Chapter 6
FEATURE EXTRACTION
Chapter 6

Feature Extraction

6.1 Introduction
6.2 Zone based symmetric density feature
6.3 Diagonal, Horizontal and Vertical features
6.4 Normalized chain code feature
6.5 Invariant moment feature
6.6 Zernike moment feature
6.7 Discrete wavelet transform

In this chapter different feature extraction techniques are elaborated. The feature extraction techniques are based on density, chain code, invariant moment, Zernike moments and wavelet transform. Also we have presented different algorithms to extract these features.

6.1 Introduction

Feature extraction is important phase in OCR prior to classification. A feature is a unique property that can describe image. The main objective of feature extraction is to reduce the size of image and represent the image object effectively in terms of a compact feature vector. Feature extraction takes image

Part of this chapter has been published in the
as input, builds initial data and finally gives feature values which are non-redundant and informative. Recognition accuracy of OCR largely depends on features extracted in this phase. In this phase unique characteristics (features) of an image are stored into feature vector for all input images which are further used for recognition purpose.

Assigning handwritten Marathi character to predefined classes is very difficult and challenging task due to interclass and intra-class similarities. Sufficient amount of work is reported for isolated handwritten Devanagari character recognition. Various feature extraction techniques like zone based symmetric density, zone based diagonal, horizontal, vertical, normalized chain code, moment invariant, Zernike moment and discrete wavelet transform are reported. The major advantage zone based symmetric density, diagonal, horizontal and vertical feature approach is that it is robust to small variations, easy to implement and yields relatively high recognition rate. Many authors have presented zoning mechanisms or regional decomposition methods to investigate the recognition rates of patterns. Normalized chain code has several advantages like it has compact representation and also translation invariant. Moment invariant and Zernike moments are very important features they are rotation invariant, independent of variability involved in the writing style of different individuals and also thinning free. In the next sections we will elaborate all the feature extraction techniques used in the present work.

### 6.2 Zone based symmetric density feature:

In this feature extraction technique hybrid zone based symmetric density features are extracted. For correct classification of handwritten characters suitable features should be extracted which are invariant with respect to shape. The objective of this hybrid approach came from its robustness to small variation, easy implementation and promising recognition accuracy. Zone based feature
extraction method gives good recognition accuracy even when certain preprocessing steps like filtering, smoothing and slant corrections are not performed. In this section, we elaborate on this feature extraction technique and the algorithm.

6.2.1 Review of earlier work:

Ashoka H. N. et. al. (2012) reported zone based feature extraction methods for handwritten numeral recognition and achieved 100% recognition accuracy. B.V. Dhandra et. al. (2011) reported zone based density feature for recognition of handwritten and printed Kannada and English numerals, and reported recognition accuracy for Kannada numerals 95.25% and for English numerals 97.05%. B.V. Dhandra and M. Hangarge [26] reported density and density ratio features as one of the features for identification of script at word level. B.V. Dhandra et. al. (2009, 2010) reported direction density estimation feature for Kannada, Telugu and Devanagari numeral recognition and achieved 99.40% recognition accuracy for Kannada numerals, 99.60% recognition accuracy for Telugu numerals and 98.40% recognition accuracy for Devanagari numerals. B.V. Dhandra et. al. (2011) reported directional density feature for Kannada numerals and achieved 98.04% recognition accuracy. Dinesh Acharya U. et. al. (2008) reported direction code frequency for horizontal and vertical blocks and achieved 92.68% recognition accuracy for printed Kannada characters. Mahesh Jangid (2011) reported pixel density and zone density features for Devanagari character recognition and achieved 94.89% recognition accuracy using SVM classifier. Vinaya Tapkir and Sushma Shelke (2012) reported pixel density feature for four zones and achieved a recognition accuracy of 92.77% for handwritten Marathi script. O.V. Ramana Murthy and M. Hanmandlu (2011) reported pixel density feature for recognition of Devanagari character recognition.
Chapter 6: Feature Extraction

It is observed from literature that for handwritten character recognition, density feature is largely used by researchers and obtained significant recognition accuracies. Structural features reflect the character’s structure information. Statistical feature is the most relevant information extracted from the raw data, which minimizes the inter-class distance and maximizes the between-class distance. Density statistical feature is commonly used for character recognition. Character’s structure feature method has a strong adaptability of character font changes, so it can easily differentiate between similar characters, but its computational complexity is large and its ability of anti-interference is bad. Character’s statistical feature has advantage of anti-interference and simple algorithm of classification and matching.

