CHAPTER 5

COLOR TEXTURE IMAGE SEGMENTATION USING EXTENDED INTERVAL TYPE-2 FUZZY C-MEANS

Image segmentation is one of the most important and classical problems in image analysis. It should partition the image into disjoint regions according to some given features such as gray level, color or texture. The segmentation process can rely on the uniformity of the features within the regions or on edge information (discontinuities in the feature space). Through segmentation, the enhanced input image is mapped into a description involving regions with common features, which can be used by the high-level image analysis tasks. Segmentation techniques can be classified as classical approaches and fuzzy approaches.

5.1 INTRODUCTION

One of the very important tasks of image processing is the segmentation of images and to obtain meaningful information out of them. Machine vision systems are considered to have two subsystems: low-level vision and high-level vision. Low level vision consists primarily of image processing operations performed on the input image to produce an output image with much more favourable characteristics. Many of these operations cause certain features of the image to be emphasized. High-level vision includes object recognition and scene interpretation at a higher level. Segmentation bridges both the subsystems.
The goal of image segmentation is partitioning the image into meaningful regions which are used to further analyze the image. In computer vision and pattern recognition, color texture segmentation is one of the most important tasks since it plays an important role in the development of high-level image analysis such as object recognition, scene understanding, medical image analysis and remote sensing of images.

Many segmentation methods are based on two basic properties of the pixels in relation to their local neighborhood: discontinuity and similarity (Fu and Mui 1981). Methods based on pixel discontinuity are called boundary-based methods, whereas methods based on pixel similarity are called region-based methods. The segmentation techniques are also divided into thresholding, edge, region and clustering-based techniques. In other cases, the segmentation algorithms are divided into local or global, supervised or unsupervised methods.

5.1.1 Classical Image Segmentation Approaches

Thresholding becomes a simple but effective tool to separate objects from the background. The thresholding methods are categorized according to the information they are exploiting, such as histogram shape, measurement, space clustering, entropy, object attributes, spatial correlation and local gray-level surface. In thresholding based image segmentation method (Weszka 1978), every peak in the histogram corresponds to a class in the image. In order to obtain the segmentation, the image is thresholded to clearly separate those peaks. In some applications, histogram thresholding is not possible because the histogram may be unimodal (Bhanu and Faugeras 1982). Every peak in the histogram represents points of similar features. Therefore, histogram thresholding technique segments the uniform regions in an image. In this method, the segmentation is automatic, and there is no need to specify in advance the number of objects into which the image is to be
segmented. In case of images with a large number of small peaks, histogram smoothing is required for the algorithm to perform well.

Region-based segmentation technique (Zucker 1976) groups the pixels with similar characteristics into regions. There are two region-based segmentation techniques: region-growing and region-splitting. Region- growing techniques (Chen and Chen 2002), (Nammalwar 2010) group pixels into larger regions by the process of pixel aggregation. The process starts with the first pixel and merges the neighboring pixels to it, based on whether a certain condition is satisfied. It is an iterative process and the process ends when there are no more pixels to be classified. In region splitting methods (Shih and Chen 2005), (Ugarriza et al 2009), the evaluation of homogeneity is made on the basis of the large sets of image elements. The entire image acts as the seed to start with. Split and merge segmentation initially subdivides an image into a set of arbitrary, disjoint regions and then merges them iteratively to satisfy a given criterion. At each step, the image is split into four disjoint quadrants and then each quadrant is again split until all of the pixels belonging to that quadrant are homogenous. If the seed is inhomogeneous, it is split into predetermined number of sub-regions, typically four. When the splitting is completed, the neighboring regions are merged until a certain condition is satisfied. For each quadrant obtained at the end of the split process, its average gray level is calculated. The region-splitting methods are less sensitive to noise than region- growing methods. In both approaches, their iterative structures lead to computationally intensive algorithms.

Edges are local changes in the image intensity. It is a fundamental process to detect the outlines of an object and boundaries among objects and the background in the image. Edge detection is the most familiar approach for detecting significant discontinuities in intensity values. The edge-based
segmentation (Davis 1975), (Peli and Malah 1982) algorithms locate the boundaries of the objects or regions in the image. Edges are computed using differentiation between neighboring pixels. There are three different types of discontinuities in the gray level like point, line and edges. Spatial masks can be used to detect all the three types of discontinuities in an image. Edge-based segmentation must deal with the small gaps in the edges. Therefore, to find the final segmentation, edges must be joined using lines or any other curve. Finally, a set of features are computed (contrast, length, etc.) and used to remove the unwanted edges. For real applications, a pre-processing is used to smooth the image and remove the noise. Then in the post-processing step, edges are closed and the unwanted ones are removed to obtain the final segmentation.

Clustering (Anderberg 1973) is a process which classifies the objects or patterns so that samples of the same group are more similar to one another than samples belonging to different groups. By clustering, one can identify dense and sparse regions and therefore, discover overall distributed patterns and interesting correlations among data attributes. Many clustering strategies have been used, such as the K-Nearest Neighbor (Coomans and Massart 1982), K–means( Coleman et al 1979), PCM (Pal et al 2005), Type-1 FCM (Bezdek 1981) etc., and each of which has its own special characteristics. The conventional hard clustering method restricts each point of the data set to exclusively in just one cluster. As a consequence, with this approach the segmentation results are often very crisp, i.e. each pixel of the image belongs to exactly just one class. However, in many real situations for images, issues like limited spatial resolution, poor contrast, overlapping intensities, noise and intensity inhomogeneities variations make crisp segmentation a difficult task. Fuzzy set theory produced the idea of partial membership of belonging described by a membership function.
5.1.2 Fuzzy Logic-Based Image Segmentation Approaches

Conventional approaches to image analysis and recognition consist of segmenting the image into meaningful regions based on crisp decisions (i.e. 0 or 1), features, primitives and properties. Since the regions in an image are not always crisply defined, uncertainty can arise within every phase of the task. A recognition system should have sufficient provision for representing and manipulating the uncertainties involved at every processing stage.


Fuzzy models are capable to handle the ambiguity and noise in the images and hence can improve the selection of optimal threshold for better image segmentation. The fuzzy thresholding (Jawahar et al 2000) algorithms use membership functions to define fuzzy object regions and then select the one which is associated with the minimum value of the grayness or geometrical ambiguity measures. The object is enhanced from the background by the optimum membership function obtained. The optimal choice of the threshold is a difficult process due to the presence of noise, vagueness and ambiguity among the classes, produced by the overlapping among them in the histogram.

Fuzzy cluster analysis allows data points to have partial memberships to different clusters which are measured as degrees in [0, 1].
This yields to the flexibility that data points can belong to more than one cluster. The fuzzy clustering algorithm uses an iterative optimization of an objective function based on weighted similarity measure between the pixels in the image and each of the ‘c’ cluster centers. Fuzzy clustering describes an image in terms of fuzzy classes, which means each pixel in an image is assigned a membership value into class clusters using a fuzzy function. Each class can consist of many disjoint segments depending on the nature of the image. A well-known clustering method is fuzzy c-means clustering algorithm.

Fuzzy rule-based segmentation (Karmakar and Dooley 2002) techniques can incorporate the domain expert knowledge and manipulate numerical as well as linguistic data. They are also capable of drawing partial inference using fuzzy IF-THEN rules. But these rules are application-domain specific and are sensitive to both the structure of the membership functions and associated parameters used in each membership function. For example, the fuzzy rule-based segmentation technique proposed by Chi and Yan (1993) for geographic map images, intuitively defined the structure of the membership functions with the related parameters automatically determining the membership function. It is very difficult to define the rules either manually or automatically so that the segmentation can be achieved successfully. The advantages of the fuzzy rule-based image segmentation over the other methods are mainly that humans can more easily understand the problems due to linguistic representation of numeric variables; it is computationally less expensive than fuzzy clustering methods; and it has the potential ability to integrate the domain expert knowledge.

