Soft Computing
https://doi.org/10.1007/s00500-019-03978-5

METHODOLOGIES AND APPLICATION

Structural representation-based off-line Tamil handwritten character recognition

M. Antony Robert Raj[1] · S. Abirami[1]

© Springer-Verlag GmbH Germany, part of Springer Nature 2019

Abstract
Tamil handwritten character recognition system enormously depends on its character features. This paper deals with the feature extraction and the three ways of feature predictions that are experimented in order to grasp features from various Tamil characters possessing variations in style and shape. Shape, shape ordering and location-based instances are the features predicted from the characters. The key features of this paper are the strip tree-based hierarchical formation which deals with the shape features of the characters, the implementation of the Z-ordering algorithm for addressing the structure ordering and finally the representation of PM-Quad tree that deals with extracting locations of the character features. A hierarchical classification algorithm based on support vector machine is used for predicting the character from its character features using divide-and-conquer procedure. Proof of this work shows that this work can address more characters and its varied shapes.

Keywords PM-Quad tree · Strip tree · Z-ordering · SVM · Chain code

1 Introduction

In the study of character recognition, several research works have come up in the categories of printed and handwritten documents that are available in off-line mode. Various works are carried out in various applications and have achieved considerably good results in online and printed (off-line) character recognition works. Increasing demand has motivated for the upcoming research works to extract machine-understandable code from handwritten character papers. Handwritten character recognition is a tough task, due to the presence of slant, discontinuities, large variation and the character's free style property. Also this recognition work is highly dependent on the writer's handwriting (Antony Robert Raj and Abirami 2012). This paper deals with the off-line recognition of handwritten Tamil language characters. Tamil is a widely spoken language that has created more attention and demand on character recognition work in the recent years. Several printed and online Tamil character recognition researches are established and have achieved considerable results. Various research works are incoming with difficulties, left open with highly challenging tasks. When analyzing handwritten Tamil characters, it was found that they are highly writer dependent and have complex structure styles, varied shapes, direction variations, location mismatch, shape similarity, structure discontinuity and are based on a large character set (247 character which includes 12 vowels, 18 consonants, 216 combined characters and one special characters). Figure 1a–e shows the structural complexities present in the handwritten Tamil characters.

Generally, in character recognition process two different methods are pursued to analyze the character portions that are essential for symbolizing a character, i.e., statistical and structural. The statistical analysis (pixel-based) alone is not sufficient to address character portions in the process. A qualitative process (structural) that is shape-based is also necessary to address the character portions. As per the previous analyses (Antony Robert Raj and Abirami 2012, 2013, 2014, 2015a, b; Shyni et al. 2015), the statistical and structural ways of representation have played a key role in recognizing few of the Tamil characters. This work depends on both structural and statistical ways of symbolization for extracting character portions.

Communicated by V. Loia.

✉ M. Antony Robert Raj
antorobert@gmail.com

S. Abirami
abirami.murugappan@gmail.com

[1] Department of Information Science and Technology, Anna University, Chennai 600 025, India

Published online: 22 April 2019

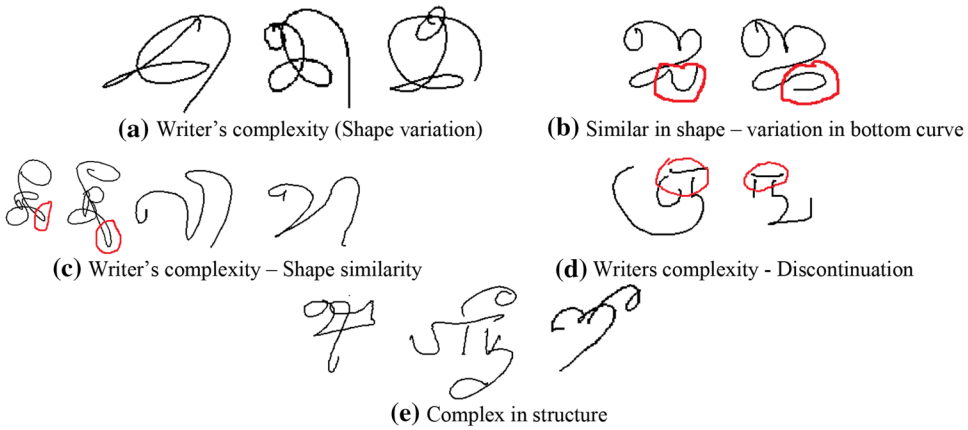


Fig. 1 **a** Writer's complexity (shape variation). **b** Similar in shape—variation in bottom curve. **c** Writer's complexity—shape similarity. **d** Writers complexity—discontinuation. **e** Complex in structure

Three different novel approaches were used to deal with unique classes of Tamil languages which are used frequently. The character structure is pre-extracted by junction point procedure (Antony Robert Raj and Abirami 2013, 2014). The junction point procedure yields the needed structure without affecting real shape, but the feature count may increase if the structure contains unnecessary loops or junction points. In general, the junction points and loop would not affect the real structure of the shapes. Since these components are essential to discover the shape of the structure, to discover the shape of the structure, strip tree (ST) (Antony Robert Raj and Abirami 2015b; Dana and Ballard 1981) approach is experimented here. ST discovers the shape by rectangular formation, and also, the same is represented in the hierarchical tree. The ST yields the same tree structure for different shapes as well. Hence, the new approach is essential with strip tree to discover several shapes with variations.

Two different statistical approaches are identified to expose the location of the structure, i.e., Z-ordering and PM-Quad tree approaches. The primary reason for applying location-based feature extraction features is recognizing the features based on their default characteristics. Each image has its own unique shapes and its lying on the respective location. Identifying the location reduces the complexities of informal writing. Here, the handwritten structure may not fit into an exact location; hence, PM-Quad tree concepts are applied to identify the approximate location. Feature representing order differs from Z-ordering and quad tree approaches; hence, both are used to get the locational features. PM-Quad tree reveals the location by quad division and the hierarchical tree. The feature representations start from the location of structures' end points to end points which are noted whether there are any bend points identified. Additionally in the PM-Quad tree approach, the quad division has taken place if the pixel values are found in the location and the same is represented by a tree. The Z-ordering locates the value during its *N*-order travel which starts from the midpoint of the super-sub-gird (*N*-order visits have taken place of each midpoint of one grid to another). If the foreground pixel is visited during its travel, then its corresponding location is taken from bit shuffling procedure. Bit shuffling approach assigns the binary value location specification to the each sub-grid. Lastly, these formations are finally converted into numerical values in accordance with the structure and length of the tree. These numerical formations have been taken as features and presented in the form of input to hierarchical SVM.

From the analysis of various works, it was found that a combination of statistical and topological features are highly supported to classify the handwritten characters of languages like Devanagari, Bangla, Latin, Telugu and Arabic (Das et al. 2012), since each of them was not enough to address the variations found in characters. It was also identified that the combination of these two features gave improved results taking into consideration all varieties and variations present in the character structure of various languages (Das et al. 2012). Multicolumn multiscale convolutional neural network (quad tree-based staggered prediction model has been proposed for faster character recognition) approach is used (Roy et al. 2016; Sarkhel et al. 2017) for classifying the characters where the quad tree-based features were used for supporting the task of classification. Statistical profile features were extracted diagonally from *X*-direction and *Y*-direction of the pixel position. These features were then given to the neural network classifier for predicting the results. Improvement was needed in the statistical algorithm to achieve the expected results in recognizing handwritten Gujarati numerals (Desai 2010). A new structural feature division technique was implemented to extract the features (Lei et al. 2012; Dubey and Sinthupinyo 2010), where the division points were collected by the recursive subdivision of the character image. Support vector machine with radial basis function was utilized here for achieving recognition results from character image (Yang et al. 2013). A structural quad tree concept based on a statistical long run features was

experimented to extract the features from the Bangla characters. Additionally, the convex hull-based feature also supported this experiment (Nibaran Das et al. 2015). Angular, distance, vertical and horizontal span features were extracted from each character for classifying the Arabic characters using the hidden Markov model. In this experiment, many character samples with high-end variations were considered. Recognition has been achieved with good results. (Mahmoud 2008). Statistical feature extraction techniques such as density of foreground pixels, number of transition, derivative features, mean, variance, min and max of histogram features and structural features like local concavity feature and skeleton-based features (feature points, cusp points and infection points) were experimented to achieve good recognition results from handwritten samples of Arabic characters (Benouareth et al. 2008).

Numerous contributions are coming up to provide solution for handwritten Tamil character recognition. Few of them possess advancement in their contribution. More importantly, purely statistics-based contribution has given a better solution for ten Tamil characters in the work of Rajashekararadhya et al. (Rajashekararadhya and Vanaja Ranjan 2008a, b, 2009; Rajashekararadhya et al. 2008). They have selected characters that denote Tamil numerals (0–9). Zoning and zone centroid-based pixel manipulation have been incorporated with the neural network classification algorithm and are found to produce good results. Another successful classification algorithm, namely the hidden Markov model, has been experimented on the feature values extracted from profile and transition concepts. Sigappi et al. (2011) have implemented this on the varying strokes of the Tamil characters. With a help of support vector machine, Shanthi and Duraiswami (2007, 2010) have recognized 34 handwritten Tamil character samples. The pixel density values have been extracted from character image as features. The level of complexity of characters is to be improved in this work. In Bhattacharya et al. work (2007), they have used unsupervised clustering algorithm to create a small class. Then a supervised classification algorithm has been applied on those classes to get the predicted class, where the samples are collected from HP handwritten Tamil datasets (HP-India 2013).

SVM is used to achieve success in several discrete feature environment such as the gradient direction histogram feature (Kirsh, Roberts, Sobel,) (Liu and Suen 2009), SURF (speeded up robust features) feature and curvature feature (Desai 2010), quad tree-based longest run features and convex hull-based features (Nibaran Das et al. 2015), convex hull-based features and CG-based quad tree partitioned longest run-based local features (Sarkhel et al. 2016), sliding window-based feature, Marti–Bunke feature (three global features are used to capture the fraction of foreground pixels, the center of gravity, the second-order moment, position of the upper and lower profile, number of foreground to background pixel transitions, fraction of foreground pixels between the upper and lower profiles and the gradient of the upper and lower profile with respect to the previous column) and pyramid histogram of oriented gradient for the middle zone (Roy et al. 2016), R-HOG (rectangle histogram of oriented gradient) and Sobel filter–Sobel kernel used to calculate the horizontal and vertical components of the gradients (Kamble and Hegadi 2015) and modified shadow features, octant centroid features, distance-based features, number of loops and longest run features (Das et al. 2012). Also, SVM used to classify the characters in handwritten Tami characters recognition as well.

2 Proposed system

2.1 Phases of THRS

Several phases that are default in pattern recognition are selectively chosen and implemented in this THCRS system (Antony Robert Raj and Abirami 2012, 2013, 2014, 2015a, c; Shyni et al. 2015). The general preprocessing algorithms used in previous works are employed here for achieving the needed image. As depicted in Fig. 2, the character image is taken into three levels of treatment: preprocessing, feature selection and classification for this recognition framework. The preprocessing steps include binarization, noise removal, skeletonization and normalization that help to get a two-tone, noiseless, thinned and standardized image to be used for further processes. Binarization is a technique which is used to get two-tone image (black and white image from color or grayscale image). This is done by Otsu's technique, where the thresholding peak values are calculated to separate the black and white image. Noise removal is a techniques used to remove if unwanted pixels are available in image with the help of Gaussian filter. To peal of the unwanted portion of the character's image structure, the skeletonization techniques are used, where the character structure is converted into single-pixel structure without affecting the real shape of the structure. Eight ways of thinning algorithmic procedure are used for skeleton the character structure. Normalization techniques are used to get standardized image in the form of 200×200 so that algorithmic approaches can be applied. Further the image is taken into the feature selection and extraction processes.

In the process of optical character recognition (OCR), segmentation is an important process, but the dataset which is considered here is taken from HP-India laboratory, where each character is considered in an individual manner. Junction point-based feature division concept is experimented here for selecting and extracting the features (Antony Robert Raj and Abirami 2012, 2013, 2014, 2015a, b; Shyni et al.

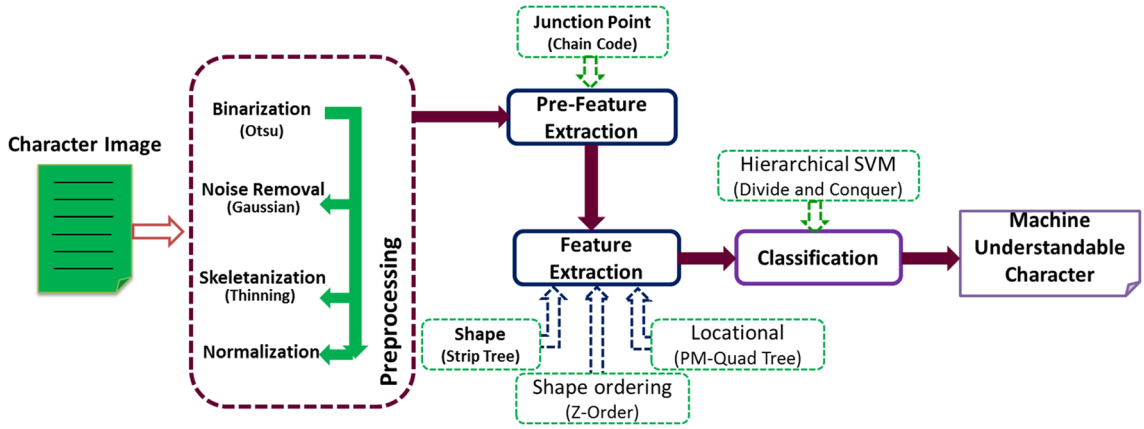


Fig. 2 Flow structure of THRS

2015). This is achieved through the eight directional chain code procedures. It is used to separate features, because the general complexity of algorithm will increase when the entire structure of the image is considered and the zone or other treatments may return huge amount of features. Junction point division helps to divide the structure and minimize the count of the same. The most important process in our phase is feature extraction. Here, the features are extracted via three ways of feature extraction processes that are done by hierarchical tree representation and bit shuffling. Finally, the characters are classified by the classification algorithm called one-vs-one hierarchical SVM.