Hence, we have chosen zone based symmetric density feature for handwritten character recognition. The feature extraction method that was used to extract features and an algorithm is described in the following sections.

6.2.2 Feature extraction method:

To extract zone based symmetric density feature, the binary image representing the handwritten character is pre-processed and is normalized to a size of 60 x 60 pixels. The size-normalized image is divided into $n$ equal zones. The input image is divided into 4, 9, 16, 25 and 36 equal zones. For 4 equal zones, one zone has 30 x 30 pixels; for 9 equal zones, one zone has 20 x 20 pixels; for 16 equal zones, one zone has 15 x 15 pixels; for 25 equal zones, one zone has 12 x 12 pixels and for 36 equal zones, one zone has 10 x 10 pixels. Therefore, features are identified for $n$=4, 9, 16, 25 and 36 equal zones and they were stored in feature vector for each image.

The density of each zone is computed by taking the ratio of total number of object pixels to total number of pixels in that zone. This is carried out for every zone in the image. Finally, 90 features are extracted from the image and feature
vector stores these 90 features. Zone based symmetric density features were calculated for \( n = 4, 9, 16, 25, 36 \) are shown in Fig. 6.1 using Equation (1).

\[
\text{Density}(Z) = \frac{\text{number of object pixels in the zone } Z}{\text{total number of pixels in this zone } Z}
\]

\[
(1)
\]

Fig. 6.1: Character image divided into \( n \) zones and feature value for corresponding zone

### 6.2.3. Algorithm:

**Algorithm:** Zone based symmetric density feature extraction algorithm.

**Input:** Gray scale character image

**Output:** Feature vector of size 90.

1. Pre-process the input image and resize it to 60 x 60 standard plane.

2. Divide the input image into four equal zones; calculate the density of each zone that will give four features as shown in Fig. 6.1(a) & (b).
3. Divide the input image into nine equal zones; calculate the density of each zone that will give nine features as shown in Fig. 6.1(c) & (d).

4. Divide the input image into 16 equal zones; calculate the density of each zone that will give 16 features as shown in Fig. 6.1(e) & (f).

5. Divide the input image into 25 equal zones; calculate the density of each zone that will give 25 features as shown in Fig. 6.1(g) & (h).

6. Divide the input image into 36 equal zones; calculate the density of each zone that will give 36 features as shown in Fig. 6.1(i) & (j).

7. Store all features extracted in Step 2, 3, 4, 5 and 6 in feature vector. Finally feature vector containing 90 features for each image is ready for experimentation.

6.3 Diagonal, Horizontal and Vertical Features:

6.3.1 Review of earlier work:

J. Pradeep et.al.(2010) reported diagonal feature extraction method for handwritten character recognition and reported 99% recognition accuracy using 69 features. Om Prakash Sharma et. al. (2012) reported zone based diagonal features for handwritten Devanagari alphabets and obtained 98.50% recognition accuracy. It is observed that diagonal, horizontal and vertical features are having quite encouraging recognition results.

Zone based diagonal, horizontal and vertical features are statistical features and gives most relevant information from the raw data, which minimize the inner-class distance and maximize the between-class distance. Diagonal, horizontal and vertical feature methods have a strong adaptability of character font changes, so it can easily differentiate the similar characters and its
computational complexity is large. Character’s statistical feature has advantage of anti-interference and simple algorithm of classification and matching.

Hence we have decided to use a combination of diagonal, horizontal and vertical features. Feature extraction technique used to extract these features and algorithm is given in the next section.

6.3.2 Feature extraction method:

To extract diagonal, horizontal and vertical features the binary image representing the handwritten character is pre-processed and is normalized to a size of 50 x 50 pixels. The size-normalized image is divided into 25 equal zones where one zone has 10 x 10 pixels. The procedure to find diagonal, horizontal and vertical features is described below.

6.3.2.1 Diagonal Features:

To extract diagonal features from the binary image representing the handwritten character is preprocessed and is normalized to a size of 50 x 50 pixels. The size-normalized image is divided into 25 equal zones each of size is 10 x 10 pixels as shown in Fig. 6.2(a). Each zone has 19 diagonal lines, each diagonal line is summed to get a single sub-feature and thus 19 sub-features are obtained from the each zone as shown in Fig. 6.2(b).

