Wavelet adaptive quantization based color image segmentation

Anand Swaminathan [1], K.Venkata Subramaniyan [2], Tiruppathirajan G. [2], and Rajkumar J [2]

[1]Mepco Schlenk Engineering College
[2]Affiliation not available

October 30, 2023

Abstract

Image segmentation is an important pre-processing step towards higher level tasks such as object recognition, computer vision or image compression. Most of the existing segmentation algorithms deal with grayscale images only. But in the modern world, color images are extensively used in many situations. A new approach for color image segmentation is presented in this paper. There are many ways to deal with image segmentation problem and in these techniques; a particular class of algorithms traces their origin from region-based methods. These algorithms group homogeneous pixels, which are connected to primitive regions. They are easy to implement and are promising. Therefore, here one of the most efficient region-based segmentation algorithms is explained. The color image is quantized adaptively, using a wavelet transform. Then the region growing process is adopted. As preprocess, before actual region merging, small regions are eliminated by merging them with neighbor regions depending upon color similarity. After this, homogeneous regions are merged to get segmented output.

Wavelet adaptive quantization based color image segmentation

S. Anand, Professor. K.Venkata Subramaniyan, G.Tiruppathirajan, J.Rajkumar,
Department of Electronics and Communication Engineering
Mepco Schlenk Engineering College, Sivakasi, 626005, Tamilnadu, India
E-mail address: ks.an@yahoo.com; sanand@mepcoeng.ac.in;
Telephone: + 91-98421-49397; +91-4562-235410; Fax +91-4562-235111

ABSTRACT

Image segmentation is an important pre-processing step towards higher level tasks such as object recognition, computer vision or image compression. Most of the existing segmentation algorithms deal with grayscale images only. But in the modern world, color images are extensively used in many situations. A new approach for color image segmentation is presented in this paper. There are many ways to deal with image segmentation problem and in these techniques; a particular class of algorithms traces their origin from region-based methods. These algorithms group homogeneous pixels, which are connected to primitive regions. They are easy to implement and are promising. Therefore, here one of the most efficient region-based segmentation algorithms is explained. The color image is quantized adaptively, using a wavelet transform. Then the region growing process is adopted. As preprocess, before actual region merging, small regions are eliminated by merging them with neighbor regions depending upon color similarity. After this, homogenous regions are merged to get segmented output.

Index Terms: Wavelet, adaptive quantization, color image segmentation.

1 INTRODUCTION

For humans, an image is analog in nature and the human brain exploits it's "learn and adapt" property, to identify the regions from the visual data fed to it. This object identification is purely based on experience. But computer systems suffer from constructional complexity and memory constraints. So they process images in the digital domain. Also, human beings use their semantic and contextual knowledge to partition a scene. Thus, the goal of segmenting a natural image into meaningful objects in a semantic sense is often unrealistic. All these algorithms can be classified into four groups, pixel-based segmentation, edge-based segmentation, region-based segmentation, hybrid segmentation. Region-based methods have a cutting edge over other methods because they can be intuitively understood [1], [2], [3]. In addition to that, they are easy to implement. Most of the segmentation algorithms suffer from two types of error. They are (i) over-segmentation (ii) under segmentation. The cost of over-segmentation is less than that of under segmentation, because

in case of over segmentation the system has more chances to match. But in under segmentation, if any object boundary is missed, then the features extracted from the region will be corrupted [4], [5], [6]. A broad classification of segmentation algorithms is based on this [7]. The first type comprises algorithms, which trace the regions, from boundaries of regions, and the next type consists of algorithms, which directly deal with region interiors. The proposed method is classified in the second case. The algorithm enjoys the advantages of dealing directly with regions. It is easy to visualize and gives an intuitive picture of what is done. In this work, the image is quantized adaptively, using discrete wavelet transform. After this, the region growing is performed. Small island regions are eliminated by a preprocessing. Finally, region merging is carried out, to get segmented output.

2 Proposed Segmentation Algorithms

The outline of the paper is explained as follows.

- The image is decomposed using DWT. Depending upon the image complexity, the number of decomposition varies.
- Palette size is determined from two well defined numerical values which are calculated after the last stage of decomposition.
- The member palette colors are obtained from the RGB space, as the palette size is known.
- The image is quantized to these colors depending on the shortest Euclidian distance.
- Region growing is performed.
- Small regions are eliminated by merging with neighbors.
- Final region merging is performed depending on shape and color similarity analysis to get segmented output

Figure 1 shows the process diagram of the proposed method.

