Chapter 3
Chapter 3

3. Methodology of Numeral Character Recognition

3.1 Introduction

In this section, we focus on methodologies of offline numeral character recognition system. Offline numeral character is acquired by the scanner. We applied various preprocessing steps such as normalization, binarization, edge detection technique and thinning. Numeral character image is converted into binary form by the binarization method. Features are extracted from the original image, skeleton image and special points. These features are used in classification stage. Fuzzy rule is based algorithm is used for the classification. Fuzzy rule based technique based on fuzzification and difuzzification process. Numeral character is classified according to the class they belong. Block diagram of proposed numeral character recognition system is as shown in Figure 6. General numeral character recognition system is divided into four stages.

1. Character Acquisition.

2. Preprocessing.
3. Feature extraction.

4. Classification.

Figure 6. Block diagram of proposed Numeral Character Recognition System
3.2 Character Acquisition

The first step in recognition process is to acquire numeral characters. Generally, we use scanner to do this job. Other sensors can be camera, video camera, paintbrush etc.

We have collected different types of samples like standard machine printed numeral character, italic numeral character and handwritten numeral characters.

The handwritten numeral samples were specifically collected from different peoples for this study. All digits used in this study are disconnected. The writers wrote the numeral character in a great variety of size, with different writing styles, and with widely varying amounts of care. The handwritten numeral characters are acquired by scanner at 300 dpi. that yielded a binary image which subsequently stored in compressed format in the memory.

3.3 Preprocessing Stages

The input data obtained by scanning of printed or handwritten numeral character is almost always contaminated with noise that contains redundant information. Preprocessing is needed to remove insignificant scanning artifacts and noise. It is also used to give some low-level organization to the data, or to reduce the redundancy
present in the data. The goal of preprocessing is to increase the quality of recognition, that means more precisely that character is transformed to such that they are more similar to the class they belongs. Preprocessing is important for character recognition process. There is important interaction between preprocessing of character images and feature extraction process. In preprocessing, preliminary steps include normalization, digitization, edge detection and thinning process.

A binary image is composed of a collection of pixels that are either 0 for the back-ground or 1 for the foreground, arranged in a two dimensional matrix.

The notation of pixel connectivity describes a relation between two or more pixels. For two pixels to be connected they have to fulfill certain conditions on the pixel brightness and spatial adjacency. First, in order for two pixels to be considered connected, their pixel values must both be from the same set of values V. For a grayscale image, V might be any range of gray levels, e.g. \( V = \{22, 23, 40\} \) for a binary image, we simply have \( V = \{1\} \).
To formulate the adjacency criterion for connectivity, we first introduce the notation of neighborhood. For a pixel \( p \) with the coordinates \((x, y)\) the set of pixels given by

\[
N_4(p) = \{(x + 1, y), (x - 1, y), (x, y + 1), (x, y - 1)\}
\]

is called its 4-neighbours. Its 8-neighbours are defined as

\[
N_8(p) = N_4 \cup \{(x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1)\}
\]

From this we can infer the definition for 4 and 8-connectivity. Two pixels \( p \) and \( q \), both having values from a set \( V \) are 4-connected if \( q \) is from the set \( N_4(p) \) and 8-connected if \( q \) is from \( N_8(p) \). General connectivity can either be based on 4- or 8-connectivity, for the following discussion we use 4-connectivity. A pixel \( p \) is connected to a pixel \( q \) if \( p \) is 4-connected to \( q \) or if \( p \) is 4-connected to a third pixel which itself is connected to \( q \). Or, in other words, two pixels \( q \) and \( p \) are connected if there is a path from \( p \) and \( q \) on which each pixel is 4-connected to the next one.

A set of pixels in an image which are all connected to each other is called a connected component. A binary image with two connected
components which are based on 4-connectivity. If the connectivity was based on 8-neighbours, the two connected components would merge into one.

Each pixel has eight neighbours. Four of these pixels are immediately adjacent, as they share an edge, and are known as 4-connect neighbours. They are, from Figure 7, pixels P0, P2, P4, and P6. A pixel also has diagonal neighbours, known as 8-connect neighbours, pixels P1, P3, P5, and P7.

![Figure 7. A pixel's immediate neighbours](image)

Each image contains only one pattern. The pattern is typically composed of only one contiguous set of foreground pixels called a blob. In our case, blobs are 8-connected, a constraint imposed by the data. Digital geometry then dictates that the background is 4-connected [89].