Fuzzy treatment of geometric and topological concepts can be performed in two distinct manners in image segmentation (Udupa and Saha
2003). The first approach applies a fuzzy image segmentation to obtain a fuzzy subset wherein every pixel has a fuzzy object membership assigned to it and then defines the geometric and topological concepts on this fuzzy subset. The second approach develops these concepts directly on the given image, which implies that these concepts have to be integrated with segmentation process. Considering the first approach, Rosenfeld (1998) extended the concepts of digital picture geometry to fuzzy subsets and generalized some of the standard geometric properties (area, perimeter and compactness of a fuzzy image subset) and relationships among regions to fuzzy subsets.

5.2 COLOR FEATURE EXTRACTION

Color is the sensation caused by the light as it interacts with our eyes and brain. Color features are the fundamental characteristics of the content of images. Human eyes are sensitive to colors, and color features enable human beings to distinguish between objects in the images. Colors are used in image processing because they provide powerful descriptors that can be used to identify and extract objects from a scene. Color features provide powerful information about images, and they are very useful for image segmentation.

5.2.1 Color Fundamentals

In 1666, Sir Isaac Newton discovered that a beam of sunlight consists of a continuous spectrum of colors when it passes through a glass prism. The colors are violet, indigo, blue, green, yellow, orange and red, which are shown in Figure 5.1 (Gonzales and Woods 2008).
Figure 5.1 Color spectrum seen by passing white light through a prism

To facilitate the specification of colors in some standard, color spaces (also called color models or color systems) are proposed. A color model is a color measurement scale or system that numerically specifies the perceived attributes of color. Color model is a method of grouping numeric values by a set of primaries. The purpose of a color model is to assist the specification of colors in some generally accepted standard. Generally, a color model is a specification of a 3-Dimensional (3D) coordinate system and a subspace within that system, where each color is represented by a single point. A pixel can be represented by different color bands that are described by different color models and stored in different data types. To describe a pixel, RGB, HSI, LUV and YCbCr color models are used, since some image processing techniques give different results for different color models. The performance of an image segmentation procedure is known to depend on the choice of color spaces. But conversion of one color to another color model is an easy process. In most digital image processing, RGB color space is used in practice for color monitors and CMY (Cyan, Magenta and Yellow) color space is used for color printing.
5.2.2 Color Space

To extract the color features from the content of an image, it is needed to select a color space and use its properties in the extraction. In common, colors are defined in three dimensional color space. Several color spaces are used to represent images for different purposes. The RGB (Gonzales and Woods 2008) color space is the most widely used color space. RGB stands for Red, Green, and Blue. RGB color space combines the three colors in different ratios to create other colors. In digital image processing, RGB color space is the most prevalent choice. The main drawback of the RGB color space is that it is perceptually nonuniform. Any color in the RGB color space can be represented by a vector of three coordinates. To overcome the drawback of the RGB color space, different color spaces are proposed.

The HSx color space is commonly used in digital image processing that converts the color space of the image from RGB color space to one of the HSx color spaces. HSx color space contains the HSI, HSV and HSB color spaces. They are common to human color perception. HS stands for Hue and Saturation. I, V, and B stand for Intensity, Value and Brightness respectively. The difference between them is their transformation method from the RGB color space. Hue describes the actual wavelength of the color. Saturation is the measure of the purity of the color. For example, red is 100% saturated color, but pink is not 100% saturated color because it contains some amount of white. Intensity describes the lightness of the color. HSV color space is widely used when converting the color space from RGB color space (Shih et al 2001).

Other color spaces are YUV and YIQ. They were developed for television broadcasting. The Y channel represents the luminance of the pixel and is the only channel used in the black and white television. The other channels (U, V, I and Q) are the chromatic components. The CIE L*u*v
and CIE L*a*b color spaces are both perceptually uniform systems and device-independent, which provide easy use of similarity metrics for comparing colors (TKALc and TASc 2003), (Deng et al. 2001).

To complete extracting color features from an image after choosing the proper color space, some effective color descriptors should be selected to represent the color of the image contents. Color histograms (Swain and Ballard 1991), color moments (Stricker and Orenge 1995) and color correlogram (Huang 1997) are some of the color descriptors that have been developed for color image representation.

5.2.2.1 RGB color model

RGB color model is the most common color model in use. The primary colors are red, green, and blue. From Figure 5.2, it is clear that the RGB model is based on a Cartesian coordinate system. The three coordinates of the system are the primary colors of the model. The color black is at the origin of the plane, and the color white is at the outermost corner from black. The different colors in this model are on or inside the cube, and are defined by vectors extending from the origin. Images represented in the RGB color model consist of three component images, one for each primary color. When a monitor displays the image, the three images are combined to produce the original image. The number of bits used to represent each pixel is called the pixel depth. For example, if each primary color in the RGB model is represented by 8 bits, then each pixel in an RGB color image is represented by using 24 bits (8 bits for each primary color). In general, the total number of colors in a 24-bit RGB image is \( (2^8)^3 \) that is equal to 16,777,216 colors (Gonzales and Woods 2008).
5.2.2.2 LUV color model

The International Commission on Illumination (also known as the CIE from its French title, the Commission Internationale de l’Eclairage) model forms the basis for most quantitative color measurement. LUV (TKALčíč and TASIč 2003) is designed to represent additive color systems. The aim is to have a color space with uniform scales and coordinates and to have an absolute representation of the object color. However, this is not strictly true, and this system represents a compromise. The LUV is approximately perceptually uniform. Since digital color images are typically stored as RGB values, a conversion between color spaces is necessary. There is no direct conversion between RGB and LUV color spaces. LUV color model represents all colors that are visible to the human eye using the following three bands

\[ L = R \times 0.299 + G \times 0.587 + B \times 0.114 \]  \hspace{1cm} (5.1)

\[ U = R \times -0.169 + G \times 0.332 + B \times 0.500 + 128 \]  \hspace{1cm} (5.2)
\[ V = R \ast -0.500 + G \ast -0.419 + B \ast -0.0813 + 128 \]  \hspace{1cm} (5.3)

where,

- **L** represents luminance
- **U** represents position of luminance. The negative values of ‘U’ yield green, where the positive values indicate red
- **V** represents position of luminance. The negative values of ‘V’ yield blue, where the positive values indicate yellow

### 5.2.2.3 HSV color model

Various color spaces have been introduced to represent and specify colors in a way suitable for storage, processing or transmission of color information in images. The HSV color space is quite similar to the way in which humans perceive colors. The colors used in HSV can be clearly defined by human perception, which is not always the case with RGB or CMYK (Cyan, Magenta, Yellow and Black). Out of many color models, HSV is one of the models that separate out the luminance component (Intensity) of a pixel color from its chrominance components (Hue and Saturation). Hue represents pure color, which is perceived when incident light is with sufficient illumination and contains a single wavelength. Saturation gives a measure of the degree by which a pure color is diluted by white light. For light with low illumination, corresponding intensity value in the HSV color space is also low. The HSV color space can be represented as a hexacone which is shown in Figure 5.3, with the central vertical axis denoting the luminance component ‘I’ (often denoted by V for Intensity Value). Hue is a chrominance component defined as an angle in the range \([0,2\pi]\) relative to the red axis with red at angle 0, green at \(2\pi/3\), blue at \(4\pi/3\) and red again at \(2\pi\). Saturation ‘S’ is the other chrominance component measured as a radial distance from the central
axis of the hexacone with value between 0 at the center to 1 at the outer surface.