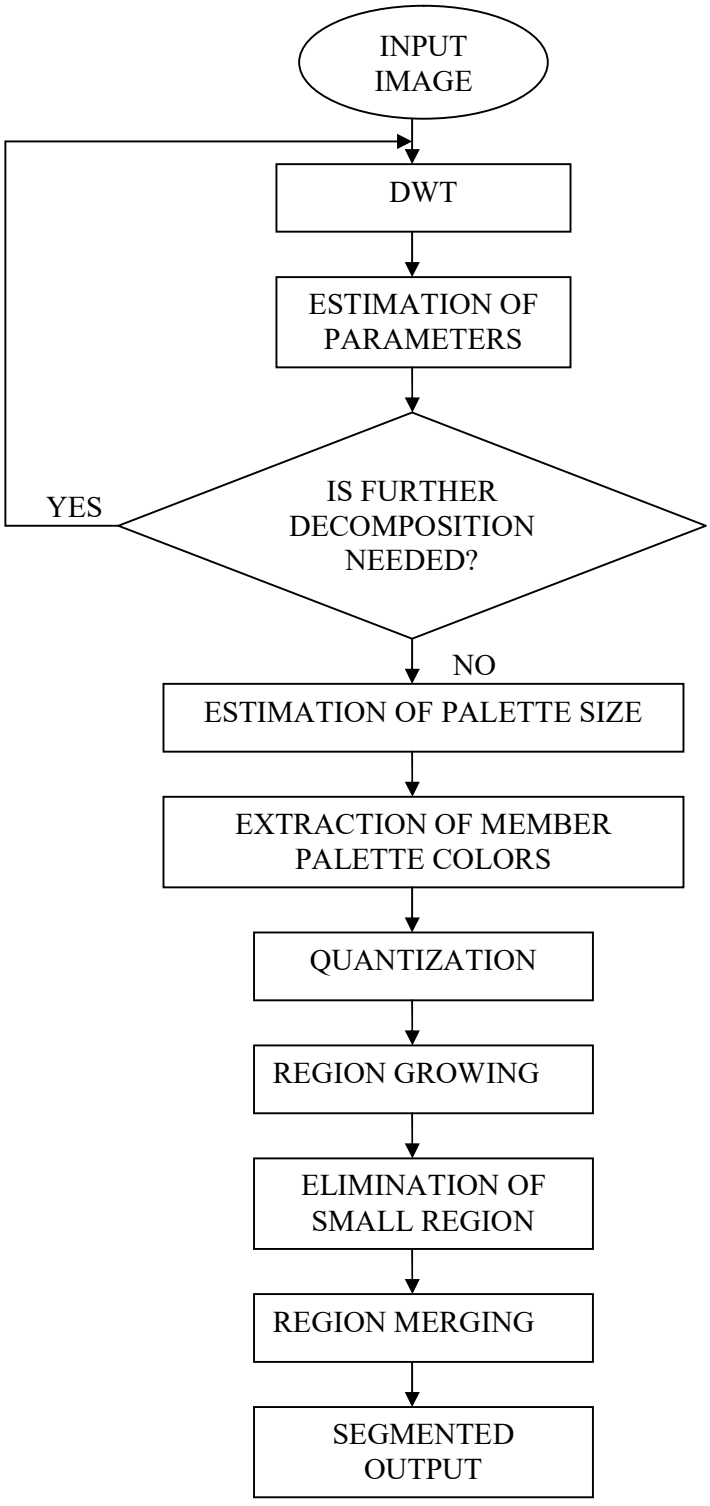

Figure 1 Process diagram of the proposed method

2.1 QUANTIZATION

The first step in the proposed framework is the quantization of the image. Many existing algorithms quantize the image, to get segmented output. But the way of quantization differs. Many algorithms have a rigid quantization module which is often proved to be inefficient. These algorithms quantize all types of images in the same way by allowing a fixed number of color bins. But efficiency is increased with an adaptive quantization module, which determines the number of color bins depending upon the image complexity.

- For smooth images, number of bins should be large and for sharp images, it should be less. So there is a requirement variation in the sharpness of the image.
- In this method, the sharpness complexity of the image is quantified into two numerical parameters. The number of bins is determined by these two parameters. From this point, if we mention palette size, it refers to the number of color bins, in this context.

A quasi-linear high-frequency model is proposed to determine the sharpness complexity of the image. This model uses discrete wavelet transform as the building block. The algorithm for this quasi linear high frequency model can be explained as,

STEP 1: transform the image in RGB color space to YUV color space.

STEP 2: apply two levels DWT on U and V images.

STEP 3: calculate the indication index and threshold.

STEP 4: if the indication index is lesser than threshold go to STEP 2 with approximation bands as inputs.

STEP 5: end.

The transformation to YUV color space is attributed to reducing the computational complexity, as in YUV color space, there is only two color information (U and V), whereas in RGB space, it is three (R, G and B). Eight sub-bands result from the DWT decomposition of U and V images.

They are named as follows.