A binary image with two connected components which are based on 4-connectivity can be seen in Figure 8.
Figure 8. Two Connected Components Based on 4-Connectivity.

3.3.1. Normalization and Digitization

Normalization is the process of equating the size of extracted character bitmap in order to match extracted character pattern against the database character [48]. The normalization procedures simplify the classification and enhance the performance.

The acquired image of numeral character is in TIFF or BMP format. In digitization, the object is converted into binary form by binarization method [89]. Object is separated from background (1-represent as a region, 0 represent as a no region). This binarized image is put through preprocessing routines so as to smooth the image and eliminate noise, and other artifacts hole produced by the digitizing process.

We have implemented the algorithm for binarization method. Here, we have used 16 bit image. Using binarization method, the image is converted into gray level form.
Binarised image is in one and zero form. That binarised image is stored in the 128x128 array that corresponds to pixel value. The input numeral character zero is as shown in Figure 8.

3.3.2. Thinning

The thinning of elongated objects is a fundamental preprocessing operation in image analysis. Thinning is a morphological operation that is used to remove selected foreground pixels from binary images, somewhat like erosion or opening. It can be used for several applications, but is particularly useful for skeletonization. In this mode it is commonly used to tidy up the output of edge detectors by reducing all lines to single pixel thickness. Thinning is normally only applied to binary images, and produces another binary image as output.

Thinning algorithm should be able to satisfy the following connectedness conditions.

- Connectedness is preserved, for both objects (the black pixels) and their complements (white pixels).
- Curves, arcs, and isolated points remain unchanged.
- Upright rectangles, whose length and width are both greater than 1, can be changed.
The objective is to develop a fast thinning algorithm that will prevent excessive erosion, and give good skeleton, i.e. skeleton that retains the shape of information of original pattern and does not have spurious tails [16][136]. Furthermore, the algorithm should be such that enough information is retained in skeleton to reconstruct, as far as possible, a pattern that is similar to original pattern.

SPTA is an iterative process. It consists of executing numerous passes over the pattern. At each pass, a few dark points are flagged. A flagged point must be such that it is an edge point but not an end point, or break point, and its deletion should not cause excessive erosion in the pattern. At the end of the pass, the thinning procedure stops. Consider a pixel p with eight neighbours in Figure 7.

In the SPTA, an edge-point is defined as black pixel with at least one white 4-neighbour, an end point defined as black point with almost one 8-neighbour and break point is defined as point whose deletion would break the connectedness of the pattern. The SPTA identifies edge-points as one or more of following four types.

1. A left edge-point, having its neighbour P4 white.
2. A right edge-point, having its neighbour P0 white.
3. A top edge-point, having its neighbour P2 white.
4. A bottom edge-point, having its neighbour P6 white.
It may be noted that an edge-point can be of many types. For example, a dark point having neighbours P2 white and P4 white will be a top edge-point and left edge-point simultaneously. To identify such a left edge-point P, SPTA needs to compare the neighbourhood of P with the four windows in Figure 9 (a) – (d). If the neighbourhood matches with any one of four windows, then P is not flagged. It should be noted that the points shown as x and y in the windows are 'don’t cares' points.

![Figure 9. (a)-(d) Possible Combinations for Marking Left-edge Points.](image)

If the neighborhood of the P matches with any of the windows shown in Figure 9(a) - (d), then two situations may arise; If all the x's are white, then P is an end-point; If at least one ‘x’ is a dark point, then P is a break point. Then, in either case, P should not be flagged.
Now, let us examine Figure 9 (a) - (d). If at least one 'x' and one 'y' are dark, then P becomes a break point and hence it should not be flagged. For further analysis, let us assume that all 'x' are white. Then there are eight possible configurations as shown in Figure 10. (S1) - (S8).

Figure 10. (S1) - (S8) Possible combinations when x's are white.

Configuration S1-S3 make P an end point, and configuration S4 makes P a break point. The possible deletion of P in configuration S5 and S6 could cause excessive erosion in slanting strokes of width 2. In configuration S7 and S8, point P is, in fact noise and these configurations never arise at the beginning of the skeletonization process. If configuration S7 were to occur in an intermediate stage of
skletonization, then P would be spurious due to short tail in the original pattern. Such a point may, however, retain some shape information of the pattern and it must not be deleted. Similarly, if configuration S8 were to occur in an intermediate stage of skeletonization, then it would be a singleton (isolated point): its deletion would totally erase the last remaining segment of the pattern. Therefore, in all the configurations of Figure 10. (S1) - (S8), P must not be flagged. By symmetry, we extend our argument to the case in which all y's are white and 'x' can take varying values of whiteness or darkness.