![HSV color model](image)

**Figure 5.3 HSV color model**

### 5.2.3 Gray Level Co-occurrence Matrix

Gray Level Co-occurrence Matrix (GLCM) was proposed by Haralick et al (1973) and is widely used for texture analysis. The GLCM method is a way of extracting second order statistical texture features by considering the spatial relationship of pixels. The GLCM is a two dimensional array which takes into account the specific position of a pixel relative to other pixels. GLCM depicts how often different combinations of gray levels co-occur in an image. The GLCM is created by calculating how often a pixel with the intensity value $i$ occurs in a specific spatial relationship to a pixel with the value $j$. The spatial relationship can be specified in different ways, the default one is between a pixel and its immediate neighbor to its right, on the other hand, this relationship can be specified with different offsets and angles. The pixel at position $(i,j)$ in GLCM is the sum of the number of times the $(i,j)$ relationship occurs in the image.
Figure 5.4 Description of the gray level co-occurrence matrix

Figure 5.4 describes how to compute the GLCM. It shows an image and its corresponding co-occurrence matrix using the default pixel’s spatial relationship (offset = +1 in x direction). For the pair (2,1) (pixel 2 followed at its right by pixel 1), it is found 2 times in the image, then the GLCM image will have 2 as a value in the position corresponding to \( I_i = 1 \) and \( I_j = 2 \). The GLCM matrix is a 256x256 matrix; \( I_i \) and \( I_j \) are the intensity values for an 8 bit image.

<table>
<thead>
<tr>
<th>(135^\circ)</th>
<th>(90^\circ)</th>
<th>(45^\circ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(180^\circ)</td>
<td>X</td>
<td>0(^\circ)</td>
</tr>
<tr>
<td>(225^\circ)</td>
<td>(270^\circ)</td>
<td>315(^\circ)</td>
</tr>
</tbody>
</table>

Figure 5.5 Directions used for computing isotropic GLCM

The GLCM can be computed for the eight directions around the pixel of interest shown in Figure 5.5. Summing results from different directions leads to the isotropic GLCM and helps achieve a rotation invariant GLCM. Extension to color images is straightforward. In color space, the GLCM and its statistical features can be computed for each band.
5.2.4 Color Co-occurrence Matrix

Palm (2004) derived Color Co-occurrence Matrix (CCM) similar to GLCM, which measures both the color distribution in an image and the spatial interaction between pixels. These matrices are defined for each color space denoted by \((C1, C2, C3)\). Let \(C_k\) and \(C_{k'}\), be the two of the three color components of this space \((k, k' \in \{1, 2, 3\})\) and \(CCM_{k,k'}\) is the color co-occurrence matrix which measures the spatial interaction between the components \(C_k\) and \(C_{k'}\) of the pixels in the image \(I\). The cell \(CCM_{k,k'}(i, j)\) of this matrix contains number of times that any pixel \(P\) whose \(k^{th}\) color component value is equal to \(i\), and \(k'^{th}\) color component value is equal to \(j\),

\[
CCM_{k,k'}(i, j) = \frac{1}{\sum_{\Delta x} \sum_{\Delta y}} \begin{cases} 
1, & \text{if } (k(x + \Delta x, y + \Delta y) = i \text{ and } (k'(x + \Delta x, y + \Delta y) = j) \\
0, & \text{otherwise}
\end{cases}
\]

Each color image \(I\) characterized by the six color co-occurrence matrices for LUV color planes are \(CCM_{L,L}\), \(CCM_{L,U}\), \(CCM_{L,V}\), \(CCM_{U,U}\), \(CCM_{U,V}\), and \(CCM_{V,V}\). Texture can be perceived at different scales. Each scale, however, requires a different window size. This is true for both human perception and computer-based texture recognition.

5.2.5 ICICM

Vadivel et al (2007) have proposed an integrated approach for capturing spatial variation of both color and intensity levels in the neighborhood of each pixel using HSV color space. In their research they have estimated the amount of color and intensity variation between each pixel and its neighbor using weight function. Suitable weighted constraints are satisfied while choosing the weight function for effectively relating visual perception of color and the HSV color space properties. Therefore, ICICM is
generalization of GLCM and CCM techniques. The advantage of ICICM is that the color and intensity variations are represented in a single composite feature. Some of the useful properties of the HSV color space and their relationship to human color perception are utilized for extracting this feature.

The generation of ICICM algorithm is summarized below

1. ICICM is two-dimensional matrix which considers relative contribution to color and gray level perception for each pixel \( p \) and its neighbor \( N_p \). ICICM consists of four sub matrices and it can be represented as follows:

\[
\begin{pmatrix}
\text{ICICM}_{CC} & \text{ICICM}_{CI} \\
\text{ICICM}_{IC} & \text{ICICM}_{II}
\end{pmatrix}
\]  

(5.5)

where

- \( \text{ICICM}_{CC} \) - color perception of pixel \( p \) and color perception of its neighbor \( N_p \)
- \( \text{ICICM}_{CI} \) - color perception of pixel \( p \) and gray level perception of its neighbor \( N_p \)
- \( \text{ICICM}_{IC} \) - gray level perception of pixel \( p \) and color perception of its neighbor \( N_p \)
- \( \text{ICICM}_{II} \) - gray level perception of pixel \( p \) and gray level perception of its neighbor \( N_p \)

Each component of \( \text{ICICM}_{CC}, \text{ICICM}_{CI}, \text{ICICM}_{IC} \) and \( \text{ICICM}_{II} \) can be written as follows:

\[
icm_{CC}(i,j)_{i=0..H-1; j=0..I-1} = Pr(h_l|p, h_l|N_p) = (i,j)
\]  

(5.6)
\[
\text{icicm}_{Cf}(i,j)_{i=0..HL-1;j=0..GL-1} = Pr\left((hl_p,gl_{N_p}) = (i,j)\right) \tag{5.7}
\]

\[
\text{icicm}_{Cc}(i,j)_{i=0..GL-1;j=0..HL-1} = Pr\left((gl_p,hl_{N_p}) = (i,j)\right) \tag{5.8}
\]

\[
\text{icicm}_{Ch}(i,j)_{i=0..GL-1;j=0..GL-1} = Pr\left((gl_p,gl_{N_p}) = (i,j)\right) \tag{5.9}
\]

Thus, \(\text{icicm}_{CC}(i,j)\) is the number of times the color perception of a pixel \(p\) denoted by \(hl_p\) equals \(i\), and the color perception of its neighbor \(N_p\) denoted by \(hl_{N_p}\) equals \(j\), as a fraction of the total number of pixels in the image. Similarly, \(\text{icicm}_{CD}(i,j)\) is the number of times the color perception of a pixel \(p\) denoted by \(hl_p\) equals \(i\), and the gray level perception of its neighbor \(N_p\) denoted by \(gl_{N_p}\) equals \(j\), as a fraction of the total number of pixels in the image.

2. The dimension of the matrix ICICM is determined by the number of quantization levels of HL and GL; (HL is the number of quantized levels of Hue and GL is the number of quantized levels of Intensity derived from the HSV color space) it can be computed as follows:

\[
HL = \left\lfloor \frac{2\pi}{Q_h} \right\rfloor + 1 \tag{5.10}
\]

\[
GL = \left\lfloor \frac{255}{Q_I} \right\rfloor + 1 \tag{5.11}
\]

where \(Q_h\) and \(Q_I\) are quantization factors for Hue and Intensity.