These 19 sub-features values are averaged to form a single feature value and placed in the corresponding zone. This procedure is sequentially repeated for the all the zones as shown in Fig. 6.2(c).

Finally, 25 features are extracted for each character. In addition, 10 features are obtained by averaging the values placed in zones row-wise and column-wise, respectively. As a result; every character is represented by 25+10 features, that is, 35 features.
6.3.2.2 Horizontal Features:

To extract horizontal feature of the binary image representing the handwritten character is first preprocessed and is normalized to size of 50 x 50 pixels. The size-normalized image is divided into 25 equal zones, each zone is of size 10 x 10 as shown in Fig. 6.3(a). Each zone has 10 horizontal lines, each horizontal line is summed to get a single sub feature and thus 10 sub-features are obtained from the each zone as shown in Fig. 6.3(b).

These 10 sub-features values are averaged to form a single feature value and assigned as horizontal feature to the corresponding zone. This procedure is sequentially repeated for the all the zones. Finally, 25 features are extracted for 25 zones for each character as shown in Fig. 6.3(c). In addition, 10 features are obtained by averaging the values placed in zones row-wise and column-wise respectively. Finally every character is represented by 35 features, that is 25+10 features.
6.3.2.3 Vertical Features

To extract vertical features from the binary image representing the handwritten character is preprocessed and is normalized to a size of 50 x 50 pixels. The size-normalized image is divided into 25 equal zones; each zone of size is 10 x 10 as shown in Fig. 6.4(a). Each zone has 10 vertical lines, each vertical line is summed to get a single sub-feature and thus 10 sub-features are obtained from the each zone as shown in Fig. 6.4(b).

These 10 sub-features values are averaged to form a single feature value and placed in the corresponding zone. This procedure is sequentially repeated for the all the zones as shown in Fig. 6.4(c). Finally, 25 features are extracted for each character. As a result every character is represented by 25 features. In addition, 10 features are obtained by averaging the values placed in zones row-wise and column-wise, respectively. Finally every character is represented by 35 features, that is 25+10 features.
6.3.3 Algorithm:

**Algorithm:** Diagonal, Horizontal and Vertical feature extraction algorithm

**Input:** Gray scale character Image

**Output:** Diagonal, Horizontal and Vertical Features.

1. Pre-process the input Image and resize to 50 x 50 standard plane.

2. Divide the input image into 25 zones, each zone is of size 10 x 10 pixels.

3. Calculate diagonal feature value for each zone, repeat the process to find diagonal features for 25 zones.

4. Calculate average values for row-wise diagonal features and column-wise diagonal features, in all five row-wise and five column-wise values.

5. Feature vector of 35 diagonal features is prepared for each image.

6. Calculate horizontal feature value for each zone, repeat the process to find horizontal features for 25 zones.

7. Calculate average values for row-wise horizontal features and column-wise horizontal features, in all five row-wise and five column-wise values.
8. Feature vector of 35 horizontal features is prepared for each image.

9. Calculate vertical feature value for each zone, repeat the process to find vertical features for 25 zones.

10. Calculate average values for row-wise vertical features and column-wise vertical features, in all five row-wise and five column-wise values.

11. Feature vector of 35 vertical features is prepared for each image.

6.4 Normalized Chain Code:

Chain codes are the features which represents the boundary of a character. There are several advantages of using normalized chain code feature extraction listed below:

1. Compact representation of a character.
2. Feature values are not affected by translation of character.

6.4.1 Review of earlier work:

Aarti Desai et.al. (2011) reported chain code features for Devanagari character recognition. They have divided a character image into 25 blocks and for each block 8 chain code features are extracted, finally they have used 200 chain code features for recognition and achieved 87% recognition accuracy. Bikash Shaw et.al. (2008) reported directional chain code feature for handwritten Devanagari word recognition and achieved 80.2% recognition accuracy. G.G. Rajput and S.M. Mali (2010) reported freeman chain code features in combination with fourier descriptor for handwritten Marathi numeral recognition and achieved 98.1% recognition accuracy. Gunvantsinh Gohil et.al. (2012) reported chain code and holistic features for printed Devanagari script and achieved 66.35% and 80.55% using ANN and SVM classifier respectively. N. Sharma et.al. (2006)
Chapter 6: Feature Extraction