2.2 Feature selection and pre-feature extraction

Instead of directly applying the feature extraction algorithm on the entire character image, it is worth dividing the character portion into different images and then applying the procedure on the sub-image portions at the microlevel. For this separation, as employed in previous works, junction points have been favored. Eight directional chain code algorithms are preferred for completing this task. It traverses all over the pixels one by one until any one junction point is found to break the journey (Shyni et al. 2015; Antony Robert Raj and Abirami 2014, 2015a, b). Junction points are selected as statistical algorithmic procedure is not stable if the structure varies. Shape-based feature extraction approaches cannot be applied directly on the character's components if the structure is complex. With this consideration, the junction point approach is proposed to separate the character structure. When applying the extraction procedures on a selected structure, the rest of the selected structures are hidden as depicted in Fig. 3.

2.3 Feature extraction

This part of work is crucial in THCRS. With suitable experiments, three ways of analysis have been identified for dealing with handwritten Tamil character structures, where one method minutely examines the structure location, another one inspects the order of the shape and the final one probes into the shape structure. All these are possible in structural mode of representation. The hierarchical tree formation and bit shuffling has been chosen to attain our goal. Two tree forms are preferred by the way of experiments, namely ST formation for shape features and the PM-Quad tree formation for locational features. With this, the Z-ordering algorithmic procedure is applied for finding the order location of the structure. The final feature values are obtained by the numerical values which are derived from these tree structures and order formations.

2.3.1 Strip tree

ST technique is used for representing map features such as contour lines, rivers and roads, which are in the form of curves. Representing curves is a challengeable task. Here, a binary tree hierarchical representation with a special datum at each node has been implemented where the datum is strip and the hierarchical binary tree containing strip data is called strip tree (Dana and Ballard 1981). The lower level of tree nodes represents the structure of curve. When the curve structure is complex or more twisted, then the leaf nodes of the tree increase (Antony Robert Raj and Abirami 2015b).

The given shape is placed within a rectangular box and a line is drawn between two end points ($x1$, $y1$) and ($x2$, $y2$) are marked. If the line touches any pixel on its traversal, then that portion will be further divided into two rectangles as shown in Fig. 4a. The width of the two rectangles is

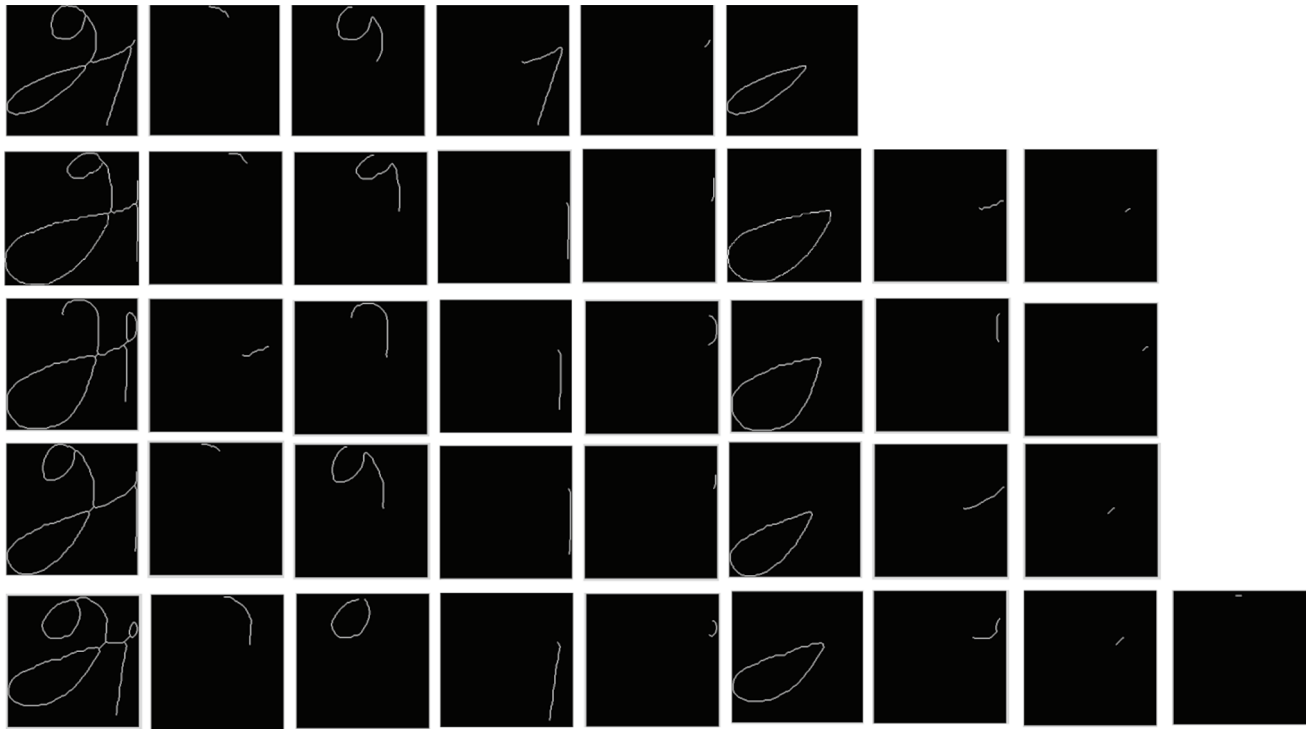
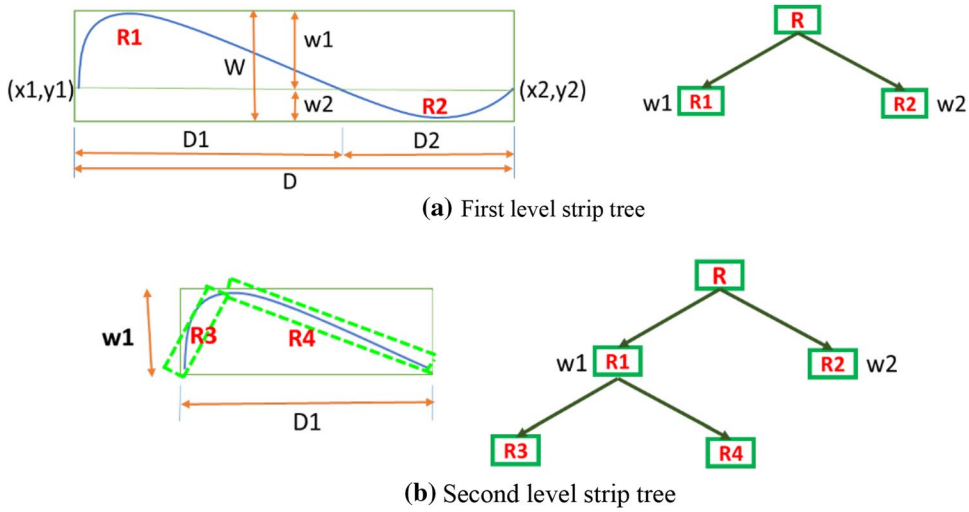


Fig. 3 Junction point-based feature selection—feature uniqueness

Fig. 4 **a** First-level strip tree. **b** Second-level strip tree



computed as $W1$ and $W2$. The width of the curve is calculated by $W = W1 + W2$, and the character portion is kept in the rectangular box (R). If the line does not meet any point (pixel) in the middle, then the rectangle is drawn between the end points with width $W1$ as shown in Fig. 4b. The distance (D) of the shape is calculated as shown in the equation

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{1}$$

The rectangles $R1$ and $R2$ are represented by leaf node in the tree of the root node R as shown in Fig. 4a. If the width of any one rectangle is less than or equal to threshold_1 (default: two-pixel distance), then no further rectangular division is possible. If not, based on the portion touching the rectangular box it can be further divided and this is marked as leaf nodes ($R3$, $R4$) in the tree as shown in Fig. 4b. Hence, the depth-first searching algorithm is used for retrieving the numerical values from the hierarchical tree

for further process. The following formations are possible for drawing the ST.

ST formation 1:

Term 1: Dot (one or two pixels are found in the image)
Term 2: Short line or short curve (very few pixels are found in the image)
Term 3: Linear shapes in the image (probably or exact)

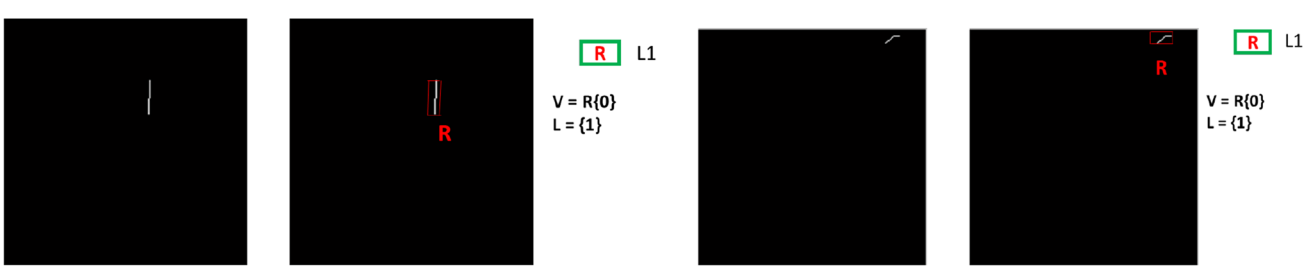
2.3.1.1 ST analysis of Term 1 (dot) The pixel count of the sub-character image does not exceed more than two; then, the width and length of the rectangle are approximately equal ($D = W$ or $D \approx W$). If the width and height are below the threshold value, then only one rectangle or square is possibly placed on the dot meant by root node R as shown in Fig. 5. The tree cannot be extended more than the root (R).

```
Tree formation
Input    : Length (D) and Width (W), Root rectangle (R), P is pixel
Output   : Level of the tree and vertex of tree

Begin
While (P is true) /*Pixels are found in the image*/
        Formation of rectangle has been completed /* Width and Length of box is W and D respectively*/
        If W and D are equal and limits to threshold value
        /*analyzed weather rectangle formation has been possible*/
R ← Root and Pointer
        If R has no edges
                Return ϕ
        For each vertex add edges if found
                Return L← 1; V←ϕ /* return Level and vertex of tree*/
End
```

- Level of this tree is one (L(1))
- Vertex of this tree is 0 (root) V(0)

Fig. 5 Strip tree representation for dot



(a) Strip tree representation for short line **(b)** Strip tree representation for short curve

Fig. 6 **a** Strip tree representation for short line. **b** Strip tree representation for short curve

2.3.1.2 ST analysis of Term 2 (short line or curve) The pixel count of the image is more than two, and the pixel distance used for width calculation is less than or equal to five (threshold_2); then, the length of the shape is higher than width, i.e., ($D \lesssim$ threshold_2 and $D \neq W$. Only one rectangle can be placed on the character portion of the image, and this rectangle is set as a root node as shown in Fig. 6a, b. Here the tree cannot proceed further beyond root R.

- Level of this tree is one ($L(1)$)
- Vertex of this tree is 0 (root) $V(0)$

ST formation 2:

Term 1: Curvy shape is found in the image and the width of the curve is more than threshold_1
Term 2: Any unshaped curve is found in the image and width of the same is more than threshold_1

Tree formation

/ distance of the structure just below the threshold value 2 (Default threshold value to max. of five pixel distance), the distance and width are not equal and width of the structure just above the threshold value 1 (default threshold value to max. of two pixel distance)*/*
Begin
Input : Length (D) and Width (W), Root rectangle (R), P is pixel
Output : Level of the tree and vertex of tree

Begin
While (P is true)
    Formation of rectangle has been completed
    If *W and D* are not equal and Any of W, D not limits to threshold value
    */*analyzed weather rectangle formation has been possible*/*
R ← Root and Pointer
    If *R has no edges*
        Return ϕ
    For each vertex add edges if found
        Return L← 1; V←ϕ */* return Level and vertex of tree*/*
End

- Level of this tree is one (L(1))
- Vertex of this tree is 0 (root) V(0)

2.3.1.3 ST analysis of Term 3 (linear shape) There is a linear shape or curved shape found in the image; then, the width and length of the image are not equal ($D \gtrsim$ threshold_2 and $D \neq W$). The width of the image is not more than two-pixel distance ($W \leq$ threshold_1); then, only one rectangle can be placed on the character portion of the image. This is set as the root node of the tree and the tree cannot be extended from the root node as shown in Fig. 7.

Tree formation

The formation which is discussed in the analysis of Term 2 is used here to form the tree.

2.3.1.4 ST analysis of Term 1 (curvy shape) The rectangle is drawn on the character portion of the image (R), and the following steps are implemented for forming the ST.