3. The ICICM matrix is updated using a weight function \(W_{col}(S,I)\) that estimates the extent of color perception of a pixel. Based on the HSV color space property the weight should be the function of both
saturation (S) and intensity (I). The $W_{col}(S,I)$ function may have any one of the following constraints:

a) $W_{col}(S,I) \in [0,1]$

b) For $S_1 > S_2$, $W_{col}(S_1, I) > W_{col}(S_2, I)$

c) For $I_1 > I_2$, $W_{col}(S, I_1) > W_{col}(S, I_2)$

d) $W_{col}(S,I)$ changes slowly with S for high values of I.

e) $W_{col}(S,I)$ changes sharply with S for low values of I.

Constraints (a) - (c) follow directly from the properties of the HSV color space. Constraint (d) follows from the fact that when intensity is high, the loss of color perception is only due to dilution of color by white light. On the other hand, constraint (e) is required since for low intensity, loss of color perception is a combined effect of cone cell de-activation and color dilution. Vadivel et al (2007) found the following weight function after detailed analysis.

$$W_{col}(S,I) = \begin{cases} S_1^{r_1} \times (255I)^{r_2}, & \text{for } I \neq 0 \\ 0, & \text{for } I = 0 \end{cases} \quad (5.12)$$

where $r_1$ and $r_2$ are constants, which are dependent on the particular applications.

4. The intensity weight of the pixel is computed as a complement of the color weight as given below:

$$W_{int}(S,I) = 1 - W_{col}(S,I) \quad (5.13)$$
5.3 TEXTURE FEATURE EXTRACTION

Texture segmentation is the identification of regions based on their texture features. The segmentation of textures requires the choice of proper texture-specific features with good discriminative power. Basically, texture feature extraction may be based on three approaches, namely (i) statistical approach (ii) structural approach and (iii) spectral approach. In statistical approach, moments of different order in a localized window can be used to represent smooth, coarse, grainy, textures and co-occurrence statistics of an image. In structural approach, spatial structure descriptors are used to identify geometric primitives and their arrangement in an image. Some techniques using this approach are: local interaction models like auto regressive model (Lu and Xu 1995), Gauss-Markov random field model(Krishnamachari and Chellappa 1997) and the fractal model(Kasparis et al 2001). The spectral approach is the transform domain approach that is used to detect global periodicity in an image by finding high energy, narrow peaks in the spectrum. This approach is not popular due to high computational cost. But a combination of spectral and spatial approaches such as Gobor filters (Kasparis et al 2001) and wavelet transform (Unser et al 1995) are becoming popular.

Maheswari et al (2011) have reported that in pattern recognition and image processing, feature extraction is a special form of dimensionality-reduction. In statistics, dimensionality-reduction is the process of reducing the number of random variables under consideration and can be divided into feature selection and feature extraction. When the input data to an algorithm is too large to be processed and it is suspected to be notoriously redundant (much data, but not much information), then the input data will be transformed into a reduced representation set of features. Transforming the input data into the set of features is called feature extraction. If the features
extracted are carefully chosen, it is expected that the feature set will extract the relevant information from the input data in order to perform the desired task using this reduced representation instead of the full size input. Features often contain information relative to gray scale, texture, shape or context. Most of the researchers used Gabor filter, Wavelet transform, Local binary pattern, Fractal models and Haralick’s features for their work.

### 5.3.1 Local Binary Patterns

The Local Binary Pattern (LBP) texture analysis operator was introduced by Ojala et al (1996). It is a gray scale invariant texture measure computed from the analysis of a 3x3 local neighborhood over a central pixel. The LBP is based on a binary code describing the local texture pattern. This code is built by thresholding a local neighborhood by the gray value of its center. The eight neighbors are labeled using a binary code \{0, 1\} obtained by comparing their values to the central pixel value. If the tested gray value is below the gray value of the central pixel, then it is labeled as 0, otherwise it is assigned the value 1:

\[
P_i' = \begin{cases} 
0 & \text{if } l(x_i, y_i) < l(x_0, y_0) \\
1 & \text{otherwise}
\end{cases}
\]

(5.14)

$P_i'$ is the obtained binary code, $P_i$ is the original pixel value at position $i$ and $P_0$ is the central pixel value. With this technique, there is 256 ($2^8$) possible patterns (or texture units). The obtained value is then multiplied by weights given to the corresponding pixels. The weight is given by the value $2^{i-1}$. Summing the obtained values gives the measure of the LBP:

\[
I_{LBP} = \sum_{i=1}^{8} P_i'2^{i-1}
\]

(5.15)
Figure 5.6 shows an example of how to compute LBP. The original 3x3 neighborhood is given in Figure 5.6 (a). The central pixel value is used as a threshold in order to assign a binary value to its neighbors. Figure 5.6 (b) shows the result of thresholding of the 3x3 neighborhood. The obtained values are multiplied by their corresponding weights. The weight kernel is given by Figure 5.6 (c). The result is given in Figure 5.6 (d). The sum of the resulting values gives the LBP measure (169). The central pixel is replaced by the obtained value. A new LBP image is constructed by processing each pixel and its 3x3 neighbors in the original image.

![LBP image example](image)

**Figure 5.6 Computation of LBP**

### 5.3.2 Haralick Features

Haralick features are considered for extracting the properties from texture images. Haralick introduced 14 texture features extracted from CCM. These features are statistical measures on the CCM of an image which allow reduction of information quantity of each matrix. Palm used 8 of these 14 Harlaiick features, namely homogeneity, contrast, correlation, variance, inverse difference moment, entropy, correction 1 and 2. Maheswari et al (2011) used only 5 of these 14 Haralick features, angular second moment ($f_1$), contrast ($f_2$), correlation ($f_3$), inverse difference moment ($f_4$) and variance ($f_5$), in order to reduce the size of the feature space. Generally, for each image
coded in a color space, it is needed to calculate 3 CCM matrices and so \( N_f = 3 \times 14 \) Haralick features are extracted from these matrices. The total number \( (N_c) \) of candidate color texture features are very high. Since it is needed to reduce the feature space, in the proposed method ICICM is calculated and 1 x 5 Haralick features are extracted from this matrix. Therefore, the total number \( (N_c) \) of candidate color and texture features are reduced significantly.

\[
\begin{align*}
 f_1(CCM) &= \sum_i \sum_j p(i,j)^2 \\
 f_2(CCM) &= \sum_{n=0}^{N_g-1} n^2 \left[ \sum_{j=0}^{N_g} \sum_{i=0}^{N_g} p(i,j) \right], |i = j| = n \\
 f_3(CCM) &= \frac{\sum_{j} \sum_{i} p(i,j) - \mu_x \mu_y}{\sigma_x \sigma_y} \\
 f_4(CCM) &= \sum_i \sum_j \frac{p(i,j)}{1 + (i - j)^2} \\
 f_5(CCM) &= \sum_i \sum_j (p(i,j) - \mu)^2 p(i,j)
\end{align*}
\]

Here \( p(i,j) \) is the gray scale value of specified pixel \( p \), \( \mu_x, \mu_y \) and \( \sigma_x, \sigma_y \) are the means and standard deviations of marginal probability matrix obtained by summing the rows/columns of \( p(i,j) \).

5.4 Fuzzy Clustering Algorithms

5.4.1 Fuzzy C-Means Algorithm

FCM is a method of clustering which allows one piece of data to belong to two or more clusters and it is frequently used in pattern recognition. In unsupervised learning, FCM clustering algorithms are best known and most powerful methods used in cluster analysis. The FCM algorithm, proposed by Dunn (1973) and generalized by Bezdek (1981), is used to describe the fuzzy classification for the pixels by calculating the fuzzy membership value. FCM algorithm is a data clustering algorithm in which
each data point belongs to a cluster, to a degree specified by a membership
grade. The FCM algorithm attempts to partition a finite collection of elements
$X = \{x_1, x_2, \ldots, x_n\}$ into a collection of $c$ fuzzy clusters with respect to some
given criterion. When fuzzy clustering is attempted, fuzzy membership design
includes various uncertainties, such as distance measure, fuzzifier and
prototype. It minimizes an objective function $J$, with respect to fuzzy
membership $U$,

$$J = \sum_{j=1}^{m} \sum_{i=1}^{n} (u_{ij})^m \|x_i^{(j)} - c_j\|^2$$  \hspace{1cm} (5.21)

The issues in FCM are that it is computationally expensive and
highly dependent on the initial choice of $U$. If data-specific experimental
values are not available, then $m = 2$ is the usual choice. Extensions that exist
simultaneously estimate the intensity inhomogeneity bias field while
producing the fuzzy partitioning.