reported directional chain code feature extraction for 49 zones for handwritten Devanagari characters and obtained 98.86% and 80.36% on Devanagari numerals and characters respectively. Ravi Sheth et.al. (2011) reported normalized chain code feature extraction technique for handwritten English character recognition and obtained 92% recognition accuracy. S. Arora et.al. (2011) reported chain code feature extraction method in combination with shadow and view based features and obtained 98.61% recognition accuracy for handwritten Devanagari characters. S. Arora et.al. [122] reported zone based chain code histogram feature in combination with shadow features for recognition of non-compound handwritten Devanagari character and obtained 90.74% recognition accuracy.

It is observed from literature review that chain code directional features are having quite encouraging recognition results. Hence we have decided to use normalized chain code features for recognition of handwritten characters. Feature extraction technique used to extract these features and algorithm is elaborated in next sections.

6.4.2 Feature extraction method:

To extract freeman chain codes first locate any boundary pixel, called as starting pixel, and then move along the boundary of character either clockwise or anticlockwise direction, find out next boundary pixel and allocate this new pixel a number depending upon its direction from the previous pixel is called code for that pixel. The process is repeated till starting pixel is not encountered. The codes may be 4-directional or 8-directional depending upon 4-connectivity or 8-connectivity of a pixel to its neighboring contour pixel. An 8-directional chain coded image is given in Fig. 6.5.
Chapter 6: Feature Extraction

The chain code extracted from above process is different for different characters as length of each chain code depends on the size of the handwritten characters.

Example shows Chain code extracted for the image shown in Fig. 5.5.

Chain code: [0 7 6 6 0 6 4 3 4 5 4 2 2 0 2 0 2]

V1 = [0 7 6 6 0 6 4 3 4 5 4 2 2 0 2 0 2]

Compute the frequency of the codes 0, 1, 2, ......, 7. For vector V1 we have the frequency vector V2 as below.

V2 = [4 0 5 1 3 1 3 1]

The normalized frequency, represented by vector V3, is computed using the formula

\[ V3 = \frac{V2}{|V1|}, \text{ where } |V1| = \sum V2 \]

For the example considered above, we have

V3 = [0.22 0.27 0.05 0.16 0.05 0.16 0.05]

Finally, V3 is the required feature vector of size 8.

6.4.3 Algorithm:

Algorithm: Normalized chain code feature extraction algorithm

Input: Gray scale character image

Output: Normalized chain code feature vector for each image.

1. Pre-process the input Image and resize to 50 x 50 standard plane.
Chapter 6: Feature Extraction

2. Extract the boundary of the character image.

3. Resample the boundary in order to obtain a uniform resampling along the running arc length of the boundary.

4. Trace the boundary in counterclockwise direction and generate 8 directional chain codes 0 to 7.

5. Compute the frequency of the codes 0 to 7.

6. Divide frequency of each code by sum of the frequencies.

7. Store eight features in feature vector.

8. Finally feature vector of 8 features is ready for each input image.

6.5 Moment Invariant

Moment invariant features are based on statistical moments of characters. They are traditional and widely-used tool for character recognition. Classical moment invariants were introduced by Hu (1962) and they were successfully used in numerous applications not only for character recognition. Hu invariants are invariant under translation, rotation and scaling. Moment invariants features are extracted for the image which contributes to improve the overall recognition accuracy.

6.5.1 Review of earlier work:

Ajmire and Warkhede (2010) reported seven moment invariant features for handwritten Marathi vowel recognition. They have computed mean and standard deviation for each feature and these 14 features were used for recognition using Gaussian distribution function. S.V.Chavan et.al.(2013) reported geometric and Zernike moments for handwritten Devanagari compound character recognition and achieved 98.78% recognition accuracy using MLP.
classifier and 95.56% recognition accuracy using k-NN classifier. Nilima Patil et.al.(2011) reported moment invariant and affine moment invariant for handwritten Marathi vowel recognition and obtained 75% recognition accuracy. R. J. Ramteke (2010) reported invariant moment based feature extraction technique for handwritten Devanagari vowels recognition using 3 different feature sets by dividing image into four or two zones. R. J. Ramteke and S. C. Mehrotra (2008) reported invariant moment based feature extraction technique for handwritten Devanagari numerals recognition using 3 different feature sets by dividing image into four or two zones and achieved 92% recognition accuracy using 78 features. Reena Bajaj et.al.(2002) reported density and moment feature extraction technique for Devanagari numeral recognition and obtained 63.4% recognition accuracy for Devanagari numerals. S. Arora et.al.(2009) reported chain code histogram and moment based features for handwritten Devanagari character recognition and reported 98.03% recognition accuracy. S. M. Mali (2012) reported moment and density features for handwritten Marathi numeral recognition and reported 97.69% recognition accuracy.