1. The end points (as discussed earlier ($x1$, $y1$), ($x2$, $y2$)) are noted
2. The length (distance D) and width (W) of the rectangle are computed
3. While $D >$ threshold_2 and ($D \neq W$ or $D = W$) then

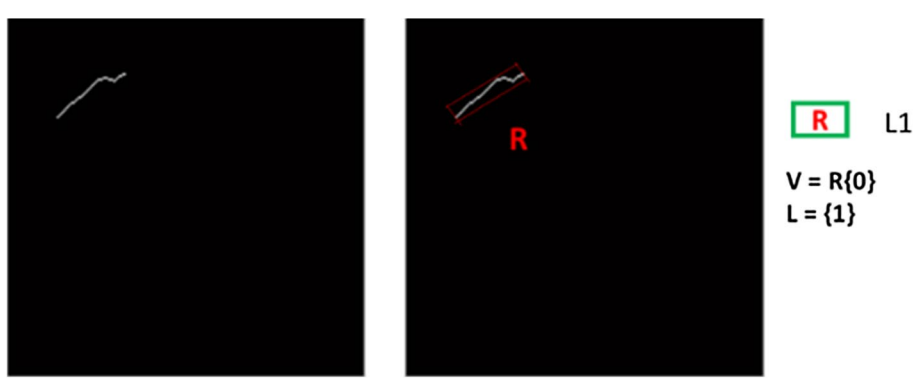


Fig. 7 Strip tree representation for line or curve

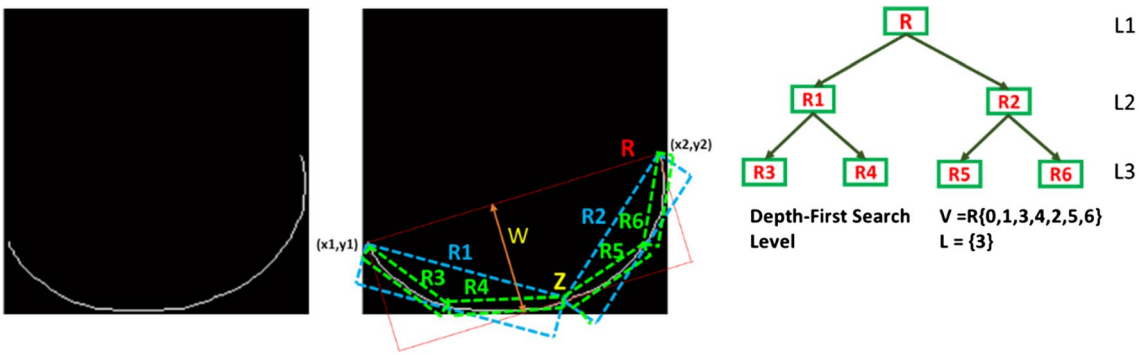


Fig. 8 Strip tree representation for curvy shape

/* distance of the structure above the threshold value 2 (default threshold value to max of five pixel distance), the distance and width are not equal or equal*/

4. If $D >$ threshold_2 is ensured
5. If Steps 3 and 4 are true, then
   - The point (Z) is identified, where the pixel point is touching any portion of the rectangle other than two end points.
   - The sub-rectangles ($R1$, $R2$) are drawn between ($x1$, $y1$) and Z, ($x2$, $y2$) and Z.
6. Steps 2–5 is repeated for the rectangles $R1$ and $R2$, and the super-sub-rectangles are drawn under them.

The above process comes to an end, if the width of the super-sub-rectangle is less than or equal to threshold value ($W <$ threshold_1)/**default threshold value not more than two-pixel distance* */; Fig. 8 shows the tree representation of a curved shape and the features extracted from the same.

```
Tree formation
/* width and height of the structure is equal or not equal*/
Input    : Length (D) and Width (W), Root rectangle (R), P is pixel, Left Sub root(LSR), Right Sub root (RSR)
Output   : Level id tree and vertex of tree
Begin
S1:  If R has edges          /* R Representing root pointer */
        If LSR has found                    /* LSR Representing Left Sub root pointer */
                If LSR has edges
                        L ← L+1             /*the level of tree is initialized into one and it has been incremented*/
                        Continue the step S1
S2              elseif LSR has no edges
                        Return ϕ
                        For each vertex add edges if found
                        Return L← 1; V←ϕ /* return Level and vertex of tree*/
                        Continue the step S3
S3:     Elseif RSR has found
                If RSR has edges            /* RSR Representing Right Sub root pointer */
                        L ← L+1             /*the level of tree is initialized into one and it has been incremented*/
                        Continue the step S1
                elseif RSR has no edges
                        Return ϕ
                        For each vertex add edges if found
                        Return L← 1; V←ϕ /* return Level and vertex of tree*/
        Continue the step S1
End
```

- Level of this tree – Last leaf level
- Vertex of this tree – Leaf node information returned by depth first search

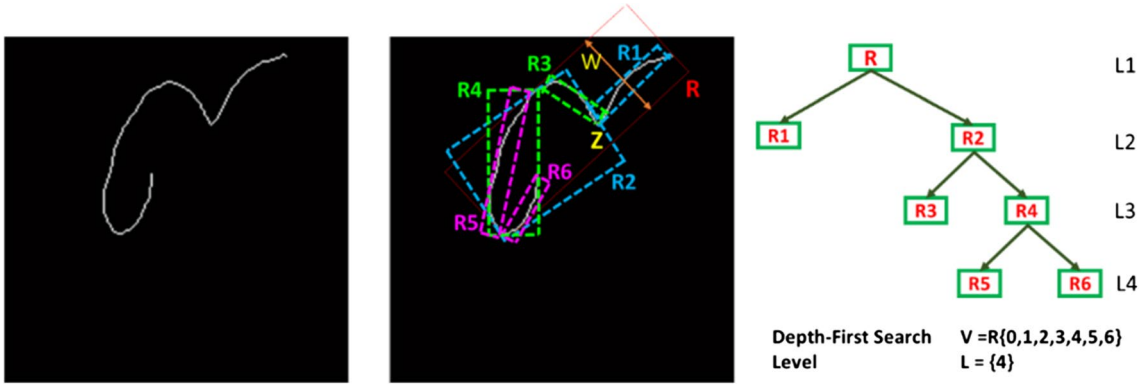


Fig. 9 Strip tree representation for unshaped curve

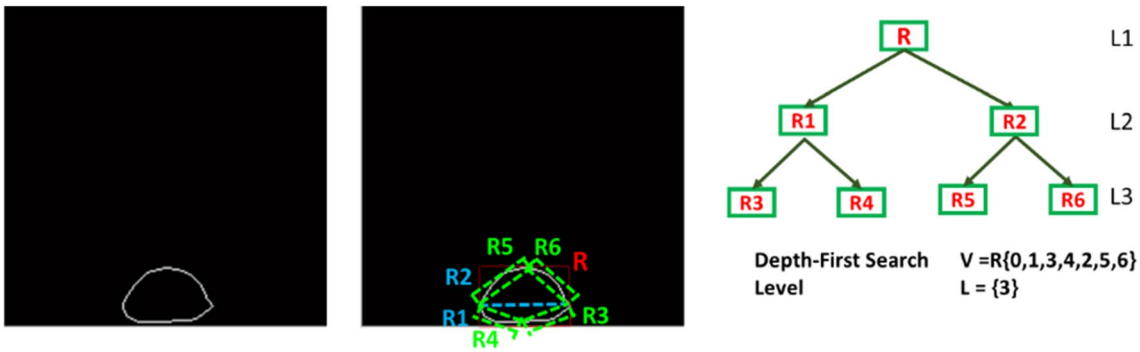


Fig. 10 Strip tree representation for closed curve

2.3.1.5 ST analysis of Term 2 (unshaped structure) The character portion is fixed in the rectangular box (*R*). If the shape of the character portions is an unstructured curve, then the complexity level of the curve structure will be high. Under this situation, the uneven breaking point (*Z*) is identified based on the pixel position and two rectangles are placed between two end points and the break point *Z* (*R*1, *R*2). Further the steps followed for the analysis of Term 1 in formation 2 are adopted.

Tree formation

The procedure discussed in analysis of Term 1 of formation 2 is utilized here also. Figure 9 shows the tree and the feature results of the analysis of Term 2

- Level of this tree—last leaf level
- Vertex of this tree—leaf node information returned by depth-first search

ST formation 3:

Term 1: The shape of the character portion is a closed curve (No end points)

2.3.1.6 ST analysis of Term 1 (closed curve)

1. A rectangle is drawn on the character portion of the image (R).
2. A horizontal line is drawn in the middle of the rectangular box. This breaks the closed curve into two halves.
3. The steps that were followed in the analysis of Term 1 in formation 2 are applied here.

Tree formation

The same procedure which was discussed in formation 2 is used to achieve the ST formation. The resultant image and features are shown in Fig. 10

- Level of this tree—last leaf level
- Vertex of this tree—leaf node information returned by depth-first search

The formations discussed above are applied on pre-extracted features extracted from 100 unique characters chosen from Tamil language. In this experiment, tree representation is formed by ST procedure and the numerical features are extracted by using the depth-first search algorithm.

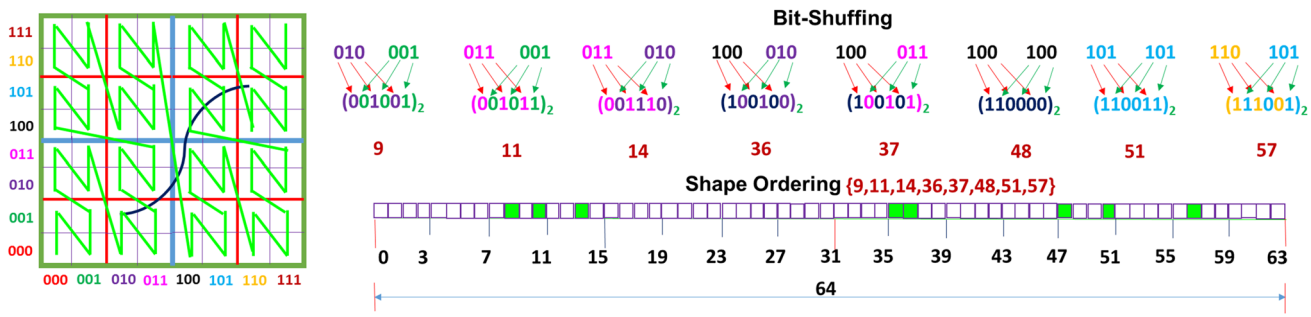


Fig. 11 Sample shape ordering representation

2.3.2 Z-ordering

In a multidimensional space, Z-ordering curve (Poullot et al. 2007) is used as a space-filling procedure, where the given space is first divided into multiple equal grid portions, and then, the '*Z*' or '*N*' shape mapping is applied on the space. The Z-ordering concept is a suitable method to handle the location ordering approximately. According to this procedure, if the space is divided into four equal grids, then only one '*Z*' or '*N*' ((2×2)—first-order single-bit binary representation) shape mapping is possible that fills all the blocks. Further, if these four grid blocks are subdivided into four more equal grids (16 (4×4)—second order), then a 2-bit binary mapping is needed to represent each block. The grid division process continues until the goal is reached. Bit shuffling approach is used for filling the 2-bit binary values in each super-sub-grid.

The same procedure is adopted in this work in order to represent the shape of the character structure by ordering the shapes. Third-order Z-order curve representation is imposed to address the order of the curve as a feature. The 'N'-order mapping starts from midpoint of the left bottom grid and travels all over the blocks as shown in Fig. 11. The pixel points are visited during this travel; then, the corresponding locations are taken as features. Figure 11 also shows the (8×8) grid sub-character image mapping procedure to address the shape ordering along with the location.

Assuming the shape in the image as shown in Fig. 11, a three binary bits (111) representation is needed to map onto a decimal value '7' that is considered as a feature. In the representation, the rows are taken as $\{r_0, r_1, r_2 \ldots r_7\}$ and the columns are taken as $\{c_0, c_1, c_2 \ldots c_7\}$. The polynomial notation of each row and column is as follows:

$$0 = 0$$
$$O^0 = 1$$
$$O^1 = x$$
$$O^2 = x + 1$$
$$O^3 = x^2$$
$$O^4 = x^2 + 1$$
$$O^5 = x^2 + x$$
$$O^6 = x^2 + x + 1$$

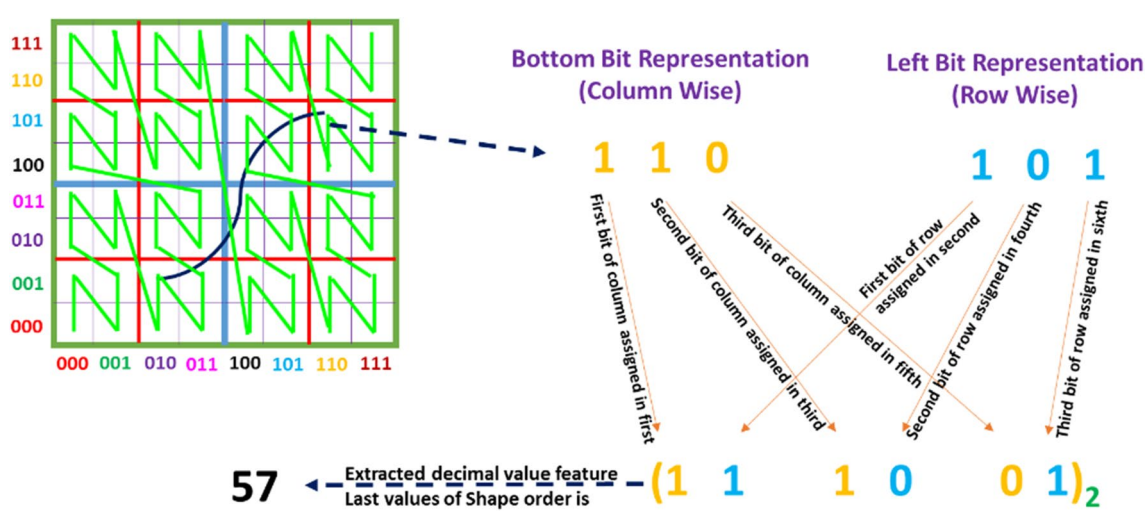


Fig. 12 Bit shuffling order for feature extraction

When applying $x = 2$, the binary bit representations are

$$r_1 = c_1 = 0 = 0 \rightarrow (000)_2$$
$$r_2 = c_2 = 1 = O^0 \rightarrow (001)_2$$
$$r_3 = c_3 = x = O^1 \rightarrow (010)_2$$
$$r_4 = c_4 = x + 1 = O^2 \rightarrow (011)_2$$
$$r_5 = c_5 = x^2 = O^3 \rightarrow (100)_2$$
$$r_6 = c_6 = x^2 + 1 = O^4 \rightarrow (101)_2$$
$$r_7 = c_7 = x^2 + x = O^5 \rightarrow (110)_2$$
$$r_8 = c_8 = x^2 + x + 1 = O^6 \rightarrow (111)_2$$

This notation is applied for each row and column of the character image which is subdivided into 64 blocks. The shape ordering values (three-order bits' values) are converted to decimal values by applying the bit shuffling procedure from Z-ordering.