U_{LL} ----------> Low-Low band of U-image

U_{LH} ----------> Low-High band of U-image

U_{HL} ----------> High-Low band of U-image

U_{HH} ----------> High-High band of U-image

V_{LL} ----------> Low-Low band of V-image

V_{LH} ----------> Low-High band of V-image

V_{HL} ----------> High-Low band of V-image

V_{HH} ----------> High-High band of V-image

Among these eight bands, except for the approximation bands, all remaining six bands have high frequency information. The root mean square of these six bands yields a chrominance image, C.

$$C(i,j)=\sqrt{(U_{LH}(i,j))^2+ U_{HL}(i,j))^2+ U_{HH}(i,j))^2+ V_{LH}(i,j))^2+ V_{HL}(i,j))^2+ V_{HH}(i,j))^2}$$

For $1\leq i \leq N$, $1\leq j \leq N$ (1)

Where N*N is the size of each sub band. The mean of the non-zero pixels in the current level chrominance image serves as a threshold.

$$\text{Threshold} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} C(i,j)}{\text{num}(C(i,j))} \quad ; C(i,j) \neq 0 \qquad (2)$$

Where num (.) is a function that returns the number of pixels. Indication index (I) is another parameter that is to be calculated, for K^{th} level of decomposition

$$I_k = \frac{\sum_{K=1}^{K}\sum_{i=1}^{N}\sum_{j=1}^{N} C(i,j)}{\text{num}(C_K(i,j))} \quad ; \text{for all K=1 to K.} \qquad (3)$$

If the indication index is less than the threshold one more level of DWT decomposition is performed. When indication index exceeds threshold, the standard deviation $\sigma_{k.}$ of the K^{th} level chrominance image is calculated.

$$\sigma_{norm} = \frac{\sigma_k}{\max(C_K(i,j)- I_k)} \qquad (4)$$

Where σ_{norm} is the normalized standard deviation. Let 'n' be the number of non-zero pixels is the final level, approximation bands. The palette size is determined by

$$\text{Palette size(P)} = f(\sigma_{norm}) * n; \qquad (5)$$

Where f(.) is piecewise step function.

Table 1 Piece wise step function

σ_{norm}	$f(\sigma_{norm})$
0-0.2	0.5
0.2-0.4	1
0.4-0.6	10
0.6-0.8	50
0.8-1	100

The maximum number of DWT decomposition will be $\log_2(N)$-1 for an N*N input image. After the determination of the palette size, we have to estimate the colors in the palette. For a palette size P, we have P^3 color bins. The center of each bin will be the palette member colors. Figure 2 explain the quantization concept in a 2-Dimensional plane. It can give an intuitive picture for 3-Dimensional plane

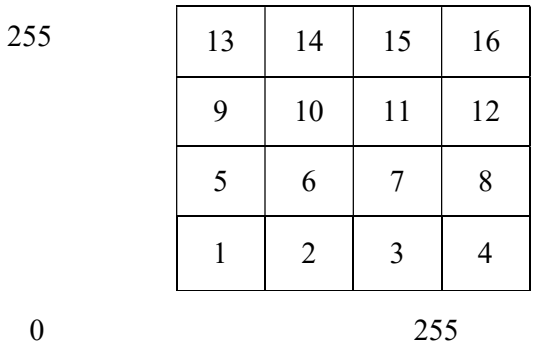

Figure 2 Examples for Quantization

1, 2….16 are centers of each bin and are the palette members. Any pixel whose color falls in any cell will be quantized to the center of the cell. In three dimension plane, a pixel having RGB coefficients as (160,192,200) will be quantized to (159,223,223). If P=4. The assignment is based on shorter a Euclidian distance calculation. Let d (P1, P2) be the distance between two points in the Euclidian color space.

$$d(P1, P2) = \sqrt{(P_{1R} - P_{2R})^2 + (P_{1G} - P_{2G})^2 + (P_{1B} - P_{2B})^2} \qquad (6)$$

Where P_{1R}, P_{1G}, P_{1B} are R, G and B component intensities of the first color.

P_{2R}, P_{2G}, P_{2B} are R, G and B component intensities of the second color.

Let C_1 ……..C_N be the centre of color bins. Where $N=P^3$; P=palette size.

The color C_p is quantized to C_Q if $\min(d(C_p, C_1), d(C_p, C_2)\ldots\ldots d(C_p, C_N)$ is $d(C_p, C_Q)$. By the introduction of a piece-wise step function, the quantization is made adaptive; this improves the efficiency of the quantization process. Thus the image is quantized to the required level based upon the complexity. The flowchart for the quantization is as in Figure 3 follows. Figure 4 shows an example quantized image.