It is observed from literature review that moment invariant features are having quite encouraging recognition results in case of handwritten characters. Hence we have decided to use moment invariant features for recognition of handwritten characters. Feature extraction technique used to extract these features and algorithm is elaborated in next sections.

6.5.2 Feature extraction Technique:

The method to calculate invariant moment is described below:

The two Dimensional moment of order (pique) of image is calculated as follows

\[ m_{pq} = \sum_{x=0}^{m-1} \sum_{y=0}^{n-1} x^p y^q f(x,y) \]
Chapter 6: Feature Extraction

Where \( p=0, 1, 2, \ldots \) and \( q=0, 1, 2, \ldots \)

Using 2D moments, central moment of order \( (p+q) \) can be calculated as follows

\[
\mu_{pq} = \sum_{x=0}^{m-1} \sum_{y=0}^{n-1} (x - \bar{x})^p (y - \bar{y})^q f(x, y)
\]

For \( p=0, 1, 2, \ldots \) and \( q=0, 1, 2, \ldots \) Where \( \bar{x} = \frac{m_{10}}{m_{00}} \) and \( \bar{y} = \frac{m_{01}}{m_{00}} \)

Normalized central moments can be derived by using above central moments as follows

\[
\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}}
\]

Where

\[
\gamma = \frac{p + q}{2} + 1
\]

For \( p+q=2, 3, \ldots \)

Set of seven invariant moments can be derived from second and third moments

\[
\phi_1 = \eta_{20} + \eta_{02}
\]

\[
\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2
\]

\[
\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2
\]

\[
\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2
\]

\[
\phi_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2]
\]

\[
+ (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]
\]

\[
\phi_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})
\]

\[
\phi_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2]
\]

\[
+ (3\eta_{12} - \eta_{30})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]
\]
6.5.3 Algorithm to compute moment invariant features:

Algorithm: Moment Invariant feature extraction algorithm.

Input: Gray scale character image.

Output: Moment invariant features for each image.

1. Pre-process the input Image and resize to 50 x 50 standard plane.
2. Compute seven moment invariant feature for the whole image and store into feature vector.
3. Divide character image into four equal zones, each zone of size 25 x 25 pixels.
4. Compute moment invariant feature for each zone. Add these 28 features into feature vector.
5. Feature vector of size 35 is ready for each image.

6.6 Zernike Moment:

6.6.1 Review of earlier work:

K. V. Kale et. al. (2014) reported Zernike moment feature extraction technique for handwritten Marathi compound character recognition. They had extraction zone based first 8 order Zernike moments and achieved 98.37% and 95.82% recognition accuracy using SVM and k-NN classifier.

The Zernike moment were first proposed in 1934 by Zernike. Zernike moments are complex numbers by which an image is mapped on to a set of two-dimensional complex Zernike polynomials. The magnitude of Zernike moments is used as a rotation invariant feature to represent a character image pattern. Zernike moments are a class of orthogonal moments and have been shown effective in terms of image representation. The orthogonal property of Zernike polynomials enables the contribution of each moment to be unique and independent of
information in an image. A Zernike moment does the mapping of an image onto a set of complex Zernike polynomials. These Zernike polynomials are orthogonal to each other and have characteristics to represent data with no redundancy and able to handle overlapping of information between the moments. Due to these characteristics, Zernike moments have been utilized as feature sets in applications such as pattern recognition and content-based image retrieval. These specific aspects and properties of Zernike moment are supposed to found to extract the features of handwritten characters. Feature extraction technique and algorithm to extract Zernike moments is elaborated in next sections.