2.3.2.1 Bit shuffling In the bit shuffling procedure, the grids that deal with the image are selected. This is used to locate the order of the pixel from the exact location. The bit values that are matching a grid location are chosen column- and row-wise as shown in Fig. 12.

The bit formation takes place as described below

- Order 1 → Bit values represented by column
- Order 2 → Bit values represented by row
- Shuffling
  - Position 1 → (left to right) First bit value represents the column
  - Position 2 → (left to right) First bit value represents the row
  - Position 3 → (left to right) Second bit value represents the column
  - Position 4 → (left to right) Second bit value represents the row
  - Position 5 → (left to right) Third bit value represents the column
  - Position 6 → (left to right) Third bit value represents the row
- Resultant binary values → Decimal value
- Features ← Decimal value

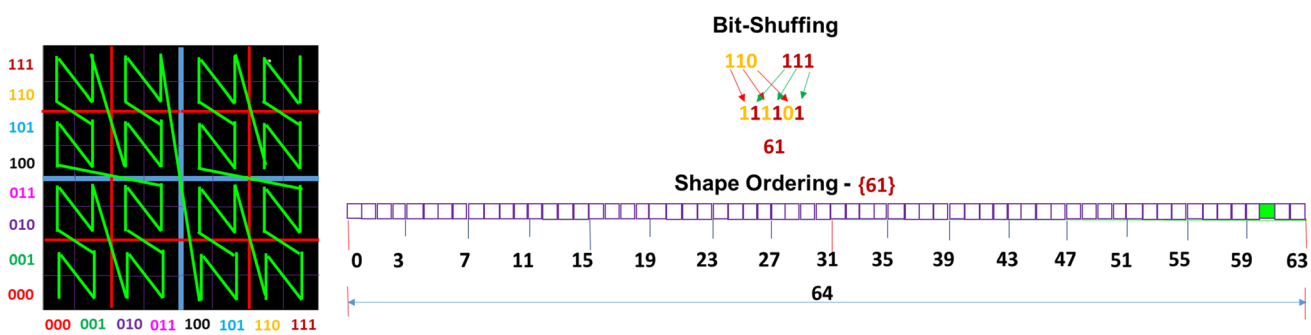


Fig. 13 Z-order representation for dot

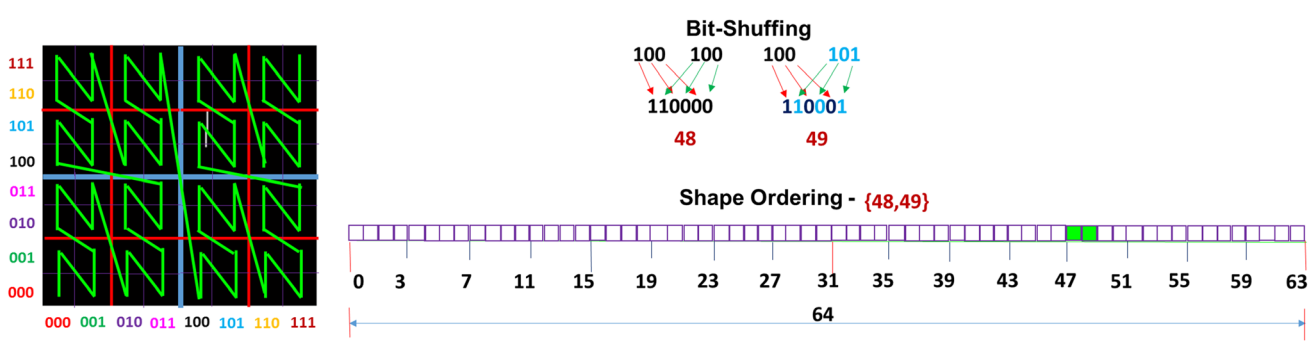


Fig. 14 Z-order representation for short line or curve

The bit values representing the column and row of a grid are taken, and the bit values are arranged as shown in Fig. 12. The resultant binary values and its equivalent decimal values specify the location of the grids. Few formations are possible when applying this procedure on the image. Due to the image nature, the formations may differ, but the conversion will not be changed.

Z-ordering formation 1:

Term 1: There is only one pixel found in the image
Term 2: Very few pixels found in the image

Z-ordering analysis of Term 1 and Term 2 (Dot):

Input: Character image
Output: Decimal values (features)

If the pixel is found in the sub-character image, then the following steps are followed for finding the features.

- The sub-character image is further divided into 64 equal blocks (8 × 8)—it is numbered as 0–63
- The three-order binary value (000–111) is assigned for each column and row as shown in Fig. 11
- Through the Z-ordering, the associated column and row which contain the pixel are considered, where the appropriate bit values are taken for finding the orders.
- Selected bit values are arranged in column- and row-wise order
- As discussed, the bit shuffling procedure has been enforced to take out the locational order of the pixel values in a decimal form.
- If there is only one decimal value being found, then it is referred as a dot or short line
- If there are two are three decimal values found, then it is referred as a small curve or short line.
- These decimal values are considered for features. The features extracted from very few pixels are shown in Figs. 13 and 14

Z-ordering formation 2:

Term 1: Closed curve is found in the image
Term 2: Long curvy shape is found in the image
Term 3: Any unshaped curve found in the image

Z-ordering analysis of Term 1 (closed curve):

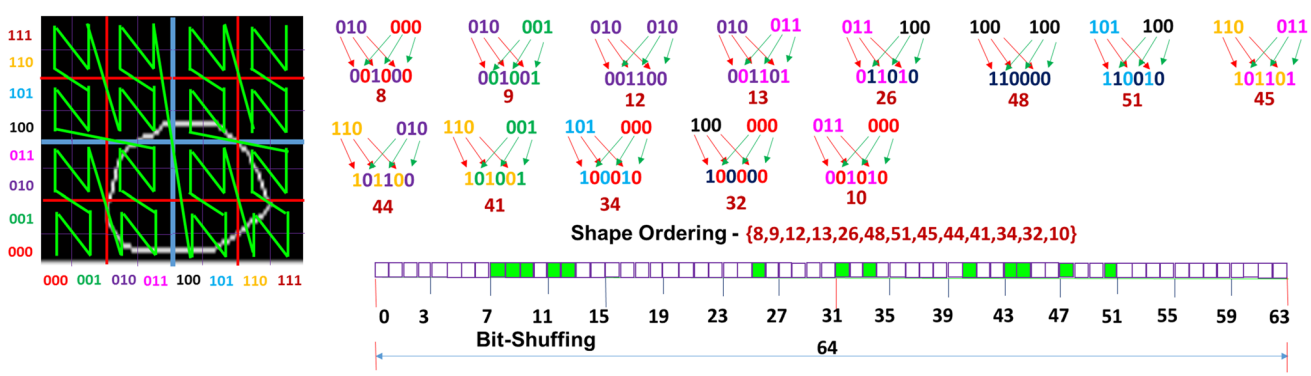


Fig. 15 Z-order representation for closed curve

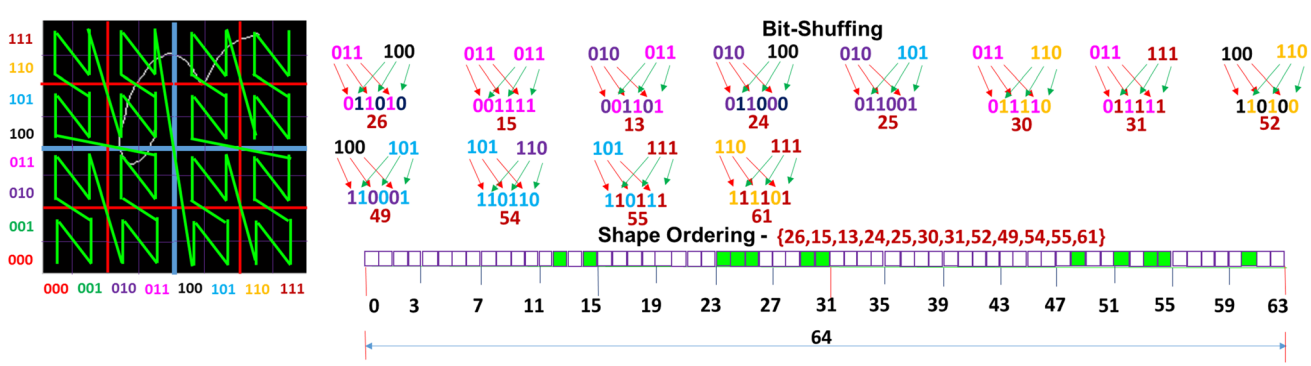


Fig. 16 Z-order representation for unshaped curve

Input: Character image
Output: Decimal values (features)

- Sixty-four equal blocks (8 × 8) have been created by assigning 8 × 8 grids on the sub-character image—name each grid as 0–63
- The three-order binary values (000–111) are allotted to take out the grid representation
- As discussed earlier, the decimal values are found out by the Z-order formation and considered as features

The start and end points returned by the Z-ordering are adjacent values if it is a closed curve (shown in Fig. 15).

Z-ordering analysis of Term 2 and Term 3 (curve or unshaped):

The same steps that are followed in the analysis of Term 1 of formation 2 have been used to extract the feature values in decimal form. Here the first and last values might not be adjacent values. Feature values may be in the order of the shape. Figure 16 shows the representations of unshaped feature extraction from the character image.

The orders extracted by Z-ordering would be highly related to the writing style, but it will slightly vary when considering varied shapes of the same character. Variations will be high when considering abnormal writing, but the order will be in the same form. In the analysis level, extraction is one of the toughest tasks when considering characters of similar shapes. One level of analysis is based on character complexity, and another level of analysis is based on the similarity among the characters. Very small variation may lead to negative results. The feature extraction must address those variations in microlevel. The Z-ordering feature extraction will be addressed by matching with shapes and its order as well as its location-based order. This can be really suitable for characters in any shape and variation. There is no much difference between Z-ordering and quad tree for locating the pixel, but procedure for locating and picking is different. The Quad tree can locate the pixels in each quadrant. This initial identification helps in directing the research in the right path. The next section describes the details of quad tree procedure.

2.3.3 PM-Quad tree

PM-Quad tree is usually used for representing lines, curves and its dissimilar shapes. This representation could be either vertex-based or edge-based. The PM-Quad tree (Nibaran Das et al. 2015; Das et al. 2012; Antony Robert Raj and Abirami 2015b) concept is stimulated for addressing the curvy features. The main objective of a PM-Quad tree usage is to locate each and every pixel point of the given structures in a collective form. This could be performed based on the breaking of vertices and edges as shown in Fig. 17 ($X \rightarrow Z$, $Z \rightarrow Y$). The breaking points of line or curves and its location are represented in data structure form. The breaking points cannot be found in all the structures of character image. Hence, the edge and vertex representation is not utilized in this work. Instead the starting point and end point of the shape structures are observed and represented in a tree form. Here the tree growth cannot be extended deeper and deeper; instead, the leaf division process is stopped in the third leaf level. The main reason behind this is that the shape addressed here which is neither big nor very small. It is normalized in a 200 × 200 pixel level. This level of quad tree is sufficient for dealing with different kinds of handwritten shapes.

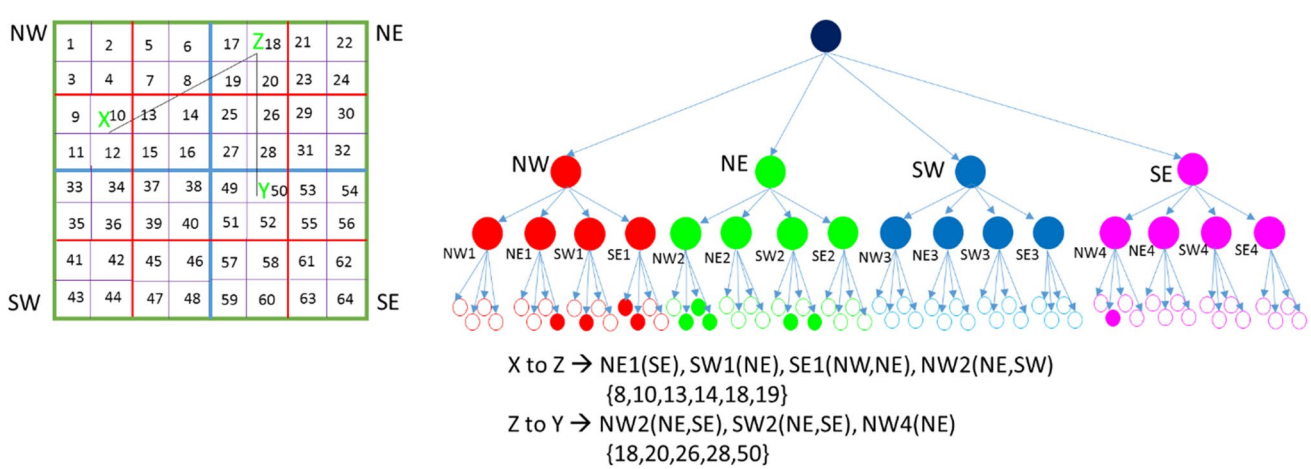


Fig. 17 PM-Quad tree—sample representation

In this experiment, the sub-character image has been divided into 64 equal grids (8 × 8 blocks). The following treatment takes place on the character structure of the image.