**FCM algorithm**

**Step 1** : Consider a set of $n$ data points to be clusters, $x_i$

**Step 2** : Assume the number of clusters $c$, is know, $2 \leq c < n$

**Step 3** : Choose an appropriate level of cluster fuzziness, $m \in \mathbb{R}_{>1}$.

**Step 4** : Initialize the $(n \times c)$ sized membership matrix $U$ to
random values such that $u_{ij} \in [0,1]$ and $\sum_{j=1}^{c} u_{ij} = 1$

**Step 5** : Calculate the cluster centers $c_j$ using

$$c_j = \frac{\sum_{i=1}^{n} (u_{ij})^m x_i}{\sum_{i=1}^{n} (u_{ij})^m}, \text{for } j = 1 \ldots c$$  \hspace{1cm} (5.22)
Step 6: Calculate the distance measures

\[ d_{ij} = \| x_i^{(j)} - c_j \| \]  \hspace{1cm} (5.23)

for all clusters j=1…c and data points j=1…n

Step 7: Update the fuzzy membership matrix U according to \( d_{ij} \).

If \( d_{ij} > 0 \) then 

\[ u_{ij} = \left[ \frac{1}{\sum_{k=1}^{c} \left( \frac{d_{ij}}{d_{ik}} \right)^{\frac{2}{m-1}}} \right]^{-1} \]

If \( d_{ij} = 0 \) then the data points \( x_j \) coincides with the cluster center \( c_j \), and so full membership can be set \( u_{ij}=1 \)

Step 8: Repeat from Step 5 to Step 6 until the change in \( U \) is less than a given tolerance.

5.4.2 IT2FCM Algorithm

IT2FSs have two membership functions, upper and lower memberships. The IT2FCM has two fuzzifiers \( m_1 \) and \( m_2 \), which represent different fuzzy degrees and give different objective functions to be minimized in FCM as

\[ J_{m_1(U,v)} = \sum_{i=1}^{N} \sum_{j=1}^{C} u_j(x_i)^{m_1} d_{ji}^2 \]  \hspace{1cm} (5.24)

\[ J_{m_2(U,v)} = \sum_{i=1}^{N} \sum_{j=1}^{C} u_j(x_i)^{m_2} d_{ji}^2 \]  \hspace{1cm} (5.25)
The Interval Type-2 Fuzzy Membership becomes

\[
\overline{u}_j(x_i) = \begin{cases} 
\frac{1}{\sum_{k=1}^{C} \left( \frac{(d_{ji}/d_{ki})}{2/(m_1-1)} \right)^{2/(m_1-1)}}, & \text{if } \frac{1}{\sum_{k=1}^{C} (d_{ji}/d_{ki})} < \frac{1}{C} \\
\frac{1}{\sum_{k=1}^{C} \left( \frac{(d_{ji}/d_{ki})}{2/(m_2-1)} \right)^{2/(m_2-1)}}, & \text{otherwise}
\end{cases}
\]

(5.26)

\[
u_j(x_i) = \begin{cases} 
\frac{\sum_{k=1}^{C} \left( \frac{(d_{ji}/d_{ki})}{2/(m_1-1)} \right)^{2/(m_1-1)}}, & \text{if } \frac{1}{\sum_{k=1}^{C} (d_{ji}/d_{ki})} \geq \frac{1}{C} \\
\frac{1}{\sum_{k=1}^{C} \left( \frac{(d_{ji}/d_{ki})}{2/(m_2-1)} \right)^{2/(m_2-1)}}, & \text{otherwise}
\end{cases}
\]

(5.27)

Algorithm

Step 1: Compute \(\underline{u}_j^{(l)}\) and \(\overline{u}_j^{(l)}\)

Step 2: Updating cluster centers

\[
v_j^{(l)} = \frac{v_j^{(l)} + v_k^{(l)}}{2}
\]

(5.28)

Step 3: Type-reduction and hard-partitioning can be obtained as follows:

\[
u_j(x_i) = \frac{u_j^R(x_i) + u_j^L(x_i)}{2}, \quad j = 1, \ldots, C
\]

(5.29)

\[
u_j^R(x_i) = \frac{\sum_{l=1}^{M} u_{jl}(x_i)}{M}
\]

(5.30)

where \(u_{jl}(x_i) = \begin{cases} 
\overline{u}_j(x_i), & \text{if } x_{il} \text{ uses } \overline{u}_j(x_i) \text{ for } v_j^R \\
u_j(x_i), & \text{otherwise}
\end{cases}\)

and

\[
u_j^L(x_i) = \frac{\sum_{l=1}^{M} u_{jl}(x_i)}{M}
\]

(5.31)
where \( u_{ji}(x_i) = \begin{cases} \overline{u}_i(x_i), & \text{if } x_{ii} \text{ uses } \overline{u}_i(x_i) \text{ for } v_i^j \\ u_j(x_i), & \text{otherwise} \end{cases} \)

**Step 4:** If \( (u_i(x_i) > u_k(x_i)) \), for \( k = 1, ..., C \) and \( j \neq k \)

Then \( x_i \) is assigned to cluster \( j \).

### 5.5 Color Texture Image Segmentation Using Extended IT2FCM

**Figure 5.7 Proposed hybrid approach – Extended IT2FCM**

The proposed hybrid approach segments the color texture image into different regions using Haralick features extracted from ICICM. Figure 5.7 is a representation of the proposed approach. The images obtained from benchmark (http://mosaic.utia.cas.cz) database are in RGB color model. HSV color space is suitable for calculating the ICICM. So RGB color image is converted into HSV color space and then ICICM is calculated. From this ICICM, the texture features are extracted. Finally, the images are segmented by using Extended IT2FCM clustering approach.

#### 5.5.1 Extended IT2FCM Algorithm

The pixels on textured image are highly correlated; therefore, the spatial relationship of neighboring pixels is an important characteristic that can increase the performance of segmentation approaches. This spatial relationship is important in clustering, but it is not utilized in a standard FCM
algorithm. The aim of this work is to develop a space-partitioning algorithm that is able to return meaningful results even when applied to composite and complex natural scenes that have large variations in color and texture. To achieve this a new clustering strategy called Extended IT2FCM is proposed whose implementation can be viewed as extension of IT2FCM. The main idea behind Extended IT2FCM is to minimize an objective function J based on hybrid approach which is extracting Haralick features from ICICM.

