Indonesian Journal of Electrical Engineering and Computer Science
Vol. 3, No. 2, August 2016, pp. 377 ~ 382
DOI: 10.11591/ijeecs.v3.i2.pp377-382

An Image Processing Method to Convert RGB Image into Binary

Ratri Dwi Atmaja*, Muhammad Ary Murti, Junartho Halomoan, Fiky Yosef Suratman
School of Electrical Engineering, Telkom University
Jalan Telekomunikasi no.1, Terusan Buah Batu, Bandung 40257, Indonesia
*Corresponding author, e-mail: ratridwiatmaja@telkomuniversity.ac.id

Abstract

It is important in image processing to extract objects from their background into binary image. Binary image is used as input to feature extraction process and have an important role in generating unique feature to distinguish several classes in pattern recognition. This paper proposes an image processing algorithm to obtain a binary image from RGB. The results showed that the binary image of the proposed algorithm contained the desired object.

Keywords: *image processing method, convert RGB to binary*

1. Introduction

In the image processing, image binarization is used as a general tool for image segmentation of discriminating objects from background in various applications such as automatic target tracking, object recognition, image compression, image analysis, and object separation [4]. Some recent researchs i.e. [7] need a perfect binary image to separate objects before doing classification and [3] also need a perfect binary image to extract wood fiber from background. Because of it's important role, many researchs of image binarization have been developed from Otsu algorithm [1] and appear some image segmentation techniques as presented in [2]. This paper is a continuation of [3] and [6] to distinguish several classes of wood in object recognition but it's still having difficulty to get a perfect binary image. This maybe one reason why [3] still has not reached maximum performance. So this paper proposes an image processing method to convert RGB image into binary.

2. Research Method

Figure 1 is the flowchart of proposed algorithm. After loading the image, it needs to convert the image to grayscale. Then determine the *histogram* of grayscale image.

Histogram is 1-dimensional matrix that is used to represent the pixel intensity in frequency distributions. *Statisticparameter* is a value that is used to determine the number of image as segmentation result. We uses the mean and standard deviation as a reference for determining *statisticparameter.* Mean and standard deviation is given by [5]:

$$\bar{X} = \frac{X_1+X_2+X_3+\cdots+X_N}{N} = \frac{\sum X}{N}$$

$$\sigma = \sqrt{\sigma^2} = \sqrt{\frac{\sum(X-\bar{X})^2}{N}}$$

Where $\bar{X}$ represents the sample mean, σ represents standard deviation, X represents individual value, and *N* represents the total number of values in the sample. *Matrix* on Figure 1 is a 1-dimensional binary matrix with the size 1 x 256. Figure 2 is the way how to find *matrix* and figure 3 is example to understand the way for finding *matrix* easily.

Received March 2, 2016; Revised June 25, 2016; Accepted July 10, 2016

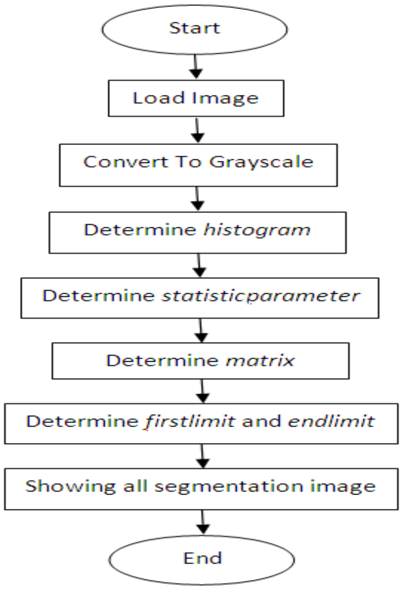


Figure 1. The Flowchart of Proposed Algorithm

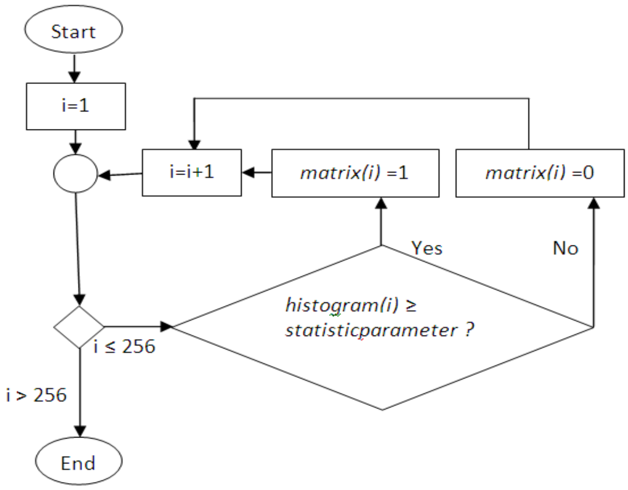


Figure 2. Flowchart to Find *Matrix*

0	1	0	3
2	3	0	0
1	1	3	3
0	1	1	0

(a)