QUAD tree formation.

```
Begin
        If P is found /*if there is valid pixels are available in image */
        R ← Root; R ← Pointer /*Root_Node (R) ← Initial Pointer */
        L ← ϕ
S1:     If L is bottom level (ie. L←3)
                Return tree nodes (NWn, NEn, SWn, SEn)   /*n ← 1 to 3 */
        Else
                L = L+1
                Build Quad region (NW_ϕ+1, NE_ϕ+1, SW_ϕ+1, SE_ϕ+1)
                Repeat until L←3
End
Tree Length → 3
Tree Final Leaf → 64 [0 to 63]
```

Feature representation

```
Input   : Sub Character Image IM(X, Y);
Output  : Locational features in numeric values
Begin
        Initiate IM(X, Y);
                if (Bx Є IM(X, Y) /*if the foreground pixel is found in the image*/
                        do (neig(Bx) equal to 1) /*neighbor of the foreground pixel is valid  */
                                X ← neig (Bx) /*First foreground pixel with one neighbor*/
                        do (next_neig(Bx) equal to 1)
                                Y ← next_neig(Bx) /*Second  foreground pixel with one neighbor */
                        do (neig(Bx) equal to 2 and Pixel distance of the neig(Bx) ≤ 2)
                        /*representing breaking points */
                                Z  ← neig(Bx)
                                link1 ← [X to Z]
                                link2 ← [Z to Y]
                /* Location Identification */
                ∀ [X to Z] → if ([X to Z] Є IM(X, Y))
                /* for all Pixel values X to Y, if the values X to Y are found in the image*/
                /* Tree Location */
                Tree → Leaf_Node_in [0  ... 63] → Features
                ∀ [Z to Y] → if ([Z to Y] Є IM(X, Y))
                /* for all Pixel values Z to Y, if the values Z to Y are found in the image*/
                /* Tree Location */
Tree formation has been processed to extract the features
The location of the character portions are identified and taken out for features
Tree → Leaf_Node_in [0  ... 63] → Features
End
```

The image is logically divided into four equal grids, named North West (NW), North East (NE), South West (SW) and South East (SW). Each grid is further divided into four equal sub-grids like ((NW1, NE1, SW1 and SE1), (NW2, NE2, SW2 and SE2), (NW3, NE3, SW3 and SE3) and (NW4, NE4, SW4 and SE4)). Each sub-grid is again divided into four equal super-sub-grids, which afford 64 equal parts numbered 1–64 as shown in Fig. 19. Assume that there is a character portion (*X*, *Y*) on the image. If any

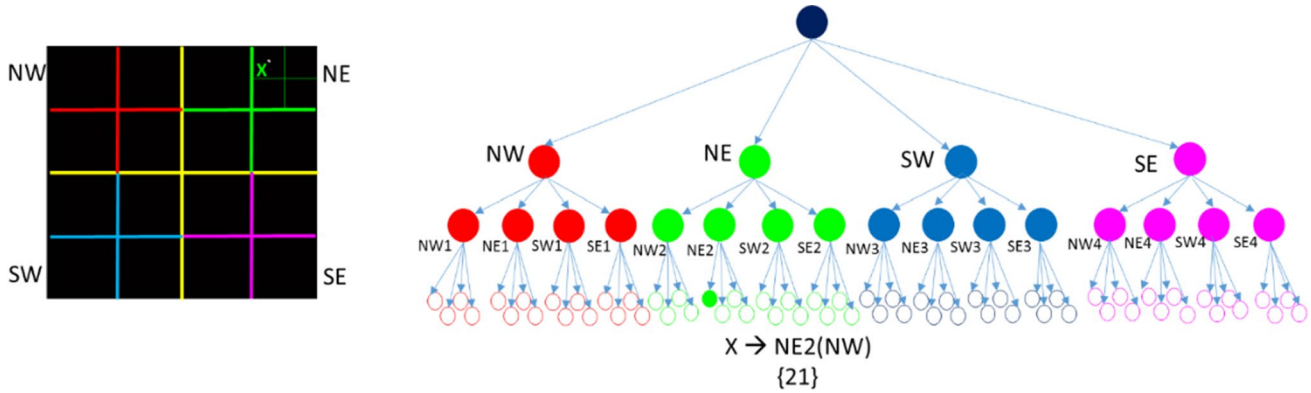


Fig. 18 PM-Quad tree representation for dot

breaking point Z exists between X and Y, then the structural locations of end points X and Y are identified based on the breaking point Z (as shown in Fig. 17). Located structures are remarked in the appropriate leaf nodes of the tree. The numeric identification of these leaf nodes are considered as features.

The following quad tree formation is used to address location of varied shapes of the given character's structures. Here the shape might be a dot, line or curve, uneven shape and closed curve. This has been addressed through few terms as follows

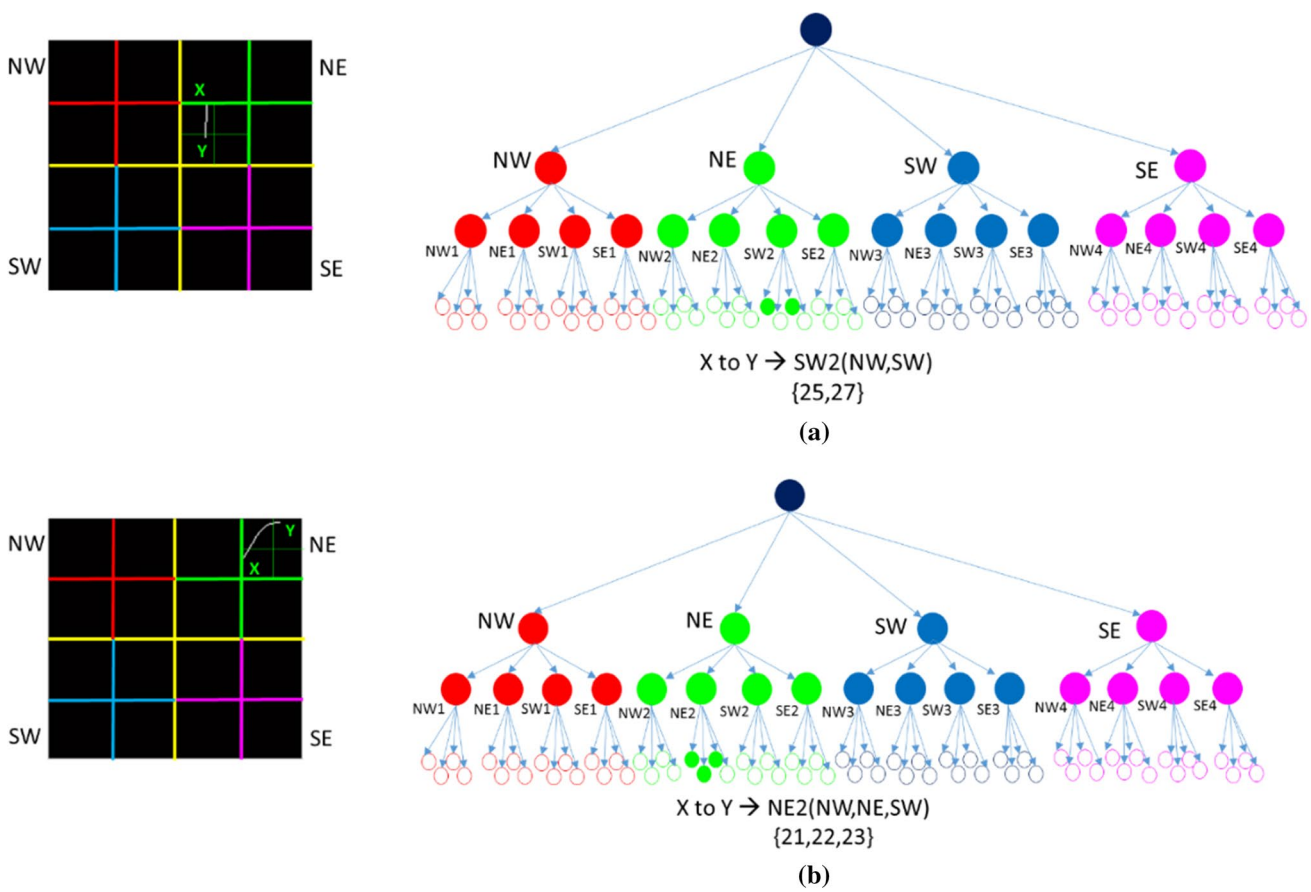


Fig. 19 **a** PM-Quad tree representation for short line. **b** PM-Quad tree representation for short curve

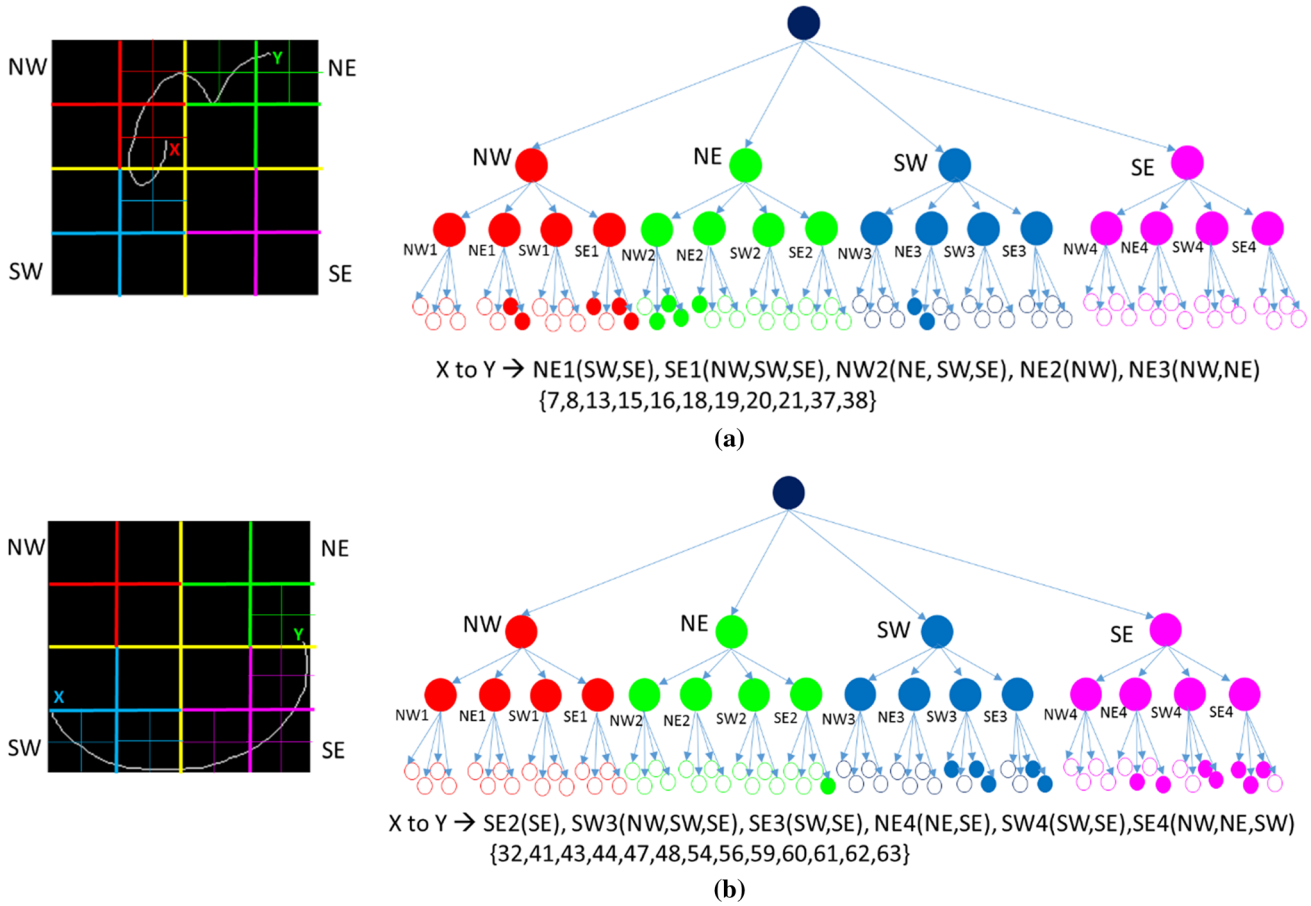


Fig. 20 **a**. PM-Quad tree representation for unshaped curve. **b** PM-Quad tree representation for curve

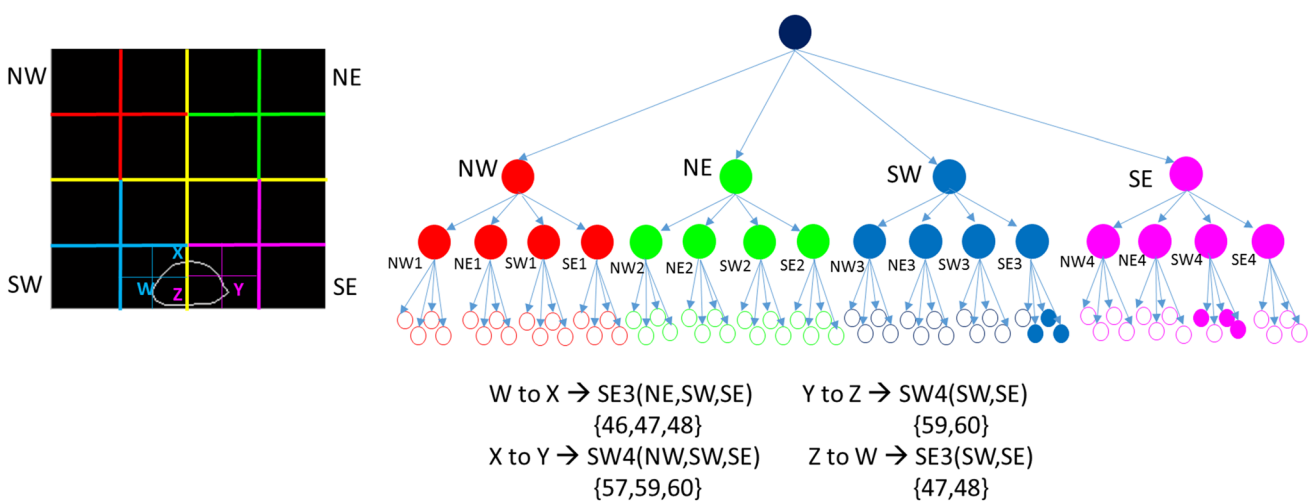


Fig. 21 PM-Quad tree representation for closed curve

PM Quad Tree Formation 1:

Term 1: There is only one pixel found in the image (dot)

Term 2: Very few pixels found in the image (might be short line or short curve)

PM Quad Tree Analysis of Term 1(Dot)

Input : Sub Character Image IM(X, Y);

Output : Locational features in numeric values

Begin

Initiate IM(X, Y);

if (Bx Є IM(X, Y)) */*if the foreground pixel is found in the image*/*

Tree formation steps are proceeded to for the tree as discussed

/* Location Identification */

∀ [X] → if ([X] Є NW(IM(Xn, Yn)))

/ for all pixel value X, if the values X is found in the image*/*

/* Tree Location */

Tree → Leaf_Node_in[NWn, NEn, SWn, SEn]

∃[NWn, NEn, SWn, SEn] → ∃[1, 2, 3, 4]→ Features

/ If there exists the foreground pixel in the Quad division → taken the location of the same as features*/*

Continued testing for NE, SW, SE

End

Description

Here, as discussed earlier the character sub-image grid (NW, NE, SW and SE) is divided into 42 sub-grids ((NW1, NE1, SW1 and SE1), (NW2, NE2, SW2 and SE2), (NW3, NE3, SW3 and SE3), (NW4, NE4, SW4 and SE4)). Each grid quadrant is checked to determine the availability of pixel (*X*). If available, sub-quadrant is divided into four equal

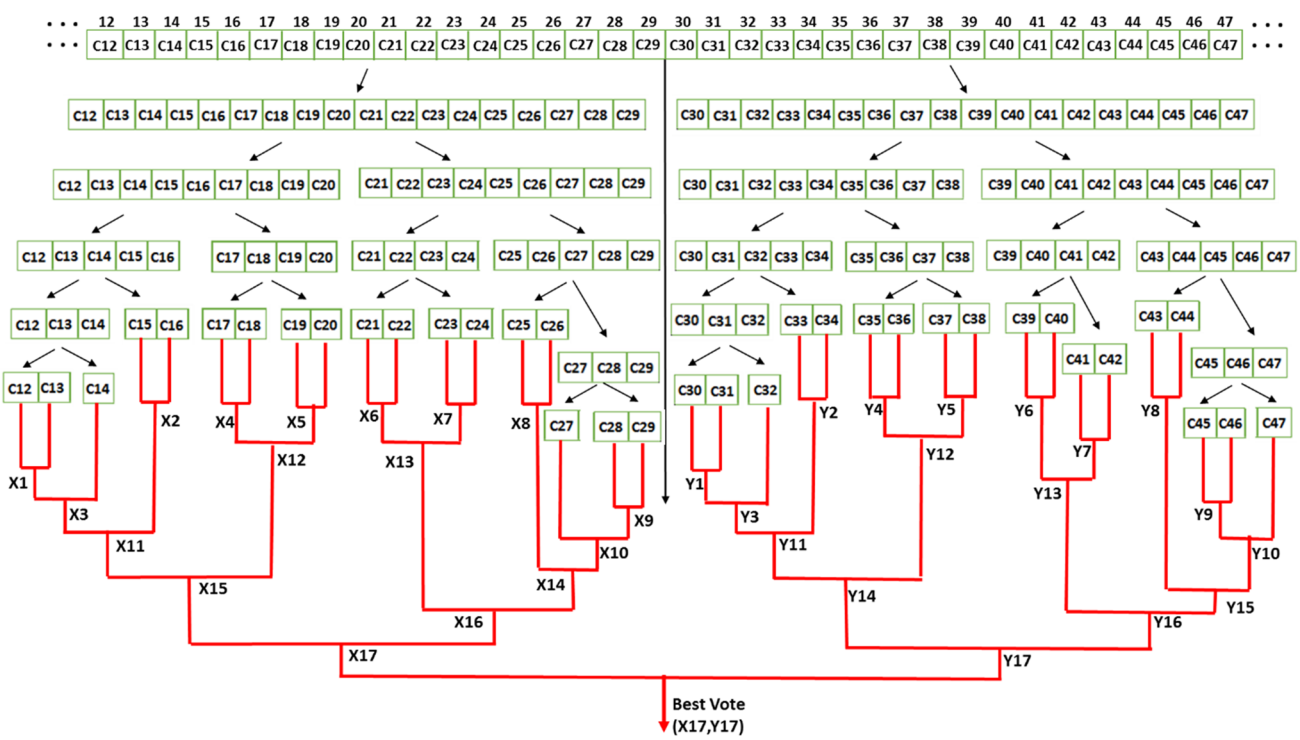


Fig. 22 Process flow of Hierarchical SVM

super-sub-quadrants. Each super-sub-quadrant is analyzed for finding the location of pixel. The pixel location is marked as super-sub-leaf in the quad tree. The numerical representation of the particular location of the quad tree will be the features. Figure 18 shows the dot representation in the quad tree and its numerical locational value

PM-Quad tree analysis of Term 2 (short line or curve)

Input: Sub-character image IM(X, Y);
Output: Locational features in numeric values.

Each grid is traversed to check whether any pixel is available. If available, then each pixel is examined to see whether it contains only one neighbor pixel. This examination might return two pixel points which contain one neighbor pixel each. The two pixel points are taken as end points (X and Y). Now the location of the given character structure is identified as discussed in the previous terms. The numerical representations of features are shown in Fig. 19a, b.

PM-Quad tree formation 2:

Term 1: Pixels found in the image with two end points (might be any shape)
Term 2: Pixels found with no end points (closed curve or shape).

PM-Quad tree analysis of Term 1(any shape)

The same procedure which is followed in analysis of previous term of formation is used here to identify the location of the image which is found between the end points X and Y. Here the location which is returned by the formation is high. Figure 20a, b shows the formation for a large pixel count.

PM-Quad tree analysis of Term 2 (closed curve):

As discussed, the sub-character image is treated for finding the pixels in image. In this formation, there is no possibility of pixels to contain only one neighbor. Here, the examination is focused to analyze the pixel which contains two neighbor pixels (denoted as a closed curve).

All pixel locations are divided by super-sub-quadrant grid procedure as discussed in the earlier formations. This closed circle is divided into four equal parts and the points (W, X, Y and Z) are noted. The same procedure which is followed in the analysis of Term 2 of formation 1 is used here to identify the location of the image found between the end points (W–X, X–Y, Y–Z and Z–W). The numerical values generated from each portion location are taken as features as discussed in Fig. 21.

Numerous samples were used here for the purpose of recognition. This type of feature extraction plays a very important role because the variation may change the shape of the character among all handwritten characters. Character features might be fitted into the location where it is placed. Few similar shape characters are also found in Tamil language. Among these characters, minor level of variations is found and can be addressed.

The whole set of feature values extracted as numerical values from all three procedures such as shape, shape ordering and location are used for classifying the character. The samples are collected and grouped as training and testing samples. Collected features from these samples are stored in separate excels. These excel details are fed into SVM algorithm which implemented in MATLAB to predict the results.

2.4 Classification

2.4.1 Hierarchical SVM

The features extracted from quad tree, strip tree and Z-ordering are discrete values. As SVM is highly suitable for discrete values, it has been selected. In this research, three classifiers, namely multilayer perceptron, modified quadratic discriminant function and SVM, are tested. Comparison results of all the three classifiers show that SVM is better than other two. Support vector machine is a very successful supervised learning algorithm (Benouareth et al. 2008; Nibaran Das et al. 2015; Das et al. 2012; Shanthi and Duraiswami 2007, 2010; Shyni et al. 2015; Antony Robert Raj and Abirami 2014, 2015d) that is used to predict results in pattern recognition environment based on function of linear discriminant. SVM can predict the features from sequential data. Here, discrete sequential information is gathered and given as input to the classifier. This classifier classifies the results by the function $W^t X+b$, where the weight factor 'W' and the bias 'b' are calculated by the training samples. The value of X varies based on the training sets. Here, two classes (two characters) are used: one in the positive side and the other in the negative side of the pattern X. That is, the term $W^t X+b<0$ for negative class and $W^t X+b>0$ positive class. The linear separator $W^t X+b=0$ is a hyperplane which is used for separating the two classes in the plane. $W^t X+b=0$ will be acting as the decision boundary for the positive class (+1) and the negative class (−1). SVM with radial basis function (RBF) kernel is used since the feature count is high.

In this work, practically experimented multiclass SVM is used for classifying the characters from the multiple character sets. This is possible by two ways, namely one-versus-one (OVO) and one-versus-all (OVA). OVO is the comparison of one class with each character one by one, and OVA is the comparison of one class with all characters at a stretch to choose the best matched character. In this work, OVA procedure is applied in a hierarchical manner and for implementing OVA for all classes, and divide-and-conquer method is utilized.

In Fig. 22, C series denote the classes, and X and Y series are the result of each one-vs-all decision. As per the

Table 1 Classification accuracy from shape and locational features

	Misclassified with	Total	Classified	Missed	%		Misclassified with	Total	Classified	Missed	%
அ	ஆ, ஓ	40	35	5	87.5	டீ	டி டீ டு டூ	40	35	5	87.5
ஆ	அ, இ, ஔ	40	33	7	82.5	ணீ	ணி	40	33	7	82.5
இ	ஆ, ஞி	40	37	3	92.5	தீ	தி ந றி று றீ	40	35	5	87.5
ஈ	ர,ரி,ரீ	40	32	8	80	நீ	நி	40	32	8	80
உ	ட	40	37	3	92.5	பீ	பி டீ டி	40	34	6	85
எ	ஏ, ர	40	32	8	80	மீ	மி மு மூ	40	32	8	80
ஏ	எ, ர	40	33	7	82.5	யீ	யி றி று றீ	40	35	5	87.5
ஐ	ஜ, ஞ	40	31	9	77.5	ரீ	ரி	40	36	4	90
ஒ	ஓ சூ	40	30	10	75	லீ	வி லீ லி	40	35	5	87.5
ஓ	ஒ ஔ	40	31	9	77.5	வீ	லீ, லி வ வி ல	40	36	4	90
க	ச த கி சூ கு	40	30	10	75	ழீ	ழி மி மீ மு மூ	40	36	4	90
ங	ந ஙி ஔ	40	33	7	82.5	ளீ	ளி ந ய ன ணீ ணி	40	34	6	85
ச	சு க கி கு	40	29	11	72.5	றீ	றி ந ற றி று	40	32	8	80
ஞ	கு கி சூ	40	32	8	80	னீ	னி ளி ளீ ந ய	40	34	6	85
ட	ப	40	37	3	92.5	கு	க கி சூ	40	32	8	80
ண	ணி ணூ ணு	40	36	4	90	ந	ங ஙி ஔ	40	34	6	85
த	ந ற றி றீ	40	32	8	80	சு	க கி சூ	40	32	8	80
ந	த ற	40	33	7	82.5	ஞ	சூ ஔ	40	33	7	82.5
ப	பு ம	40	33	7	82.5	டு	கு சூ டூ	40	35	5	87.5
ம	ய மி மீ	40	34	6	85	ணு	ணூ	40	34	6	85
ய	ம மி மு மூ	40	32	8	80	து	தூ சு கி கு	40	32	8	80
ர	ஈ	40	37	3	92.5	நு	நூ டூ	40	34	6	85
ல	வ வி ல லி லீ	40	30	10	75	பு	டீ டி	40	32	8	80
வ	வி ல லி லீ	40	28	12	70	மு	மூ டீ டி டு டூ	40	29	11	72.5
ழ	ம மி மீ மு மூ	40	35	5	87.5	யு	யூ ஞசூ	40	33	7	82.5
ள	ன ளி ளீ ந ன	40	34	6	85	ரு	ரூ த சூ கு	40	32	8	80
ற	ந த றி று றீ	40	32	8	80	லு	லூ ஞ ஞூ சூ	40	32	8	80
ன	ள ந ய ன	40	33	7	82.5	வு	வூ டி டு	40	35	5	87.5
கி	கீ தி தீ து	40	34	6	85	ழு	ழ ஙு ஞூசூ	40	36	4	90
ஙி	ஙீ லீ	40	36	4	90	ஞு	ஞ சூ ஙு ஞூ	40	33	7	82.5
சி	சீ தி தீ	40	31	9	77.5	று	றி ற	40	33	7	82.5
ஞி	ஞீ	40	34	6	85	னு	ளி ளீ ந ய ன	40	35	5	87.5
டி	டீ லீ	40	32	8	80	கூ	சூ ஙு ஞூ	40	37	3	92.5
ணி	ணீ ணூ ணு	40	34	6	85	நூ	ஞ ஞூ சூ	40	37	3	92.5
தி	தீ து	40	32	8	80	சூ	நூ சூ கு	40	32	8	80
நி	நீ தி தீ து	40	29	11	72.5	ஞூ	ஞிசூ	40	31	9	77.5
பி	டி வி லீ	40	30	10	75	டூ	டு கு	40	33	7	82.5
மி	ம ய ல ள யு	40	35	5	87.5	ணூ	ணு	40	34	6	85
யி	யீ மி மீ	40	32	8	80	தூ	து கி கு	40	32	8	80
ரி	ரீ	40	31	9	77.5	நூ	நு ற	40	33	7	82.5
லி	வி, லீ	40	29	11	72.5	பூ	பு டி	40	34	6	85
வி	லி, லீ	40	28	12	70	மூ	மு மி மீ	40	33	7	82.5
ழி	ழீ மி மீ மு மூ	40	28	12	70	யூ	ய ல ள யு	40	32	8	80
ளி	னி ளீ ந ய ன	40	32	8	80	ரூ	ஞீ ஙு ஞூசூ	40	34	6	85
றி	றீ ற ங	40	33	7	82.5	லூ	ஞீ ஙு சூ	40	32	8	80
னி	னீ ளி ளீ ந	40	31	9	77.5	வூ	வு ஙு ஞூ சூ	40	29	11	72.5
கீ	சீ தி தீ து	40	30	10	75	ஷ	ழி ழ	40	35	5	87.5
ஙீ	நூ ஞீ ஙு ஞூ	40	35	5	87.5	ஸ	ண, ணி	40	37	3	92.5
சீ	கீ தி தீ து	40	30	10	75	ஹ	ணூ ணி ணு	40	37	3	92.5
ஞீ	நூ ஙு ஞூ சூ	40	31	9	77.5	ஜ	ஜ ற றி ங	40	32	8	80

Table 2 Pros and cons of shape and locational features

S. nos.	Pros	Cons
1	Location and shape can address regular shape of the character	Shape or location alone may not be sufficient to address more characters
2	Locational complexity level is low	Shape variation may lead to negative results
3	Shape complexity level is moderate	Over all complexity level is high
4	Good for minimal set of shapes	Need more attention on shape

divide-and-conquer procedure, the total number of classes N is separated into two groups obtained by $N/2$. Let $N1$ and $N2$ be the number of classes available in each group. Each group is divided into subgroups ($N1/2$, $N2/2$) further to super-subgroups. This division comes to an end until all groups contain only two classes. In the middle of the procedure, if odd number of classes is found in any groups, then the groups are subdivided in such a way that one group contain one class and the other contains two classes. One-versus-all predication will be on the two classes first; then, result of this will be compared with the single class group.