A left edge point that matches with any one of the four windows in Figure 9. (a)- (d) is called a left safe point. For a left safe point the Boolean expression T4 is false, where

\[ T4 = P_0 (P_1 + P_2 + P_6 + P_7)(P_2 + P_3)(P_6 + P_5) \]

A Boolean variable is true, when its corresponding point is dark and is not flagged. It is false otherwise (i.e. if the point is flagged or it is white) thus, tom test whether the left edge point is safe point, the SPTA needs only two tests against Boolean expression T4. The subscript in T4 indicates that the left neighbor of P4 is white. Similarly, we can derive the following Boolean expression.
For right safe – point.

\[ T_0 = P_4 (P_5+P_6+P_2+P_3)(P_6+P_7)(P_2+P_1) \]

For top safe – point.

\[ T_2 = P_6 (P_7+P_0+P_4+P_5)(P_0+P_1)(P_4+P_3) \]

For bottom safe – point.

\[ T_6 = P_2 (P_3+P_4+P_0+P_1)(P_4+P_5)(P_0+P_7) \]

A pass SPTA consists of two scans, where a scan examines every point in pattern. The scanning sequence is either row wise or column wise. In the first scan, all the left edge point and right edge point that are not the left safe point and the right safe point respectively are flagged.

Similarly, in second scan, corresponding top edge points and bottom edge-point are flagged. If there are no flagged points, the procedure stops, else it starts at the next pass.

Extended SPTA

Applications of SPTA on the handwritten characters result in the skeleton along with some fringes or barbs. To remove the fringes we need to incorporate some extension SPTA. As we know SPTA work on character from left to right for removal of pixels. This generally results in some fringes that arise in the right boundary of the character and some fringes may also arise in left boundary. To obtain
the skeleton of the character, deletion should be done uniformly from all the sides of the character. SPTA does not take care of this aspect like the continuity of character. The extended approach uses SPTA for continuity check and consider for removal only those points, which have at least one white ("0") neighbours such that deletion of unnecessary dark pixel (1 pixel) is done from all the side uniformly.

The steps in algorithm are as follows.

Step 1. Scan binary file from left to right.
Step 2. Scan one pixel
Step 3. If the pixel is "1", go to step 4.

    Otherwise go to step 2.

Step 4. Check if the pixel has any "0" neighbours. If yes then go to step 5.

    Otherwise go to step 2.

Step 5. Apply SPTA condition on that pixel to flag it.
Step 6. Repeat the above steps till the end of the character reached and then go to step 7.
Step 7- If any pixel is flag make it "0"

and go to step 1.

Figure 11. Extended SPTA Thinning Algorithm.
The Extended SPTA Thinning algorithm is applied for the numeral character. It gives thinned image.

The original image of numeral character four and thinned image of numeral character four is as shown in Figure 12.

![Image of original and thinned numeral character four]

Figure 12. Original Image and Thinned Image of Numeral Character four.

Edges are places in the image with strong intensity contrast. Since edges often occur at image locations representing object boundaries, edge detection is extensively used in image segmentation when we want to divide the image into areas corresponding to different objects.
We have developed an algorithm for edge detection technique. Edge detection algorithm is used to convert original image into skeleton image. The algorithm is as given below.

Step 1. Scan binary file from left to right.
Step 2. Scan transition pixel.
   Form "1" pixel to "0"
Step 3. If the pixel is "0" then
   Set pixel "1"
   Otherwise go to step 2.
Step 4. Check if the pixel has any "0" neighbor. If yes
   then go to step 5.
   Otherwise go to step 2.
Step 5. Repeat the above steps till the end of the
   character reached and then go to step 6.
Step 6- If any pixel is flag make it "0"
   stop.

Figure 13. Skeleton Image Algorithm.
Using this Edge detection algorithm is used to convert original image zero numeral character into skeleton image. Skeleton image of the numeral character zero is shown in Figure 14.

Figure 14. The Original Image and Skeleton Image