6.6.2 Feature extraction method:

The Zernike moments introduce a set of complex polynomials which form a complete orthogonal set over the interior of a unit circle, i.e., \( x^2 + y^2 \leq 1 \). Zernike moments are the projection of the image function on some orthogonal basis functions. Let the set of these basis functions be denoted by \( V_{n,m}(x, y) \). These polynomials are defined by

\[
V_{n,m}(x, y) = V_{n,m}(\rho, \theta) = R_{n,m}(\rho)e^{ jm\rho}
\]  

(1)

where \( n \) is a non-negative integer, \( m \) is a non-zero integer subject to the following constrain: \( n - |m| \) is even and \( |m| < n \). Also, \( \rho \) is the length of the vector from origin to the \((x, y)\) pixel, \( \theta \) is the angle between vector \( \rho \) and x axis in a counterclockwise direction, and \( R_{n,m}(\rho) \) is the Zernike radial polynomial. The Zernike radial polynomials, \( R_{n,m}(\rho) \), are defined as :

\[
R_{n,m}(\rho) = \sum_{s=0}^{n-|m|} \frac{(-1)^s (n-s)!}{s! \left( \frac{n + |m|}{2} - s \right)! \left( \frac{n - |m|}{2} - s \right)!} \rho^{n-2s}
\]

Note that \( R_{n,m}(\rho) = R_{n,-m}(\rho) \). The Zernike moment of order \( n \) with repetition \( m \) for a digital image is
\[ Z_{n,m} = \frac{n + 1}{\pi} \sum_{x^2 + y^2 \leq 1} f(x, y) V_{n,m}^*(x, y) \Delta x \Delta y \]

where \( V_{n,m}^*(x, y) \) is the complex conjugate of \( V_{n,m}(x, y) \).

To compute the Zernike moments of a given image, the image center of mass is taken as the origin.

**Table 6.1: First eight order Zernike moments**

<table>
<thead>
<tr>
<th>Order</th>
<th>Dimension</th>
<th>Zernike moment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>( Z_{0,0} )</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>( Z_{1,1} )</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>( Z_{2,0}, Z_{2,0} )</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>( Z_{3,1}, Z_{3,3} )</td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>( Z_{4,0}, Z_{4,2}, Z_{4,4} )</td>
</tr>
<tr>
<td>5</td>
<td>12</td>
<td>( Z_{5,1}, Z_{5,3}, Z_{5,5} )</td>
</tr>
<tr>
<td>6</td>
<td>16</td>
<td>( Z_{6,0}, Z_{6,2}, Z_{6,4}, Z_{6,6} )</td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td>( Z_{7,1}, Z_{7,3}, Z_{7,5}, Z_{7,7} )</td>
</tr>
</tbody>
</table>

**6.6.3 Algorithm:**

**Algorithm:** Zernike moment feature extraction algorithm.

**Input:** Gray scale character image

**Output:** First eight orders of Zernike moment features for each image.

1. **Pre-process the input Image and resize to 50 x 50 standard plane.**
2. **Compute first eight orders Zernike moment feature for the whole image and store into feature vector.**
3. **Divide character image into four equal zones, each zone of size 25 x 25 pixels.**
4. Compute first eight orders Zernike moment feature for each zone and append into feature vector.

5. Feature vector containing first eight orders Zernike moment is ready for experiments.

6.7 **Discrete Wavelet Transform:**

6.7.1 **Review of earlier work:**

The Discrete Wavelet Transform (DWT) provides a decomposition of an image into details having different resolutions and orientations; it is a bijection from the image space onto the space of its coefficients. It has been mainly used for image compression. Diego J. Romero et.al. (2007) has reported directional continuous wavelet transformed for recognition of handwritten numerals. G.G.Rajput and Anita H. B. (2010) has reported discrete cosine transform and discrete wavelet transform for handwritten script recognition. Pritpal Singh and Sumit Budhiraja (2012) has reported wavelet transformation for handwritten Gurumukhi character recognition. Sushama Shelke and Shaila Apted (2010) has reported discrete wavelet transform for the recognition of handwritten Marathi compound character.