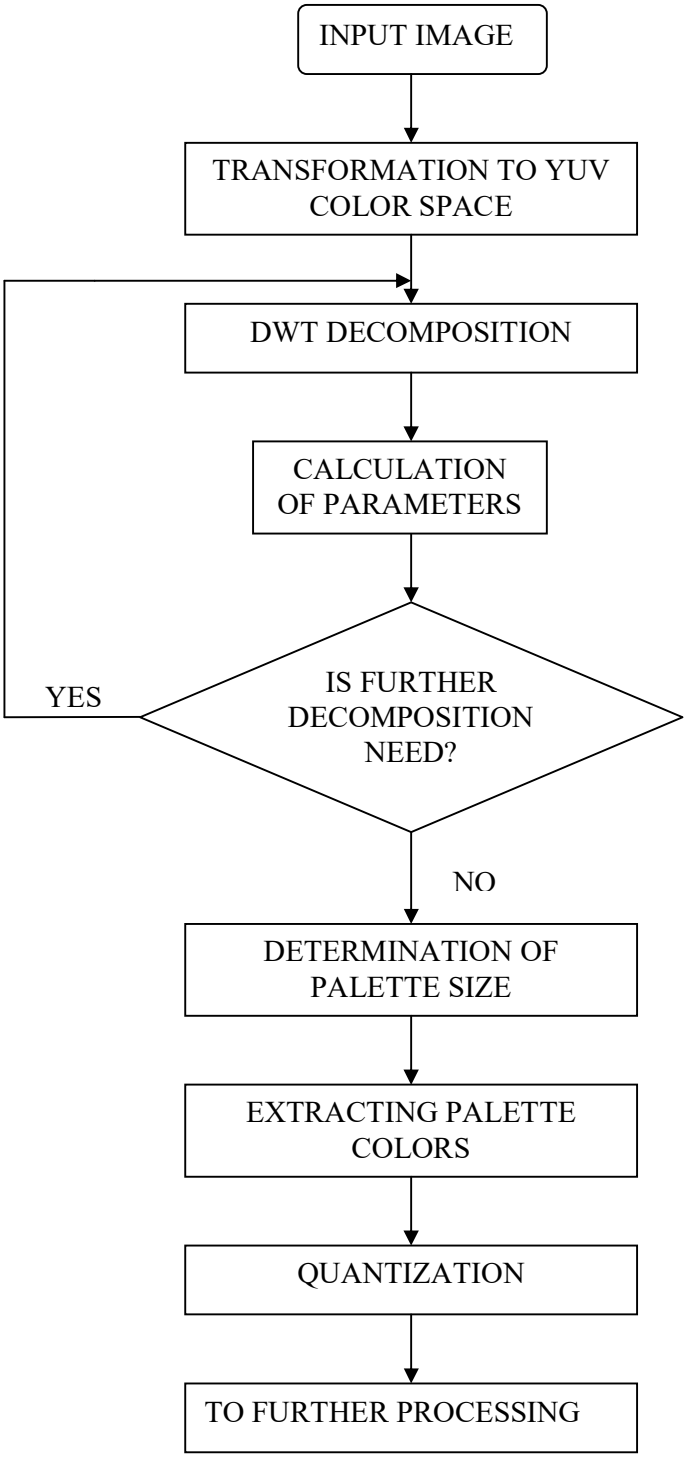

Figure 3 Flow chart for Quantization

(a) **(b)**

Figure 4 (a) Original image (b) Quantized image

2.2 REGION GROWING

After quantization, the next step in the proposed framework is region growing. Region growing is the process of dividing the image into regions based on certain criteria. The criterion in this paper is the color similarity of the pixels. The difference between existing region growing algorithms and this approach is the identification of region seeds. Also in existing algorithms, the seed lies in center of the region. But here, the seed lies in the boundary of the region. In many existing algorithms, the region seeds are identified before the region growing process. But we have developed an approach that identifies the seeds in the course of the algorithm. The computational complexity in region growth is reduced by compressing the three color planes namely R, G, B in one plane, where a numbered representation of colors is available. For example, if the palette size is 2, we get 8 colors in the quantized image, whereas in the original image the number of possible colors is 2^{24}. The RGB components of eight colors and their numbered representation is as in Table 2 follows,

Table 2 Representation of colors.

R	G	B	Color number
64	64	64	1
64	64	192	2
64	192	64	3
64	192	192	4
192	64	64	5

R	G	B	Color number
192	64	192	6
192	192	64	7
192	192	192	8

An example is given here for sake of understanding. Consider a 4×4 image with eight colors.

R-plane				G-plane				B-plane			
64	64	192	64	64	192	192	192	64	64	64	64
64	64	64	192	192	192	192	192	192	192	64	64
64	64	192	192	64	64	192	64	64	64	192	192
64	192	192	192	64	64	192	64	64	64	192	192

Numbered representation

1	3	7	3
4	4	3	7
1	1	8	6
1	5	8	6

Thus the three color information is compressed into one plane. Now starting from the position (1, 1), the pixels of the same color are grouped. The grouping process is explained pictorially as in Figure 5. The algorithm looks for the eight neighbor pixels starting from the pixel in the top, to the group.

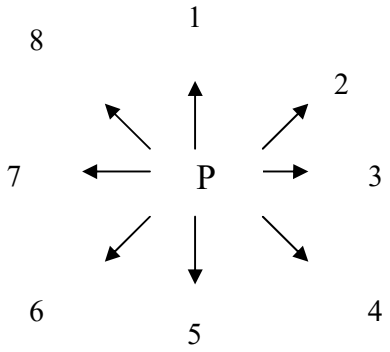

Figure 5 Order of neighbor pixels

Where P, is the pixel of interest. A matrix called mark is used to mark the pixels which are grouped already. A stack is used to store the position of the pixel when the control shifts from one pixel to its neighbor. A variable called counter which is initialized as 1 is used to denote region number. The region is a matrix that gives final result. Here the process is explained with the help of a 4×4 matrix.

