V	2	0	7n8n7n8n7n8n7n8n7n8n7n8n7n8n7n8n7n8n 7n8n7n8n7n8n7n8n1n2n3n2n3n2n3n 2n3n2n3n2n3n2n3n2n3n2n3n2n 3n2n3n2n3n2n3n2n3n2n3n2n
Z	2	0	1n8n7n6n7n6n7n6n7n6n7n6n7n6n 7n6n7n6n7n6n7n6n7n6n7n6n7n6n7n6n7n8n1n

```
Name: Text recognizing algorithm
Input: candidate objects
Output: Text
Begin
    For J = first object To the last object Do
        Call thinning algorithm (J)
        Call terminal-point filter and 3-connection filter
        Extract the features code
        Mapping the features code with features code of characters
        dictionary

        If feature code) correspond isolated features code in dictionary
        then
            Retrieve the character name
        Else
            Apply chain code on the Object (J)
            If chain code corresponds isolated chain code in
            dictionary then
                Retrieve the character name
            Else
                Reject object (J)
            End if
        End if
    End for
End
```

Figure 9. The proposed text recognizing algorithm.

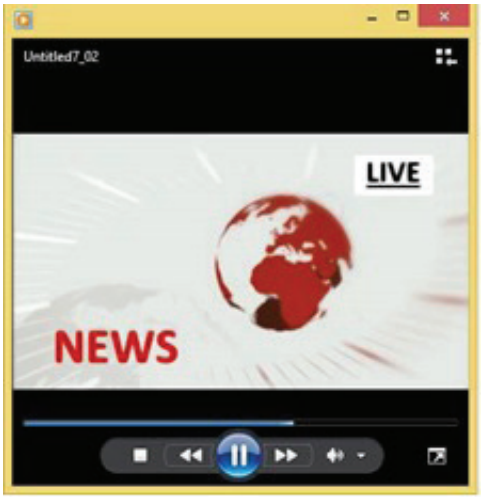

Figure 10. The input movie.

column. Calculating the number of terminal points and 3-connection points will enable us to extract the feature code for the thinned objects. The third and fourth columns in Table 4 will show the numbers of terminal points and 3-connection points in thinned object. Then mapping these feature code with the dictionary explained in section 2.7, if the feature code corresponding isolate feature code in dictionary, retrieve the character name and turn on the true text flag in the Table 4, if the true text flag is false we checked the chain code of the object shown in

Table 4. The feature code of candidate objects matched with the dictionary

Candidate object	Thinned object	Terminal-point filter	3-connections filter	Character Name	True Text
		2	0	-	No
		3	1	E	Yes
		5	3	W	Yes
		2	0	-	No
		2	0	-	No
		2	0	-	No
		2	0	-	No
		2	0	-	No
		3	1	E	Yes

Figure 11. The candidate object.

Figure 12. The result true text.

Table 5, if the chain code corresponds isolated chain code in dictionary we can retrieve the character name, otherwise reject the object that isn't character. Figure 12 show the result objects that is the true text.

4. Conclusions

This article introduced a new hybrid methodology for recognizing candidate objects as texts embedded in the movie based on both morphology and chain code operations. This morphology operation does not recognize all the objects for two reasons, either the feature code matches more than one feature code or does not match any feature code in the dictionary. This leads to using the second filtration operation to recognize the remaining objects by using chain code of those objects. If the two filtration operations do not recognize this object, the system can conclude that the object is not a character.

Table 5. The chain code of reminding candidate objects matched with the dictionary

Candidate object	Thinned object	Chain Code	Character Name	True Text
		3n2n1n8n7n8n7n8n7n8n7n8n7n8 n7n8n7n8n1n8n1n2n3n	N	Yes
		7n8n1n8n1n2n1n2n3n4n5n4n5n4 n5n4n3n4n3n2n3n2n1n8n7n8n	S	Yes
		1n	-	Reject
		7n8n1n	L	Yes
		7n	I	Yes
		7n8n7n8n7n8n7n8n7n8n2n3n2n3 n2n3n2n3n2n	V	yes

References

1. Qixiang Y, Qingming H, Wen G, Debin Z. Fast and robust text detection in images and video frames. Image and Vision Computing. 2005 Jun 1; 23(6):565–76.
2. Datong C, Olobez JM, Bourlard H. Text segmentation and recognition in complex background based on markov random field. 16th International Conference on Pattern Recognition, Proceedings; 2002. p. 227–30.
3. Jian Y, Peng Y, Xiao J. Using multiple frame integration for the text recognition of video. 10th International Conference on Document Analysis and Recognition, ICDAR'09; 2009 Jul 26–29. p. 71–75.
4. Xin Y, Lijuan C, Mou C, Dake Z. Classification and recognition of character using WP decomposition, Zernike moments and fuzzy integral. Sixth International Conference on Fuzzy Systems and Knowledge Discovery, FSKD'09; 2009 Aug 14–16. p. 397–401.
5. Amir N. Persian handwritten digits recognition by using zoning and histogram projection. 3rd Joint Conference of AI & Robotics and 5th RoboCup Iran Open International Symposium (RIOS); 2013 Apr 8–8. p. 1–5.
6. 6. Xuan NP, Wang K, Belongie S. Video text detection and recognition: Dataset and benchmark. IEEE Winter Conference on Applications of Computer Vision (WACV); 2014 Mar 24–26. p. 776–83.
7. Dineshkumar R and Suganthi J. Sanskrit character recognition system using neural network. Indian Journal of Science and Technology. 2015; 8(1):65–69. doi: 10.17485/ijst/2015/v8i1/52878.
8. Tawfiq A, Israa H, Maryam B. Extracting objects from movie as candidate artificial text. International Journal of Digital Content Technology & its Applications. 2015 Aug; 9(4):10–18.
9. Dwayne P. Image processing. C. R & D Publications; 1994.
10. Abdulkhaleq AT, Joda FA. Locational image compression based on chain code representation. IOSR Journal of Engineering . 2014 Jan; 4(1):24–29.