6.7.2 **Feature extraction method:**

Discrete wavelet transforms (DWT) are applied to discrete data sets and produce discrete outputs. Transforming signals and data vectors by DWT is a process that resembles the fast Fourier transform (FFT), the Fourier method applied to a set of discrete measurements. Discrete wavelet transforms map data from the time domain (the original or input data vector) to the wavelet domain. The result is a vector of the same size. Wavelet transforms are linear and they can be defined by matrices of dimension \( m \times n \) if they are applied to inputs
of size \( n \). Depending on boundary conditions, such matrices can be either orthogonal or "close" to orthogonal. When the matrix is orthogonal, the corresponding transform is a rotation in \( \mathbb{R}^n \) in which the data (a \( n \)-tuple) is a point in \( \mathbb{R}^n \). The coordinates of the point in the rotated space comprise the discrete wavelet transform of the original coordinates. The discrete wavelet transform (DWT) has a large number of applications in computer science. It is used for signal coding, to represent a discrete signal in a more redundant form, often as a preconditioning for data compression. Practical applications can also be found in signal processing of accelerations for gait analysis, in digital communications and many others.

It is shown that discrete wavelet transform (DWT) is discrete in scale and shift, and continuous in time. DWT is successfully implemented as analog filter bank in biomedical signal processing for design of low-power pacemakers and also in ultra-wideband (UWB) wireless communications. Wavelets are localized basis functions which are translated and dilated versions of some fixed mother wavelet. The decomposition of the image into different frequency bands is obtained by successive low-pass and high-pass filtering of the signal and down-sampling the coefficients after each filtering. Here various discrete wavelet transforms Daubechies is used.

DWT Single-level discrete 1-D wavelet transform. Single-level one-dimensional wavelet decomposition with respect to Daubechies wavelet transform is used. It performs a multilevel one-dimensional wavelet analysis using Daubechies wavelet and returns the wavelet decomposition of the signal. The output decomposition structure contains the wavelet decomposition vector \( C \) and the bookkeeping vector \( L \). Compute the approximation coefficients using the wavelet decomposition structure \([C, L]\).
The wavelet transform exhibits the features like separability, scalability, translatability, orthogonality and multiresolution capability. The discrete wavelet transform of an image $f(x, y)$ of size $M \times N$ is

$$W_\varphi(j_0, m, n) = \frac{1}{\sqrt{mn}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \varphi_{j_0,m,n}(x, y)$$

$$W_\psi(j, m, n) = \frac{1}{\sqrt{mn}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \psi_{j,m,n}(x, y)$$

Where,

$$\varphi_{j,m,n} = 2^j \varphi(2^j x - m, 2^j y - n)$$

And,

$$\psi_{j,m,n} = 2^j \psi(2^j x - m, 2^j y - n)$$

are the two dimensional scaling and wavelet functions respectively and the index $i$ identifies the directional wavelets that takes the values H, V and D i.e. horizontal, vertical and diagonal details respectively. $j_0$ an arbitrary starting scale and the $W_\varphi(j_0, m, n)$ coefficients define an approximation of $f(x, y)$ at scale $j_0$. The $W_\psi(j, m, n)$ coefficients add horizontal, vertical and diagonal details for scales $j \geq j_0$. Normally, $j_0 = 0$ $N = M = 2^j$ so that $j = 0, 1, 2..., J-1$ and $m, n = 0, 1, 2..., 2^{j-1}$. The discrete wavelet transform can be implemented using digital filters and down samplers. The high pass or detail component characterizes the image’s high-frequency information with vertical orientation; the low-pass, approximation component contains its low-frequency, vertical information. Both sub images are then filtered column wise and down sampled to yield four quarter size output images.
6.7.3 Algorithm:

Algorithm: Discrete wavelet transforms feature extraction algorithm.
Input: Gray scale character image
Output: Eight discrete wavelet transform features for each image.

1. Pre-process the input Image and resize to 50 x 50 standard planes.
2. Number of black pixels along each row of the binarized image has been counted to form a 50 sized vector.
3. The 1D discrete wavelet transform on row count vector at level 3 using Daubechies db1 wavelet has been applied.
4. Compute approximation coefficients and add to these four values to feature vector.
5. Number of black pixels along each column of the binarized image has been counted to form a 50 sized vector.
6. The 1D discrete wavelet transform on column count vector at level 3 using Daubechies-db1 wavelet has been applied.
7. Compute approximation coefficients and add to these four values to feature vector.
8. Feature vector containing eight discrete wavelet transformations is ready for experiments.

All these feature extraction methods are used to extract the features and the extracted features are further used for classification. In the next chapter SVM and k-NN classifiers are discussed. Also results are presented for SVM and k-NN classifier.