Table 5.3 shows the steps involved in the region growing. The initial condition of the variables and input is,

$$\text{Mark} = \begin{matrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix} \qquad \text{region} = \begin{matrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix} \qquad \text{counter=1}$$

Let the input be,

$$\begin{matrix} 1 & 1 & 2 & 3 \\ 1 & 4 & 1 & 6 \\ 7 & 8 & 1 & 2 \\ 1 & 2 & 2 & 1 \end{matrix}$$

Table 3 Steps in the region growing

Marker	Region	Counter	Stack
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1	(1,1)
1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1	(1,1),(1,2)
1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0	1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0	1	(1,1),(1,2),(2,3)
1 1 0 0 0 0 1 0 0 0 1 0 0 0 0 0	1 1 0 0 0 0 1 0 0 0 1 0 0 0 0 0	1	(1,1),(1,2),(2,3),(3,3)
1 1 0 0 0 0 1 0 0 0 1 0 0 0 0 1	1 1 0 0 0 0 1 0 0 0 1 0 0 0 0 1	1	(1,1),(1,2),(2,3),(3,3),(4,4)

Marker	Region	Counter	Stack
1 1 0 0 0 0 1 0 0 0 1 0 0 0 0 1	1 1 0 0 0 0 1 0 0 0 1 0 0 0 0 1	1	(1,1),(1,2),(2,3),(3,3)
1 1 0 0 0 0 1 0 0 0 1 0 0 0 0 1	1 1 0 0 0 0 1 0 0 0 1 0 0 0 0 1	1	(1,1),(1,2),(2,3)
1 1 0 0 0 0 1 0 0 0 1 0 0 0 0 1	1 1 0 0 0 0 1 0 0 0 1 0 0 0 0 1	1	(1,1),(1,2)
1 1 0 0 1 0 1 0 0 0 1 0 0 0 0 1	1 1 0 0 1 0 1 0 0 0 1 0 0 0 0 1	1	(1,1),(1,2),(2,1)
1 1 0 0 1 0 1 0 0 0 1 0 0 0 0 1	1 1 0 0 1 0 1 0 0 0 1 0 0 0 0 1	1	(1,1),(1,2)
1 1 0 0 1 0 1 0 0 0 1 0 0 0 0 1	1 1 0 0 1 0 1 0 0 0 1 0 0 0 0 1	1	(1,1)
1 1 0 0 1 0 1 0 0 0 1 0 0 0 0 1	1 1 0 0 1 0 1 0 0 0 1 0 0 0 0 1	2	-----
1 1 1 0 1 0 1 0 0 0 1 0 0 0 0 1	1 1 2 0 1 0 1 0 0 0 1 0 0 0 0 1	2	(1,3)
1 1 1 0 1 0 1 0 0 0 1 0 0 0 0 1	1 1 2 0 1 0 1 0 0 0 1 0 0 0 0 1	3	-----

Marker	Region	Counter	Stack
1 1 1 1 1 0 1 0 0 0 1 0 0 0 0 1	1 1 2 3 1 0 1 0 0 0 1 0 0 0 0 1	3	(1,4)
1 1 1 1 1 0 1 0 0 0 1 0 0 0 0 1	1 1 2 3 1 0 1 0 0 0 1 0 0 0 0 1	4	-----
1 1 1 1 1 1 1 0 0 0 1 0 0 0 0 1	1 1 2 3 1 4 1 0 0 0 1 0 0 0 0 1	4	(2,2)
1 1 1 1 1 1 1 0 0 0 1 0 0 0 0 1	1 1 2 3 1 4 1 0 0 0 1 0 0 0 0 1	5	-----
1 1 1 1 1 1 1 1 0 0 1 0 0 0 0 1	1 1 2 3 1 4 1 5 0 0 1 0 0 0 0 1	5	(2,4)
1 1 1 1 1 1 1 1 0 0 1 0 0 0 0 1	1 1 2 3 1 4 1 5 0 0 1 0 0 0 0 1	6	-----
1 1 1 1 1 1 1 1 1 0 1 0 0 0 0 1	1 1 2 3 1 4 1 5 6 0 1 0 0 0 0 1	6	(3,1)
1 1 1 1 1 1 1 1 1 0 1 0 0 0 0 1	1 1 2 3 1 4 1 5 6 0 1 0 0 0 0 1	7	-----
1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 1	1 1 2 3 1 4 1 5 6 7 1 0 0 0 0 1	7	(3,2)

Marker	Region	Counter	Stack
1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 1	1 1 2 3 1 4 1 5 6 7 1 0 0 0 0 1	8	-----
1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1	1 1 2 3 1 4 1 5 6 7 1 8 0 0 0 1	8	(3,4)
1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1	1 1 2 3 1 4 1 5 6 7 1 8 0 0 8 1	8	(3,4),(4,3)
1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1	1 1 2 3 1 4 1 5 6 7 1 8 0 8 8 1	8	(3,4),(4,3),(4,2)
1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1	1 1 2 3 1 4 1 5 6 7 1 8 0 8 8 1	8	(3,4),(4,3)
1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1	1 1 2 3 1 4 1 5 6 7 1 8 0 8 8 1	8	(3,4)
1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1	1 1 2 3 1 4 1 5 6 7 1 8 0 8 8 1	9	-----
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1 1 2 3 1 4 1 5 6 7 1 8 9 8 8 1	9	(4,1)

Thus we get the region grown output. This will be used in further processing. The algorithm explained above can be implemented easily using the recursive methodology.