- When adding the spatial information, the feature partition will be formulated as the minimization of a new objective function given by

\[
J_{m_1(u,v)} = \sum_{i=1}^{N} \sum_{j=1}^{C} u_j(x_i)^{m_1} d_{ij}^2 + \alpha \sum_{i=1}^{N} \sum_{j=1}^{C} \frac{1}{n} \sum_{\delta \epsilon w} u_j(x_i)^{m_1} d_{ji+\delta}^2
\] (5.32)

\[
J_{m_2(u,v)} = \sum_{i=1}^{N} \sum_{j=1}^{C} u_j(x_i)^{m_2} d_{ij}^2 + \alpha \sum_{i=1}^{N} \sum_{j=1}^{C} \frac{1}{n} \sum_{\delta \epsilon w} u_j(x_i)^{m_2} d_{ji+\delta}^2
\] (5.33)

\[
\alpha \text{ – is selected experimentally}
\]

\[
w \text{ – Set of neighbors that exist in a window around a central pixel.}
\]

- The Interval Type-2 Fuzzy Membership becomes

\[
\bar{u}_j(x_i) = \begin{cases} \frac{\sum_{k=1}^{C} ((d_{ji}/d_{kl}) + \alpha (d_{ji}/d_{kl}) \delta)^{2/(m_1-1)}}{\sum_{k=1}^{C} ((d_{ji}/d_{kl}) + \alpha (d_{ji}/d_{kl}) \delta)^{2/(m_2-1)}} & \text{if } \frac{1}{\sum_{k=1}^{C} (d_{ji}/d_{kl})} \\ \frac{\sum_{k=1}^{C} ((d_{ji}/d_{kl}) + \alpha (d_{ji}/d_{kl}) \delta)^{2/(m_2-1)}}{\sum_{k=1}^{C} ((d_{ji}/d_{kl}) + \alpha (d_{ji}/d_{kl}) \delta)^{2/(m_2-1)}} & \text{otherwise} \end{cases}
\] (5.34)
\[ u_j(x_i) = \begin{cases} 
\left( \frac{\sum_{k=1}^{C} \left[ \left( \frac{d_{ji}}{d_{ki}} \right) + \alpha \left( \frac{d_{ji}}{d_{ki}} \right) \delta \right]^{2/(m_1-1)} }{\sum_{k=1}^{C} \left( \frac{d_{ji}}{d_{ki}} \right) + \alpha \left( \frac{d_{ji}}{d_{ki}} \right) \delta } \right)^{2/(m_1-1)}, & \text{if } \frac{1}{\sum_{k=1}^{C} (d_{ji}/d_{ki})} > 0, \\
\left( \frac{\sum_{k=1}^{C} \left[ \left( \frac{d_{ji}}{d_{ki}} \right) + \alpha \left( \frac{d_{ji}}{d_{ki}} \right) \delta \right]^{2/(m_2-1)} }{\sum_{k=1}^{C} \left( \frac{d_{ji}}{d_{ki}} \right) + \alpha \left( \frac{d_{ji}}{d_{ki}} \right) \delta } \right)^{2/(m_2-1)}, & \text{otherwise} 
\end{cases} \]

- Updating cluster centers

\[ v_j = \frac{v_L + v_R}{2} \quad \text{(5.36)} \]

- Type-reduction and hard-partitioning can be obtained as follows:

\[ u_j(x_i) = \frac{u_j^R(x_i) + u_j^L(x_i)}{2}, \quad j = 1, \ldots, C \quad \text{(5.37)} \]

\[ u_j^R(x_i) = \frac{\sum_{l=1}^{M} u_{jl}(x_i)}{M} \quad \text{(5.38)} \]

where \( u_{jl}(x_i) = \begin{cases} u_j(x_i), & \text{if } x_{it} \text{ uses } \overline{u}_j(x_i) \text{ for } v_j^R \text{ and} \\
\overline{u}_j(x_i), & \text{otherwise} \end{cases} \)

\[ u_j^L(x_i) = \frac{\sum_{l=1}^{M} u_{jl}(x_i)}{M} \quad \text{(5.39)} \]

where \( u_{jl}(x_i) = \begin{cases} \overline{u}_j(x_i), & \text{if } x_{it} \text{ uses } \overline{u}_j(x_i) \text{ for } v_j^L \text{ and} \\
\overline{u}_j(x_i), & \text{otherwise} \end{cases} \)

When any algorithm is applied in texture segmentation, two texture properties are to be considered: homogeneity property and texture boundary property. In homogenous regions, the spatial functions simply strengthen the original membership and the clustering result remains unchanged. But for a
texture boundary property, the formula reduces the weighting of a noisy cluster by the labels of its neighboring pixels. As a result, misclassified pixels from noisy regions or spurious blobs can easily be corrected. Clustering is a two-pass process at each iteration. In the first pass, the membership function in the spectral domain is calculated. In the second pass, the membership information of each pixel is mapped to the spatial domain. The iteration is stopped when the maximum difference between the two cluster centers at two successive iterations is less than a threshold. After the convergence, defuzzification is applied to assign each pixel to a specific cluster whose membership is the maximum value.

5.5.2 Experimental Result

The proposed hybrid approach is tested on a large number of benchmark database images (http://mosaic.utia.cas.cz) and natural color texture images in order to evaluate its performance with respect to the identification of parameter values, and the experimental result is compared with T1FCM, IT2FCM, and Extended IT2FCM. Two sets of experiments have been carried out: In the first set, after the Haralick features are extracted in CCM, the clustering approach is performed using (i) T1FCM, (ii) IT2FCM and (iii) Extended IT2FCM. The second set of experiment is carried out by extracting the Haralick features in ICICM and then clustering approach is performed using (i) T1FCM, (ii) IT2FCM and (iii) Extended IT2FCM.

5.5.2.1 Evaluation of segmentation result

A large number of experiments were carried out to analyze the performance of the proposed color texture segmentation algorithm using MATLAB tool. Figure 5.8 shows the sample test images taken from the
http://mosaic.utia.cas.cz, benchmark database and natural images. To demonstrate the capability of the proposed method, four images (namely Txt_1.jpg, Txt_2.jpg, Rimg_1.jpg, Rimg_2.jpg) are evaluated in detail to highlight the advantage of the proposed approach. The performance of Extended IT2FCM approach is evaluated by comparing the algorithmic efficiency and the segmentation results with T1FCM, IT2FCM and Extended IT2FCM.

![Figures](image1.jpg, image2.jpg, image3.jpg, image4.jpg)

**Figure 5.8 Sample test images**

The ICICM method’s effectiveness is based on two factors, they are the quantization levels (HL and GL) and choice of parameter ($r_1$ and $r_2$). Here a fuzzy set is extended into an IT2FCM by incorporating two different
values of fuzzifier m (m₁ and m₂). Therefore the proposed method has six parameters, namely (HL and GL), (r₁ and r₂) in ICICM algorithm and (m₁ and m₂) in Extended IT2FCM algorithm. Four different combinations of hue and intensity levels HL and GL are considered. For example, the combination (4, 2) means that the color weight - color weight will update a (2, 2) area, intensity weight-color weight updates (4,4) area, color weight-intensity weight updates (4,2) area and intensity weight-intensity weight updates (2,4) area of ICICM. To determine the best combination of r₁ and r₂ three sets of values (.05, .8), (.1,.85) and (.15,.9) are experimented. Three possible combinations of m₁ and m₂ are experimented (.4, .6), (.7, .9) (.8, 1.0) with ICICM parameters.