Result is predicted between the members of final subgroup. The classification divisions are combined subgroup by subgroup in reverse manner as divided. In OVA prediction, the resultant classes will be obtained by receiving more number of votes among the classes. Finally the test result will be given to the one who is voted as the best result of $N1$ and $N2$. Final result is predicted from $N1$ and $N2$ groups. Total $(N-1)$ iterations are completed in all predictions, where N is number of classes. Total iteration that starts from even group will face seven comparisons, but the iteration that starts from an odd group will face only six comparisons. The analysis and results show that this algorithm predicts the result accurately.

3 Results and discussion

3.1 Data collection

The data are collected from HP-India handwritten Tamil character dataset (HP-India 2013), and handwritings of people in various age groups are considered. Hundred Tamil characters are chosen for testing in this work. 100 samples collected for each of the 100 characters. Therefore, a total of 10,000 characters are used for this experiment, where 6000 characters (60 from each 100) are chosen as training samples and 4000 characters (40 from each 100) are chosen as testing samples.

The experiment is practically implemented using MATLAB, and the features are extracted numerically. Shape and locational features are extracted and stored in an excel file. This is given first as input for classification. Further, along with locational features the shape ordering features are combined and given as input. Finally features extracted from all the three algorithms are combined in an excel format and given as input to the classifier. For further advancement the result of each combination has been analyzed. Further section discusses analysis of result of each phase.

3.2 Features organization

Feature counts are not stable in ST; hence, it limits to 64 node divisions maximum. Test features are extracted from 64 grid divisions; hence, maximum 64 features are extracted from quad and Z-ordering approaches. Moreover, the SVM is highly supported for discrete features. The following sequence of the feature organization has taken place

- $(ST_1, ST_2, \ldots, ST_{64})_U (ZO_1, ZO_2, \ldots, ZO_{64})$
- $(ZO_1, ZO_2, \ldots, ZO_{64})_U (QT_1, QT_2, \ldots,)$
- $(ST_1, ST_2, \ldots, ST_{64})_U (ZO_1, ZO_2, \ldots, ZO_{64})_U (QT_1, QT_2, \ldots, QT_{64})$

where ST: ST values; ZO: Z-ordering values; QT: quad tree values; and 1 to $n \rightarrow$ number of components extracted from junction points

3.3 Shape and locational features

Locational features are considered in all the phases of this testing. The main reason behind this is that locational features are essential to approximate the location of each character portion when variations are high. In the first iteration, the ST-based shape analysis has been combined with PM-Quad tree-based locational features to achieve good results. Table 1 reveals the results obtained from the shape and locational features.

The average result achieved from shape and locational features is 82.4%. This result is not enough to address more shapes because some misclassification has been found during the classification. Misclassification might happen when shapes are closely related to others characters. The process is continued to the next level of procedure for testing. The pros and cons of this algorithm are discussed in Table 2.

Table 3 Classification accuracy from Z-ordering and locational features

	Misclassified with	Total	Classified	Missed	%			Misclassified with	Total	Classified	Missed	%
அ	ஆ	40	37	3	92.5		டீ	டி டீ	40	35	5	87.5
ஆ	அ, இ,	40	36	4	90		ணீ	ணி	40	33	7	82.5
இ	ஆ, ஞி	40	38	2	95		தீ	தி றிறீ	40	37	3	92.5
ஈ	ர,ரி	40	37	3	92.5		நீ	நி	40	32	8	80
உ	ட	40	37	3	92.5		பீ	பி டீ டி	40	34	6	85
எ	ஏ, ர	40	34	6	85		மீ	மி மு மூ	40	34	6	85
ஏ	எ, ர	40	35	5	87.5		யீ	யி றி று றீ	40	35	5	87.5
ஐ	ஜ, ஞ	40	34	6	85		ரீ	ரி	40	37	3	92.5
ஒ	ஓ சூ	40	33	7	82.5		லீ	வி லீ லி	40	33	7	82.5
ஓ	ஒ சூ	40	34	6	85		வீ	லீ, லி வ வி ல	40	35	5	87.5
க	ச த கி	40	35	5	87.5		ழீ	ழி மி மீ	40	37	3	92.5
ங	ங ஙி	40	37	3	92.5		ளீ	ளி னீ னி	40	37	3	92.5
ச	சு க கி	40	33	7	82.5		றீ	றி ந ற றி	40	36	4	90
ஞ	கு கி சூ	40	35	5	87.5		னீ	னி ளி ளீ	40	37	3	92.5
ட	ப	40	38	2	95		கு	க கி சூ	40	35	5	87.5
ண	ணி ணு	40	38	2	95		ங	ங ஙி வூ	40	34	6	85
த	ந ற	40	36	4	90		சு	க கி சூ	40	37	3	92.5
ந	த ற	40	35	5	87.5		ஞு	சூ வூ	40	32	8	80
ப	பு	40	38	2	95		டு	கு சூ டு	40	36	4	90
ம	ய மி மீ	40	36	4	90		ணு	ணு	40	36	4	90
ய	ல	40	37	3	92.5		து	தூ சு கி கு	40	35	5	87.5
ர	ஈ	40	32	8	80		நு	நூ டு	40	34	6	85
ல	வ வி ல லி லீ	40	33	7	82.5		பு	டீ டி	40	36	4	90
வ	வி ல லி லீ	40	31	9	77.5		மு	மூ டீ டி டு டு	40	34	6	85
ழ	ம மி மீ மு மூ	40	37	3	92.5		யு	யூ ஞசூ	40	34	6	85
ள	ன ளி ளீ ங னீ	40	36	4	90		ரு	ரூ த சூ கு	40	35	5	87.5
ற	ந த றி று றீ	40	35	5	87.5		லு	லூ ஞ ஞூ சூ	40	37	3	92.5
ன	ள	40	37	3	92.5		வு	வூ டி டு	40	33	7	82.5
கி	கீ தி தீ து	40	33	7	82.5		ழு	ழ ஞு ஞூசூ	40	35	5	87.5
ஙி	ஙீ லீ	40	37	3	92.5		ளு	ஞ சூ ஞு ஞூசூ	40	38	2	95
சி	சீ தி தீ	40	33	7	82.5		று	றி ற	40	38	2	95
ஞி	ஞீ	40	35	5	87.5		னு	ளி ளீ ங யு னீ	40	36	4	90
டி	டீ லீ	40	37	3	92.5		கூ	சூ ஞு ஞூ	40	34	6	85
ணி	ணீ	40	38	2	95		ஙூ	ஞ ஞூ சூ	40	35	5	87.5
தி	தீ து	40	36	4	90		சூ	ஙூ சூ கு	40	34	6	85
நி	நீ தி தீ து	40	34	6	85		ஞூ	ஞிசூ	40	33	7	82.5
பி	டி வி லீ	40	35	5	87.5		டூ	டு கு	40	34	6	85
மி	ம ய	40	37	3	92.5		ணூ	ணு	40	35	5	87.5
யி	யீ மி மீ	40	36	4	90		தூ	து கி கு	40	34	6	85
ரி	ரீ	40	36	4	90		நூ	நு ற	40	35	5	87.5
லி	வி, லீ	40	35	5	87.5		பூ	பு டி	40	37	3	92.5
வி	லி, லீ	40	34	6	85		மூ	மு மி மீ	40	36	4	90
ழி	ழீ மி மீ	40	36	4	90		யூ	ய ல யு	40	36	4	90
ளி	னி ளீ னீ	40	34	6	85		ரூ	ஞீ ஞு ஞூசூ	40	35	5	87.5
றி	றீ ற ங	40	35	5	87.5		லூ	ஞீ ஞு சூ	40	34	6	85
னி	னீ ளி ளீ ங	40	33	7	82.5		வூ	வு ஞு சூ	40	36	4	90
கீ	சீ தி தீ து	40	33	7	82.5		ஷ	ழி	40	39	1	97.5
ஙீ	ஙூ ஞீ ஞு ஞூ	40	35	5	87.5		ஸ	ண, ணி	40	38	2	95
சீ	கீ தி தீ து	40	35	5	87.5		ஹ	ணு	40	39	1	97.5
ஞீ	ஙூ ஞு ஞூ சூ	40	36	4	90		ஜ	ஜ ற றி ங	40	35	5	87.5

Table 4 Pros and cons of Z-ordering and locational features

S. nos.	Pros	Cons
1	Z-ordering addresses the writing nature	Addressing the shape is necessary to solve the structure complexity
2	Ordering and locational complexity level is low	Over all complexity level is moderate
3	Good for considerable set of characters	Location mismatch

3.4 Z-ordering and locational features

The shape ordering features are closely related to character's traveling nature. In order to achieve better results, the locational features are experimented with the shape ordering. The final result and pros and cons of this are discussed in Tables 3 and 4.

The average result shows that our experiment is successful when various shapes of numerous characters are considered. The average result achieved from this experiment is 88.3%. When the shape complexity is increased hugely, this experiment leads to a challenge in the results. In order to solve this issue, the experiment is moved further to test all features together.

3.5 Combinational of all three procedures

There is a unique pattern in each feature. To combine the benefits of all features, this experiment has taken place. Few changes happened during this experiment. The final results show slight improvement as shown in Table 5. The overall results achieved from this combined experiment are 90.3%. Table 6 describes the advantages and disadvantages of the combined approach which supports further continuation of this research.

Ten characters are chosen from Tamil vowel for this experiment. The training and testing samples for these ten characters are implemented separately for the purpose of testing. The confusion matrix for the SVM over the ten characters is shown in Table 7. The average results achieved for vowels alone are 95%.

The same experiment was continued for all testing samples of the Tamil consonants (18 characters). The average result achieved from this experiment is 94.7%. Confusion matrix for the same is shown in Table 8.

Few other considerable results are compared with our achievements (Table 9), where 100 characters with 40 variations are successfully tested in this work. Most of the works that come under Tamil character recognition systems are based on statistical concepts. This work concentrated more on the character's structure. From the result of this experiment, it is explicit that this work has proved that structural representation can address all the shape of handwritten characters, but to improve results statistical concepts are essential.

A variation ஒ was misclassified with ஐ in the initial stage, but it has been classified by other statistical approaches. Likewise ஒ which looks like the shape ர has been misclassified when applying statistical approaches, but the same has been classified by shape-based approaches. A curve misleads the character ம (shown in Table 10) during the process of shape-based approach and the same has been eliminated by location-based approaches. The shape of ள looks like ன which was misclassified in all the stages. The shape ள was misclassified with ழ when applying the shape-based approach and the same is solved by the location-based approach. The style of writing caused the misclassification of ளி character, but in few cases there was no solution found in any of the approaches. Table 10 shows few misclassified samples when applied to all combination of algorithmic approaches.