2.3 ELIMINATION OF SMALL REGIONS

In the previous section, the region growing process is explained in detail. This section is dedicated to provide an understanding of preprocessing before the final region merging. This preprocessing is essential for reducing the computational complexity in the final step of region merging. The elimination of very small regions results in an output (flow shown in Figure 6) which is visually smooth. Also, color patches and island regions, which don't contribute much information in the image, are merged with neighboring regions. The small regions are merged with one of its neighboring regions, based on the shortest Euclidian distance concept. In crude words, this may be explained as the small region is merged with one of its neighbors whose color component is more similar to the color of the small region, than any other neighbor regions. In this paper, a new parameter is explained for calculating the color similarity. This parameter is called as "color similarity parameter" (CSP).

$$CSP = 1 - \frac{\sqrt{(R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2}}{255\sqrt{3}} \qquad (5.7)$$

Where,

R_1, G1, B1 are color components of region 1.

R_2, G2, B2 are color components of region 2.

The numerical value of $255\sqrt{3}$ corresponds to the Euclidian distance between black and white. This value is used for normalization. The subtraction from unity is to provide a direct proportion between color similarity and CSP. So, more the CSP more is the color similarity. A region is determined as "small", if it is less than a size threshold. This size threshold is tunable. The size threshold of 1 produces no change.

$1 \leq$ size threshold $\leq N^2$ for N*N image.

This elimination of small regions module explained by the flow chart below.

The assumption here is, there are 'T' regions.

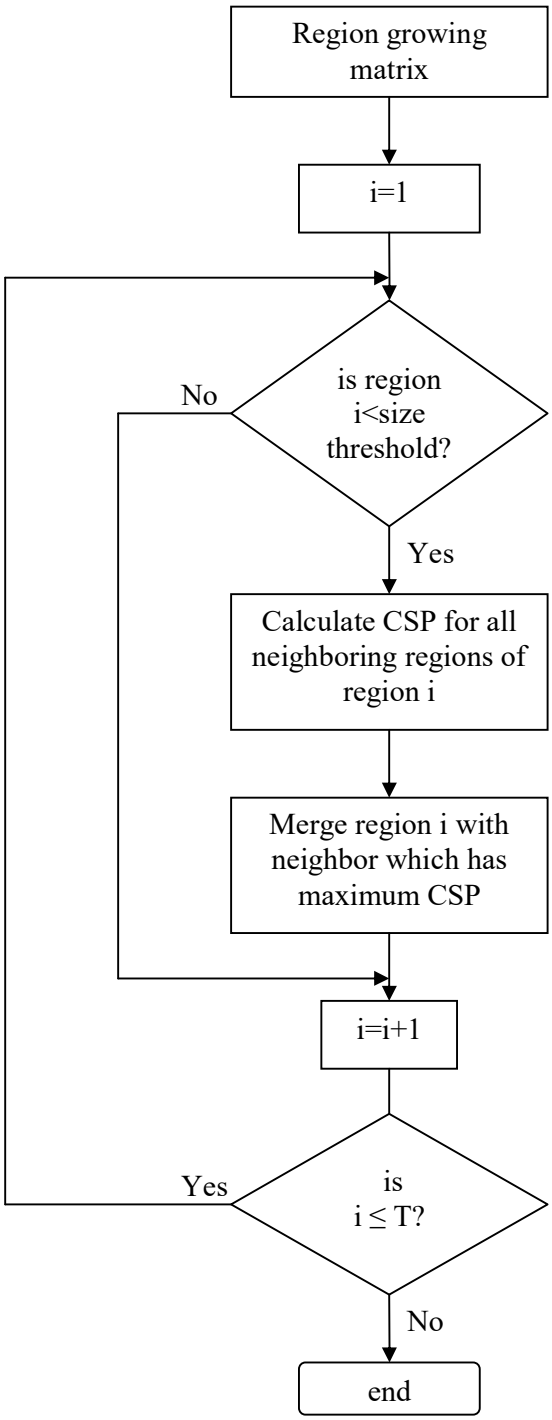

Figure 6 Flow chart for the elimination of small regions

This module produces an output with S number of regions. Where $S \leq T$. the output of this module is the input for the actual region merging module, which produces the segmented output Figure 7.