The first set of experiment is carried out to extract the Haralick features in CCM, then clustering approach is performed using T1FCM, IT2FCM and Extended IT2FCM with various combination values of (HL and GL), (r₁ and r₂) and (m₁ and m₂). The experimental segmentation results are depicted in Figures 5.9 to 5.12. Figures 5.9 (a) to 5.12 (a) show the segmentation results using the T1FCM algorithm. Here the algorithm over segments the image and boundary is lost. Figures 5.9 (b) to 5.12 (b) present segmentation based on IT2FCM; here it shows that the number of segments are reduced, but the pixels of different texture are grouped together. This improper grouping among different texture regions leads to the loss of the homogeneity property of textures. Figures 5.9 (c) to 5.12 (c) present segmentation based on Extended IT2FCM, which shows that homogeneous regions are extracted with less number of misclassifications.
Figure 5.9 Segmentation result for image Txt_1.jpg - Extracting the Haralick features in CCM, then clustering approach is performed using (a) T1FCM (b) IT2FCM (c) Extended IT2FCM
Figure 5.10  Segmentation result for image Txt_2.jpg - Extracting the Haralick features in CCM, then clustering approach is performed using (a)T1FCM (b)IT2FCM (c) Extended IT2FCM
Figure 5.11  Segmentation result for image Rimg_1.jpg- Extracting the Haralick features in CCM, then clustering approach is performed using (a) T1FCM (b)IT2FCM (c) Extended IT2FCM
Figure 5.12  Segmentation result for image Rim_2.jpg- Extracting the Haralick features in CCM, then clustering approach is performed using (a) T1FCM (b) IT2FCM (c) Extended IT2FCM

The second set of experiments were carried out to extract the Haralick features in ICICM, and then clustering approach is performed using T1FCM, IT2FCM and Extended IT2FCM with various combination values of (HL and GL), \( r_1 \) and \( r_2 \) and \( m_1 \) and \( m_2 \). The experimental segmentation results are depicted in Figures 5.13 to 5.16. Figures 5.13 (a) to 5.16 (a) are the
segmentation results using the T1FCM algorithm. Here the algorithm under segments, the image and boundary are not clear. Figures 5.13 (b) to 5.16 (b) present segmentation based on IT2FCM, where it shows that the number of misclassified pixel is reduced when comparing the results obtained by T1FCM.

![Image](image1.png)

(a) ![Image](image2.png)

(b) ![Image](image3.png)

(c)

**Figure 5.13** Segmentation result for image_txt_1.jpg- Extracting the Haralick features in ICICM, then clustering approach is performed using (a)T1FCM (b)IT2FCM (c)Extended IT2FCM
Figures 5.13(c) to 5.16 (c) show the result of the proposed approach. It shows that the segment accuracy is increased by reducing misclassification of pixels in the boundary region; also homogenous regions are properly segmented with very good pixel level clarity. Table 5.1 shows the number of segments produced by different algorithms.

Figure 5.14  Segmentation result for image Txt_2.jpg- Extracting the Haralick features in ICICM, then clustering approach is performed using (a)T1 FCM (b)IT2FCM (c)Extended IT2FCM
Figure 5.15  Segmentation result for image Rimg_1.jpg- Extracting the Haralick features in ICICM, then clustering approach is performed using (a)T1 FCM (b)IT2FCM (c)Extended IT2FCM
Figure 5.16  Segmentation Result for image Rimg_2.jpg- Extracting the Haralick features in ICICM, then clustering approach is performed using (a)T1FCM (b)IT2FCM (c)Extended IT2FCM
Table 5.1 Number of segments produced by different algorithms

<table>
<thead>
<tr>
<th>Methods</th>
<th>T1FCM</th>
<th>IT2FCM</th>
<th>Extended IT2FCM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Images</td>
<td>CCM</td>
<td>ICICM</td>
<td>CCM</td>
</tr>
<tr>
<td>Txt_1.jpg</td>
<td>16</td>
<td>12</td>
<td>14</td>
</tr>
<tr>
<td>Txt_2.jpg</td>
<td>16</td>
<td>14</td>
<td>13</td>
</tr>
<tr>
<td>Rimg_1.jpg</td>
<td>10</td>
<td>08</td>
<td>09</td>
</tr>
<tr>
<td>Rimg_2.jpg</td>
<td>11</td>
<td>09</td>
<td>11</td>
</tr>
</tbody>
</table>

Table 5.2 summarizes various parameter settings in the ICICM and Extended IT2FCM and corresponding number of segment count (four combinations of HL and GL, three combinations of r₁ and r₂ and three combinations of m₁ and m₂) and numerical results. According to the numerical values depicted in Table 5.2, it is found that when (HL=4 and GL=4), (r₁=.05 and r₂=.8) and (m₁=.7 and m₂=.9), it produces a good number of segments in the experimental image.

Table 5.2 Various parameter settings in the ICICM and extended IT2FCM and corresponding number of segment count (four combinations of HL and GL, three combinations of r₁ and r₂ and three combinations of m₁ and m₂)

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Number of Segment count</th>
</tr>
</thead>
<tbody>
<tr>
<td>HL=2, GL=2</td>
<td></td>
</tr>
<tr>
<td>r₁</td>
<td>r₂</td>
</tr>
<tr>
<td>.05</td>
<td>.8</td>
</tr>
<tr>
<td>.05</td>
<td>.8</td>
</tr>
<tr>
<td>.05</td>
<td>.8</td>
</tr>
<tr>
<td>.1</td>
<td>.85</td>
</tr>
<tr>
<td>.1</td>
<td>.85</td>
</tr>
<tr>
<td>.1</td>
<td>.85</td>
</tr>
<tr>
<td>.15</td>
<td>.9</td>
</tr>
<tr>
<td>.15</td>
<td>.9</td>
</tr>
<tr>
<td>.15</td>
<td>.9</td>
</tr>
<tr>
<td>HL=4, GL=4</td>
<td>$\beta$</td>
</tr>
<tr>
<td>------------</td>
<td>--------</td>
</tr>
<tr>
<td>0.05</td>
<td>0.8</td>
</tr>
<tr>
<td>0.05</td>
<td>0.8</td>
</tr>
<tr>
<td>0.1</td>
<td>0.85</td>
</tr>
<tr>
<td>0.1</td>
<td>0.85</td>
</tr>
<tr>
<td>0.1</td>
<td>0.85</td>
</tr>
<tr>
<td>0.15</td>
<td>0.9</td>
</tr>
<tr>
<td>0.15</td>
<td>0.9</td>
</tr>
<tr>
<td>0.15</td>
<td>0.9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HL=2, GL=4</th>
<th>$\beta$</th>
<th>$\alpha$</th>
<th>$\gamma$</th>
<th>$\delta$</th>
<th>$\epsilon$</th>
<th>$\zeta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>0.8</td>
<td>0.7</td>
<td>0.9</td>
<td>13</td>
<td>13</td>
<td>08</td>
</tr>
<tr>
<td>0.05</td>
<td>0.8</td>
<td>0.8</td>
<td>0.1</td>
<td>12</td>
<td>12</td>
<td>06</td>
</tr>
<tr>
<td>0.1</td>
<td>0.85</td>
<td>0.4</td>
<td>0.6</td>
<td>12</td>
<td>12</td>
<td>07</td>
</tr>
<tr>
<td>0.1</td>
<td>0.85</td>
<td>0.7</td>
<td>0.9</td>
<td>12</td>
<td>11</td>
<td>07</td>
</tr>
<tr>
<td>0.1</td>
<td>0.85</td>
<td>0.8</td>
<td>0.1</td>
<td>11</td>
<td>11</td>
<td>06</td>
</tr>
<tr>
<td>0.15</td>
<td>0.9</td>
<td>0.4</td>
<td>0.6</td>
<td>13</td>
<td>10</td>
<td>07</td>
</tr>
<tr>
<td>0.15</td>
<td>0.9</td>
<td>0.7</td>
<td>0.9</td>
<td>13</td>
<td>12</td>
<td>07</td>
</tr>
<tr>
<td>0.15</td>
<td>0.9</td>
<td>0.8</td>
<td>0.1</td>
<td>11</td>
<td>12</td>
<td>06</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HL=4 GL=2</th>
<th>$\beta$</th>
<th>$\alpha$</th>
<th>$\gamma$</th>
<th>$\delta$</th>
<th>$\epsilon$</th>
<th>$\zeta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>0.8</td>
<td>0.7</td>
<td>0.9</td>
<td>11</td>
<td>11</td>
<td>06</td>
</tr>
<tr>
<td>0.05</td>
<td>0.8</td>
<td>0.8</td>
<td>0.1</td>
<td>11</td>
<td>12</td>
<td>06</td>
</tr>
<tr>
<td>0.1</td>
<td>0.85</td>
<td>0.4</td>
<td>0.6</td>
<td>11</td>
<td>13</td>
<td>07</td>
</tr>
<tr>
<td>0.1</td>
<td>0.85</td>
<td>0.7</td>
<td>0.9</td>
<td>12</td>
<td>12</td>
<td>08</td>
</tr>
<tr>
<td>0.1</td>
<td>0.85</td>
<td>0.8</td>
<td>0.1</td>
<td>12</td>
<td>12</td>
<td>08</td>
</tr>
<tr>
<td>0.15</td>
<td>0.9</td>
<td>0.4</td>
<td>0.6</td>
<td>11</td>
<td>11</td>
<td>06</td>
</tr>
<tr>
<td>0.15</td>
<td>0.9</td>
<td>0.7</td>
<td>0.9</td>
<td>10</td>
<td>13</td>
<td>06</td>
</tr>
<tr>
<td>0.15</td>
<td>0.9</td>
<td>0.8</td>
<td>0.1</td>
<td>10</td>
<td>11</td>
<td>07</td>
</tr>
</tbody>
</table>