4 Conclusion

This experiment is highly based on the features exaction techniques. General and suitable preprocessing techniques are used to get the needed image for applying the feature selection and extractions. The junction point procedure is applied to select the needed character's structure without affecting real shape. For feature extraction, three ways of feature representation techniques are used. Initially, strip tree-based shape discovering features have been implemented by rectangle formation and the same is represented in tree. In addition, location of the character's structures is extracted by two ways PM-Quad tree and Z-ordering techniques where PM-Quad tree locates the location of shapes which are represented in tree. And Z-ordering technique is implemented to get structure order and its location where bit shuffling approach is used to extract these features. These three different feature combinations are implemented practically in MATLAB. Each character sample is unique and is highly user dependent. Here the shape ordering plays an important role in completing the task. SVM is used in hierarchical manner with divide-and-conquer approach to classify the correct character obtained from those three ways of features. As per our final result analysis, this combination can address all unique 100 characters, but it will be challenging when the level of complexity is high. This algorithm is designed for recognizing the characters which has curvy

Table 5 Classification accuracy from all three procedures

	Misclassified with	Total	Classified	Missed	%		Misclassified with	Total	Classified	Missed	%
அ	ஆ	40	37	3	92.5	டீ	டி டீ	40	35	5	87.5
ஆ	அ, இ,	40	37	3	92.5	ணீ	ணி	40	37	3	92.5
இ	ஆ, ஞி	40	38	2	95	தீ	தி றீ	40	37	3	92.5
ஈ	ர,ரி	40	37	3	92.5	நீ	நி	40	36	4	90
உ	ட	40	35	5	87.5	பீ	பி டீ	40	38	2	95
எ	ஏ, ர	40	36	4	90	மீ	மி மு மூ	40	36	4	90
ஏ	எ, ர	40	35	5	87.5	யீ	யி றி றீ	40	35	5	87.5
ஐ	ஜ, ஞு	40	35	5	87.5	ரீ	ரி	40	37	3	92.5
ஒ	ஓ சூ	40	34	6	85	லீ	வி லீ லி	40	35	5	87.5
ஓ	ஒ சூ	40	37	3	92.5	வீ	லீ, லி வ வி ல	40	35	5	87.5
க	ச த கி	40	35	5	87.5	ழீ	ழி	40	38	2	95
ங	ங ஙி	40	37	3	92.5	ளீ	ளீ ளி	40	38	2	95
ச	சு க கி	40	36	4	90	றீ	றி ந ற	40	36	4	90
ஞ	கு கி சூ	40	35	5	87.5	னீ	னி ளீ	40	37	3	92.5
ட	ப	40	38	2	95	கு	க கி சூ	40	35	5	87.5
ண	ணி ணு	40	38	2	95	ங	ங ஙி ஷ	40	36	4	90
த	ந ற	40	36	4	90	சு	க சூ	40	37	3	92.5
ந	த ற	40	35	5	87.5	ஞு	சூ ஷ	40	34	6	85
ப	பு	40	38	2	95	டு	கு சூ டு	40	36	4	90
ம	ய மி மீ	40	35	5	87.5	ணு	ணூ	40	36	4	90
ய	ல	40	37	3	92.5	து	தூ கு	40	35	5	87.5
ர	ஈ	40	36	4	90	நு	நூ டு	40	35	5	87.5
ல	வ வி லி	40	36	4	90	பு	டீ டி	40	36	4	90
வ	வி ல லி லீ	40	34	6	85	மு	மூ டு டு	40	34	6	85
ழ	மு மூ	40	37	3	92.5	யு	யூ ஞசூ	40	34	6	85
ள	ன ளி னீ	40	36	4	90	ரு	ரூ த சூ கு	40	35	5	87.5
ற	ந த றி று	40	35	5	87.5	லு	லூ ஞ	40	37	3	92.5
ன	ள	40	35	5	87.5	வு	ஷ டி டு	40	36	4	90
கி	கீ தி தீ	40	36	4	90	ழு	ழ ஞு ஞூசூ	40	37	3	92.5
ஙி	ஙீ லீ	40	37	3	92.5	ஞு	ஞ	40	38	2	95
சி	சீ தி தீ	40	34	6	85	று	றி ற	40	38	2	95
ஞி	ஞீ	40	35	5	87.5	ணு	ளி	40	36	4	90
டி	டீ லீ	40	37	3	92.5	கூ	சூ	40	37	3	92.5
ணி	ணீ	40	38	2	95	ஙூ	ஞ ஞூ சூ	40	35	5	87.5
தி	தீ து	40	36	4	90	சூ	ஙூ சூ கு	40	36	4	90
நி	நீ தி தீ து	40	34	6	85	ஞூ	ஞிசூ	40	36	4	90
பி	டி வி லீ	40	35	5	87.5	டூ	டு கு	40	34	6	85
மி	ம ய	40	37	3	92.5	ணூ	ணு	40	35	5	87.5
யி	யீ மி மீ	40	36	4	90	தூ	து கி கு	40	36	4	90
ரி	ரீ	40	36	4	90	நூ	நு ற	40	35	5	87.5
லி	வி, லீ	40	35	5	87.5	பூ	பு டி	40	37	3	92.5
வி	லி, லீ	40	36	4	90	மூ	மு மி மீ	40	36	4	90
ழி	ழீ மி மீ	40	36	4	90	யூ	ய ல யு	40	36	4	90
ளி	னி ளீ னீ	40	36	4	90	ரூ	ரீ சூ	40	37	3	92.5
றி	றீ ற ங	40	35	5	87.5	லூ	ரீ ஞு சூ	40	35	5	87.5
னி	னீ ளி	40	37	3	92.5	வூ	வு ஞு சூ	40	36	4	90
கீ	சீ தீ	40	37	3	92.5	ஷ	ழி	40	39	1	97.5
ஙீ	நூ ஞீ	40	37	3	92.5	ஸ	ண, ணி	40	38	2	95
சீ	கீ தீ	40	37	3	92.5	ஹ	ணு	40	39	1	97.5
ஞீ	ஙூ சூ	40	36	4	90	ஜ	ஜ	40	37	3	92.5

Table 6 Pros and cons of final analysis

S. no.	Pros	Cons
1	Addressing more characters	Writers complexity to be improved
2	More shape can be addressed	Location is not matching with writers' nature
3	Considered character complexity	Writer independent
4	Highly matching with character nature	
5	Character dependent	

Table 7 Confusion matrix for recognition result on Tamil vowels

Classes	அ	ஆ	இ	ஈ	உ	எ	ஏ	ஐ	ஒ	ஓ
அ	37	3	0	0	0	0	0	0	0	0
ஆ	2	37	1	0	0	0	0	0	0	0
இ	0	1	39	0	0	0	0	0	0	0
ஈ	0	0	0	40	0	0	0	0	0	0
உ	0	0	0	0	40	0	0	0	0	0
எ	0	0	0	0	0	37	3	0	0	0
ஏ	0	0	0	0	0	4	36	0	0	0
ஐ	0	0	0	0	0	0	0	40	0	0
ஒ	0	0	0	0	0	0	0	0	36	4
ஓ	0	0	0	0	0	0	0	0	2	38
Recognition rate	92.5	92.5	97.5	100	100	92.5	90	100	90	95

Table 8 Confusion matrix for recognition result on Tamil consonants

Classes	க	ங	ச	ஞ	ட	ண	த	ந	ப	ம	ய	ர	ல	வ	ழ	ள	ற	ன
க	36	0	2	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
ங	0	40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ச	2	0	38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ஞ	0	0	0	40	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ட	0	0	0	0	38	0	0	0	2	0	0	0	0	0	0	0	0	0
ண	0	0	0	0	0	40	0	0	0	0	0	0	0	0	0	0	0	0
த	0	0	0	0	0	0	36	3	0	0	0	0	0	0	0	0	1	0
ந	0	0	0	0	0	0	2	35	0	0	0	0	0	0	0	0	3	0
ப	0	0	0	0	0	0	0	0	40	0	0	0	0	0	0	0	0	0
ம	0	0	0	0	0	0	0	0	0	37	3	0	0	0	0	0	0	0
ய	0	0	0	0	0	0	0	0	0	0	37	0	3	0	0	0	0	0
ர	0	0	0	0	0	0	0	0	0	0	0	40	0	0	0	0	0	0
ல	0	0	0	0	0	0	0	0	0	0	0	0	38	2	0	0	0	0
வ	0	0	0	0	0	0	0	0	0	0	0	0	3	37	0	0	0	0
ழ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	40	0	0	0
ள	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	38	0	2
ற	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	37	0
ன	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	35
%	90	100	95	100	95	100	90	87.5	100	92.5	92.5	100	95	92.5	100	95	92.5	87.5

Table 9 Comparison on ours with few considerable works in Tamil

Paper ref	Feature extraction	Classification	Characters	Results
Sigappi et al. (2011)	Vertical and word profile	HMM	40 words	90%
Shanthi and Duraiswami (2007)	Zoning	SVM	34	82.4
Shanthi and Duraiswami (2010)	Zoning	SVM, NNC	10	93.9, 94.9
Rajashekararadhya et al. (2008)	ROI	SVM, HMM	–	92–94
Rajashekararadhya and Vanaja Ranjan (2008b)	Zoning	NNC	10	96
Proposed	Tree representation	Hierarchical SVM	100	90.3

Table 10 Samples misclassified in each algorithmic procedure

Handwritten shape	Real shape	Classified by ST and Z-order	Classified by Quad tree and Z-order	All approaches
	ஔ	Misclassified	Classified	Classified
	ஔ	Misclassified	Classified	Classified
	ஔ	Classified	Misclassified	Classified
	ஔ	Classified	Misclassified	Classified
	ற	Misclassified	Misclassified	Misclassified
	ம	Misclassified	Classified	Classified
	ர	Misclassified	Classified	Classified
	ய	Misclassified	Classified	Classified
	ழ	Misclassified	Misclassified	Misclassified
	றா	Classified	Misclassified	Classified
	எ	Misclassified	Misclassified	Misclassified
	எ	Misclassified	Classified	Classified
	தி	Misclassified	Misclassified	Misclassified
	னி	Classified	Misclassified	Classified
	னி	Misclassified	Misclassified	Misclassified

nature and capable to address all Tamil characters. It is suitable any language which has curvy nature by default or by writer dependent. This work is to be continued in future for attaining shape analysis and locational analysis.

Complience with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

Antony Robert Raj M, Abirami S (2012) A survey on tamil handwritten character recognition using OCR techniques. Second Int Conf Comput Sci Eng Appl (CCSEA) 05:115–127

Antony Robert Raj M, Abirami S (2013) Analysis of statistical feature extraction approaches used in Tamil handwritten. In: OCR 12th Tamil internet conference-INFITT, pp 144–150

Antony Robert Raj M, Abirami S (2014) Offline Tamil handwritten character recognition using chain code and zone based features. In: 13th Tamil internet conference-INFITT, pp 28–34

Antony Robert Raj M, Abirami S (2015a) Strip tree based offline Tamil handwritten character recognition. In: International conference on ICT for intelligent systems, Springer SIST 2190-3018, pp 367–374

Antony Robert Raj M, Abirami S (2015b) Offline Tamil handwritten character recognition using statistical features. AENSI J Adv Nat Appl Sci 9(6):367–374

Antony Robert Raj M, Abirami S (2015c) Hybrid features based offline Tamil handwritten character recognition. In: 14th International Tamil internet conference, vol 2313–4887, pp 360–370

Antony Robert Raj M, Abirami S (2015d) Offline Tamil handwritten character recognition using statistical based quad tree. Aust J Basic Appl Sci 10(2):103–109

Benouareth A, Ennaji A, Sellami M (2008) Semi-continuous HMMs with explicit state duration for unconstrained Arabic word modeling and recognition. Pattern Recognit Lett Elsevier 29:1742–1752

Bhattacharya U, Ghosh SK, Parui SK (2007) A two stage recognition scheme for handwritten tamil characters. Ninth Int Conf Doc Anal Recognit 1:511–515

Dana H, Ballard (1981) Strip trees: a hierarchical representation for curves. ACM, Gr Image Process 24:310–321. ISSN: 0001-0782

Das N, Reddy JM et al (2012) A statistical–topological feature combination for recognition of handwritten numerals. Appl Soft Comput Elsevier 12(2486–2495):54

Desai AA (2010) Gujarati handwritten numeral optical character reorganization through neural network. Pattern Recognit Elsevier 43:2582–2589

Dubey P, Sinthupinyo W (2010) New approach on structural feature extraction for character recognition. In: ISCIT, IEEE, pp 946–949

HP-India (2013) http://lipitk.sourceforge.net/datasets/tamilchardata.htm

Kamble PM, Hegadi RS (2015) Handwritten Marathi character recognition using R-HOG feature, Elsevier. Procedia Comput Sci 45:266–274

Lei LI, Li-liang ZHANG, Jing-fei SU (2012) Handwritten character recognition via direction string and nearest neighbor matching. J China Univ Posts Telecommun Elsevier 19(2):160–165

Liu C-L, Suen CY (2009) A new benchmark on there cognition of handwritten Bangla and Farsi numeral characters, Elsevier. Pattern Recognit 42:3287–3295

Mahmoud S (2008) Recognition of writer-independent off-line handwritten Arabic (Indian) numerals using hidden Markov models. Signal Process Elsevier 88:844–857

Nibaran Das AN, Sarkar R, Basu S et al (2015) Handwritten Bangla character recognition using a sot computing paradigm embedded in two pass approach. Pattern Recognit Elsevier 48:2054–2071

Poullot S, Buisson O, Crucianu M (2007) Z-grid-based probabilistic retrieval for scaling up content-based copy detection. In: Proceedings of the 6th ACM international conference on image and video retrieval, ACM, pp 348–355

Rajashekararadhya SV, Vanaja Ranjan P (2008a) Neural network based handwritten numeral recognition of Kannada and Telugu script. In: IEEE TENCON conference, pp 1–5

Rajashekararadhya SV, Vanaja Ranjan P (2008b) Efficient zone based feature extraction algorithm for handwritten numeral recognition of four popular South Indian scripts. Int J Theor Appl Inf Technol 4:1171–1181

Rajashekararadhya SV, Vanaja Ranjan P (2009) Zone-based hybrid feature extraction algorithm for handwritten numeral recognition of two popular Indian script. In: World congress on nature and biologically inspired computing, pp 526–530

Rajashekararadhya SV, Vanaja Ranjan P, Manhunath Aradhya VN (2008) Isolated handwritten Kannada and Tamil numeral recognition: a novel approach. In: First IEEE international conference on emerging trends in engineering and technology, pp 1192–1195

Roy PP, Bhunia AK, Das A, Dey P, Pal U (2016) HMM-based Indic handwritten word recognition using zone segmentation. Pattern Recognit 60:1057–1075

Sarkhel R, Das N, Saha AK, Nasipuri M (2016) A multi-objective approach towards cost effective isolated handwritten Bangla character and digit recognition. Pattern Recognit 58:172–189

Sarkhel R, Das N, Das A et al (2017) A multi-scale deep quad tree based feature extraction method for the recognition of isolated handwritten characters of popular indic scripts, Elsevier. Pattern Recognit 71:78–93

Shanthi N, Duraiswami K (2007) Performance comparison of different image size for recognizing unconstrained handwritten Tamil character using SVM. J Comput Sci 3(9):760–764

Shanthi N, Duraiswami K (2010) A novel SVM-based handwritten Tamil character recognition system. Springer Pattern Anal Appl 13(2):173–180

Shyni SM, Antony Robert Raj M, Abirami S (2015) Offline Tamil handwritten character recognition using sub line direction and bounding box techniques. Indian J Sci Technol 8(S7):110–116

Sigappi AN, Palanivel S, Ramalingam V (2011) Handwritten document retrieval system for Tamil language. Int J Comput Appl 31:42

Yang G, Liang H, Su Y (2013) Generating Chinese characters based on stroke splitting and feature extraction. Displays 34(4):258–269

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.