Figure 7 Result after elimination of small regions

2.4 REGION MERGING

This is the final step in the proposed method. This produces a segmented output. The region merging is performed between two regions if they satisfy a criterion. The criterion proposed is a novel one and incorporates the condition that the two regions are expected to satisfy. The condition is shape and color similarity. The intensity of the similarity can be viewed by a parameter which is called Merging Parameter (MP). MP = CSP*SSP. Where CSP → Color Similarity Parameter; SSP → Shape Similarity Parameter. The two regions are merged if MP is more than a Threshold. This Threshold is tunable to; $0 \leq$ Threshold ≤ 1. This Threshold is set to an optimum level to obtain the segmented output. Thus various levels of segmentation can be achieved by varying this threshold. For smooth images, the threshold is set nearly to 0.8 and for sharp images, it is set nearer to 0.6. Lesser the threshold, more merging will occur. This threshold is the key to the proposed algorithm, and setting a smaller threshold than the required one, will produce under segmented image. A larger threshold will produce over-segmented image. The CSP is explained in the previously. The shape similarity parameter mentioned above, is numerically the normalized ratio of boundary a region shares with its neighbor.

For example, let us consider a 4*4 image.

1	1	1	2
7	3	4	1
5	3	3	4
5	3	6	7

Region 1 has 13 pixels as neighbors. The pixel (1, 1) has 1 neighbor; The pixel (1, 2) has 2 neighbors; The pixel (1, 3) has 3 neighbors; The pixel (2, 1) has 3 neighbors, The pixel (2, 4) has 4 neighbors. So totally there are 13 neighbors. Among them region 1 has region 2, two times as its neighbor. So SSP of region 2 with respect to region 1 is 2/13. The SSP and CSP parameters are calculated for all neighbors of a region, and MP is calculated for all of them. If their MP satisfies a threshold they are merged. This process is continued for all regions, to get the final segmented output. Let there be S regions after the elimination of small regions. The flow chart for this module can be explained as below in Figure 9. The output of this region merging process is the segmented output Figure 8. This approach is one of the most efficient methods to segment color images as it involves both checking of color and shape similarity.

Figure 8 Segmented images

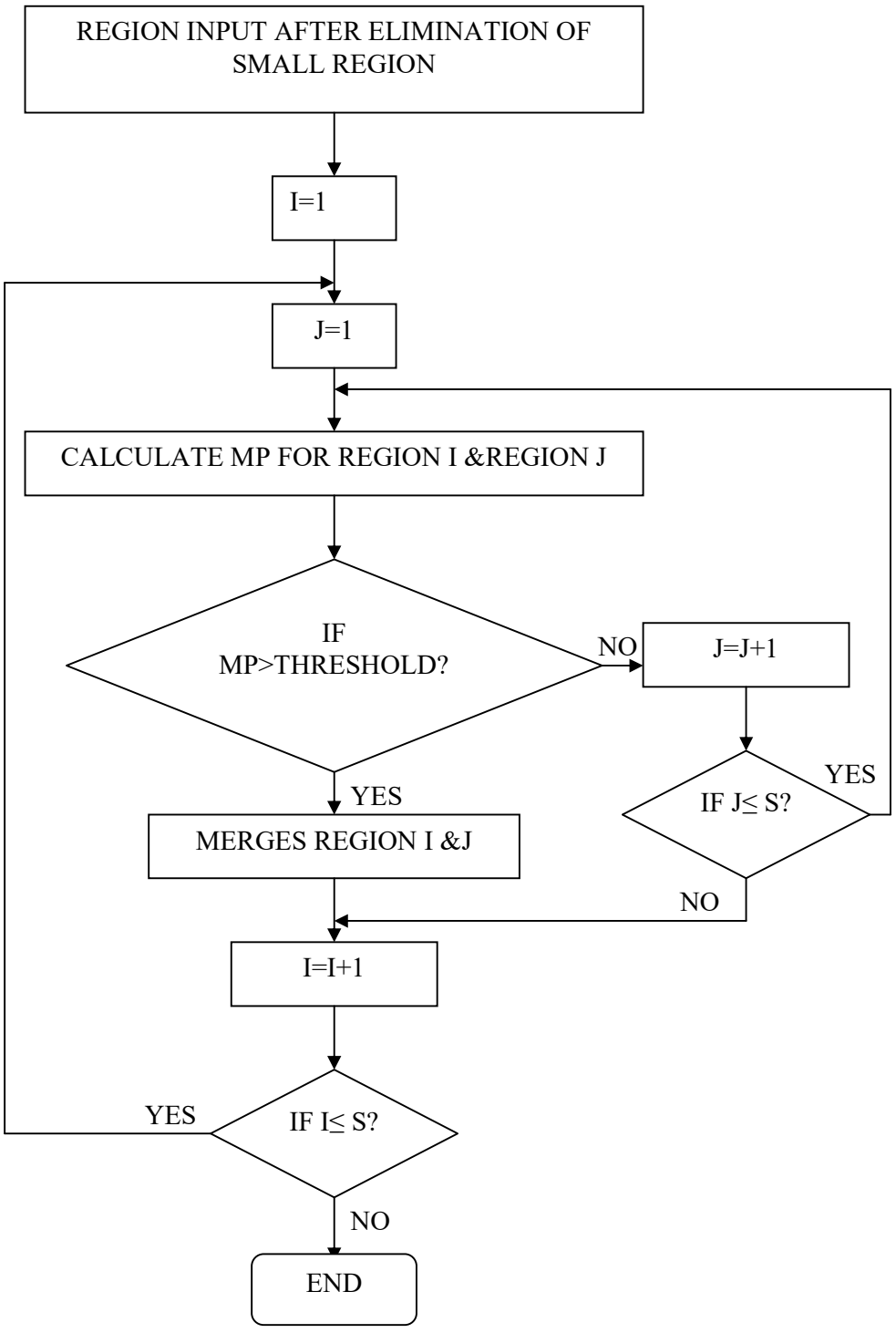

Figure 9 Flow chart for region merging

3. RESULTS AND CONCLUSION

The input data set contains more than 300 images [8] of size 256×256. Some of the results are presented in Figure 10, Figure 11, and Figure 12 as shown below.