The efficiency of the Extended IT2FCM approach and other approaches are compared with the execution time. The execution time for these clustering approaches is tabulated in Table 5.3. The execution time of the proposed approach is relatively less than T1FCM and IT2FCM.
Table 5.3 Execution time (in seconds) for various clustering approaches

<table>
<thead>
<tr>
<th>Methods</th>
<th>T1FCM</th>
<th>IT2FCM</th>
<th>Extended IT2FCM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Images</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Txt_1.jpg</td>
<td>24.0</td>
<td>18.3</td>
<td>16.4</td>
</tr>
<tr>
<td>Txt_2.jpg</td>
<td>22.2</td>
<td>19.2</td>
<td>15.3</td>
</tr>
<tr>
<td>Rimg_1.jpg</td>
<td>16.3</td>
<td>14.2</td>
<td>10.9</td>
</tr>
<tr>
<td>Rimg_2.jpg</td>
<td>18.4</td>
<td>14.8</td>
<td>11.4</td>
</tr>
</tbody>
</table>

5.5.2.2 Evaluation on cluster quality

To evaluate the cluster quality, several important cluster validity measures have been proposed by researchers. Tan and Mat-Ia (2011) used two evaluation functions to evaluate the cluster quality. The first benchmark is Bezdek’s (1974), where the membership function is IT2FCM, so the evaluation function is defined as follows

\[ V_{PC} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{M} \left( \frac{\mu_{ij} + \mu_{ij}}{2} \right)^2 \]  

(5.40)

This benchmark is used to measure the fuzziness of a clustering result and the \( V_{PC} \) value can take on any value ranging from 0 to 1. From the context of validation, a good clustering algorithm must produce a better clustering result that is less fuzzy with larger \( V_{PC} \) value.

The second benchmark is the Xie and Beni (1991) function, where the membership function is IT2FCM, so the evaluation function is defined as follows
\[ V_{XB} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{M} \left( \frac{u_{ij}(x_i)}{u_{ij}(x_i) + u_{ij}(x_i)} \right) - c_j^2}{2 * N \min_{j \neq k} \{ c_j - c_k^2 \}} \] (5.41)

According to Xie and Beni, the \( V_{XB} \) should be decreased monotonically when the cluster number is close to the number of pixels in the image and furthermore, a better clustering result should produce smaller \( V_{XB} \) value. The \( V_{PC} \) values of the T1FCM, IT2FCM and Extended IT2FCM approaches are tabulated in Table 5.4. It shows that the proposed approach produces relatively larger \( V_{PC} \) values than other approaches, which shows that the general cluster distribution is better than other approaches.

**Table 5.4**  Comparison of results for different algorithms using ICICM in \( V_{PC} \) and \( V_{XB} \)

<table>
<thead>
<tr>
<th>Methods</th>
<th>T1FCM</th>
<th>IT2FCM</th>
<th>Extended IT2FCM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Images</td>
<td>( V_{PC} )</td>
<td>( V_{XB} )</td>
<td>( V_{PC} )</td>
</tr>
<tr>
<td>Txt_1.jpg</td>
<td>.625</td>
<td>.160</td>
<td>.698</td>
</tr>
<tr>
<td>Txt_2.jpg</td>
<td>.678</td>
<td>.192</td>
<td>.687</td>
</tr>
<tr>
<td>Rimg_1.jpg</td>
<td>.716</td>
<td>.231</td>
<td>.752</td>
</tr>
<tr>
<td>Rimg_2.jpg</td>
<td>.634</td>
<td>.201</td>
<td>.678</td>
</tr>
</tbody>
</table>

The \( V_{XB} \) values of the T1FCM, IT2FCM and Extended IT2FCM approaches are tabulated in Table 5.5. It shows that the proposed approach produces relatively smaller \( V_{XB} \) values than the other approaches.
Table 5.5  Comparison of results for different algorithms using ICICM segment count

<table>
<thead>
<tr>
<th>Methods</th>
<th>Type-1 FCM</th>
<th>IT2 FCM</th>
<th>Extended IT2FCM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CCM</td>
<td>ICICM</td>
<td>CCM</td>
</tr>
<tr>
<td>Txt_1.jpg</td>
<td>16</td>
<td>12</td>
<td>14</td>
</tr>
<tr>
<td>Txt_2.jpg</td>
<td>16</td>
<td>14</td>
<td>13</td>
</tr>
<tr>
<td>Rimg_1.jpg</td>
<td>10</td>
<td>08</td>
<td>09</td>
</tr>
<tr>
<td>Rimg_2.jpg</td>
<td>11</td>
<td>09</td>
<td>11</td>
</tr>
</tbody>
</table>

5.6  CONCLUSION

In this chapter various color texture feature extraction methods are studied. The ICICM is an integrated approach for capturing spatial variation of both color and intensity levels in the neighborhood of each pixel using HSV color space. In this work ICICM is used to extract the color feature from an image. Haralick introduced the 14 texture features for extracting texture from an image. But researchers have used only 5 of these 14 Haralick features, angular second moment (f1), contrast (f2), correlation (f3), inverse difference moment (f4) and variance (f5), in order to reduce the size of the feature space. Therefore, here also the same Haralick features are extracted from the image. Next IT2FCM was studied for clustering the features.

This chapter presents a new hybrid approach of Haralick feature extraction from ICICM and Extended IT2FCM for color texture image segmentation. The advantages of Extended IT2FCM are to generate accurate color descriptor using ICICM, and the spatial information is also included in the clustering process. In the conventional method, it is needed to calculate 3 CCM matrices and Haralick features are extracted from these matrices so the
size of the feature space is high. But in the proposed method, feature space is reduced because the 1X5 Haralick features are extracted from ICICM matrix. Therefore, from the experiment it is found that the computational complexity of IT2 fuzzy is slightly more when compared to Type-1 Fuzzy, but the computation time is reduced because the feature space is reduced.

The proposed method is evaluated and compared with conventional T1FCM and IT2FCM. The performance of the developed color texture segmentation algorithm has been evaluated on a large number of benchmark database images and natural images. The parameter value of HL=4 and GL=4 and \( r_1=0.05 \) and \( r_2=0.8 \) in ICICM and the parameter value of IT2FCM \( m_1=0.7 \) and \( m_2=0.9 \) produce a good number of segments in the experimental image. The results show that the proposed method is able to produce accurate segmentation results.