10. (a) Original image-caps

10 (b) Quantized image-caps

10. (c) Resulting image after elimination of small regions-caps

10. (d) Segmented image-caps

Fig

Figure 11 (a) Original image-bird

Figure 11 (b) Quantized image-bird

Figure 11 (c) Result after elimination of small regions-bird

Figure 11 (d) Segmented image-bird

Fig

Figure 12 (a) Original image-baloon

Figure 12 (b) Quantized image-baloon

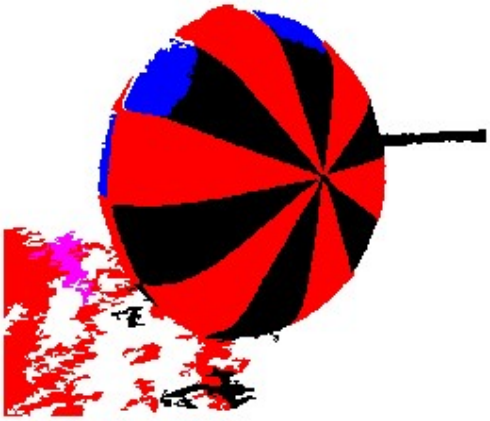

Figure 12 (c) Result after elimination of small regions-baloon

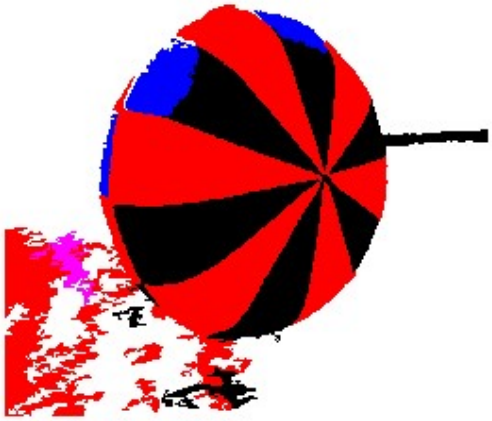

Figure 12 (d) Segmented image-baloon

4 CONCLUSIONS

The method adopted here, for the segmentation of color images is a new approach and one leap works towards perfection in this domain. The threshold used in this paper is tunable and one threshold is calculated automatically. The future of this paper lies in automatic settings of this threshold and we are working on it. The efficiency of the method can be dramatically improved if texture similarity is also considered. Energy measures are to be used for texture similarity analysis.

This algorithm works well for images with low depth of space. Due to this fact, this method can be implemented in practical domains such as moving object recognition, target acquisition, biometric based security system, missile defense shields etc.

REFERENCES

[1]. Siddesha, S., S. K. Niranjan, and VN Manjunath Aradhya. "A Study of Different Color Segmentation Techniques for Crop Bunch in Arecanut." *Environmental and Agricultural Informatics: Concepts, Methodologies, Tools, and Applications*. IGI Global, 2020. 1078-1105.

[2]. Razalli, Husniza, Rusyaizila Ramli, and Mohammed Hazim Alkawaz. "Emergency Vehicle Recognition and Classification Method Using HSV Color Segmentation." *2020 16th IEEE International Colloquium on Signal Processing & Its Applications (CSPA)*. IEEE, 2020.

[3]. Chan, Din-Yuen, Chih-Hsueh Lin, and Wen-Shyong Hsieh. "Image segmentation with fast wavelet-based color segmenting and directional region growing." *IEICE transactions on information and systems* 88.10 (2005): 2249-2259.

[4]. Jing, Feng, et al. "Unsupervised image segmentation using local homogeneity analysis." *Proceedings of the 2003 International Symposium on Circuits and Systems, 2003. ISCAS'03.*. Vol. 2. IEEE, 2003.

[5]. Premkumar, R., and S. Anand. "Secured and compound 3-D chaos image encryption using hybrid mutation and crossover operator." *Multimedia Tools and Applications* 78.8 (2019): 9577-9593.

[6]. Anand, S., and NM Mary Sindhuja. "Spot edge detection in microarray images using balanced GHM multiwavelet." *2009 International Conference on Control, Automation, Communication and Energy Conservation*. IEEE, 2009.

[7]. Gonzalez, Rafael C., and Richard E. Woods. "Digital Image Processing (preview)." (2002).

[8]. http://perso.Wanadoo.fr/polyvalens/clemens/wavelets/wavelets.html.