6	5	1	4

(b)

1	1	0	1

(c)

Figure 3. Example of 2 Bit Image with the size 4 x 4 (b) The *Histogram* and (c) The *Matrix* using Mean as *Statisticparameter* (mean = 4)

The next process is finding *firstlimit* and *endlimit. Firstlimit* is 1-dimensional matrix that contains the first index of the selected range at histogram. While *endlimit* is 1-dimensional matrix that contains the end index of the selected range at histogram. This selected range is obtained from *matrix* that contains pixel 1 and represents the pixel intensity whose occurrence is greater than the specified *statisticparameter*. From Figure 3, it is obtained *firstlimit* = [0 3] and *endlimit* = [1 3]. It means that there are 2 selected ranges at histogram, i.e. from 0 to 1 and from 3 to 3. Figure 4 is the way how to find *firstlimit* and *endlimit.*

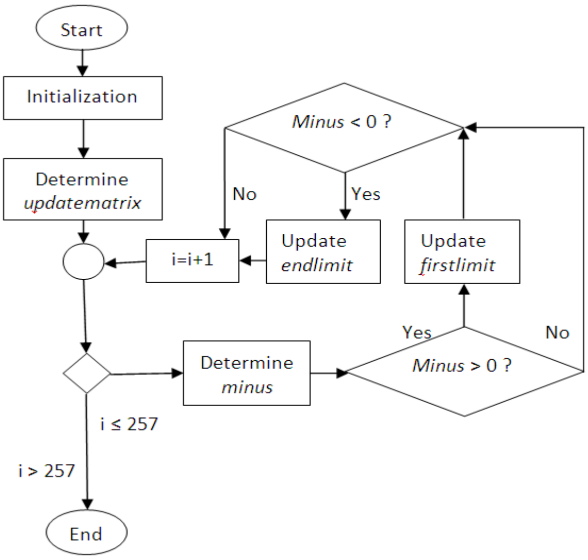


Figure 4. Flowchart to Find *Firstlimit* and *Endlimit*

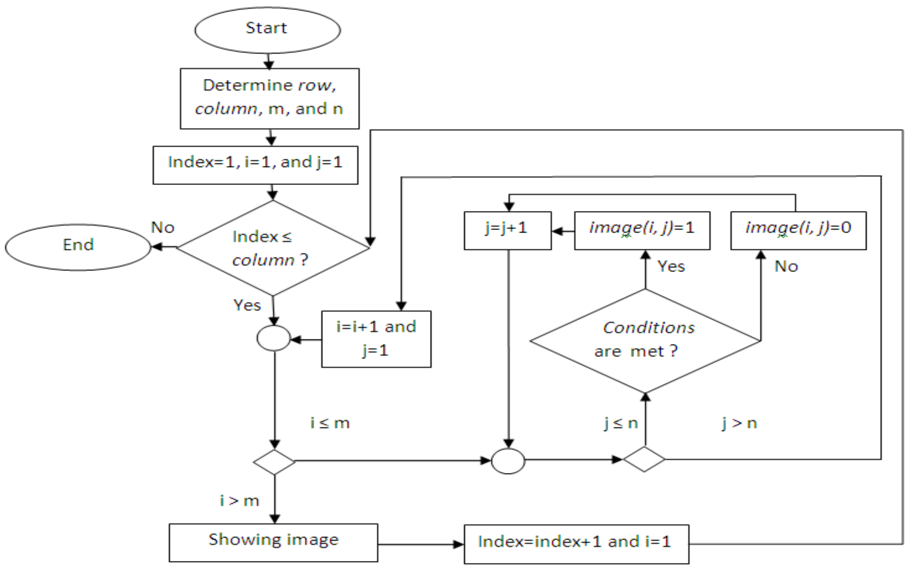


Figure 5. Flowchart to Find and Show All *Images*

In Figure 4, do the initialization as below:

firstlimit=[];
endlimit=[];

The initialization is intended to make the template of *firstlimit* and *endlimit*. *Updatematrix* is 1-dimensional matrix formed by adding 0 at first index of *matrix*. From Figure 3(c) it is obtained *updatematrix* = [0 1 1 0 1]. *Minus* can be found with this equation:

Minus = Updatematrix(i) - Updatematrix(i-1)

Then update *firstlimit* dan *endlimit* with this equation:

firstlimit = [*firstlimit* (i-2)]
endlimit = [*endlimit* (i-3)]

The next process is showing all segmentation images that can be found in Figure 5. *Image* on Figure 5 is binary image results from segmentation and can be more than one. *Row* x *column* is the size of *firstlimit*, while m x n is the size of grayscale image on Figure 1. *Conditions* are defined by the following rule:

grayscale (i , j) ≥ firstlimit (index) AND *grayscale (i , j) ≤ endlimit (index)*

Where *grayscale* is grayscale image on Figure 1.

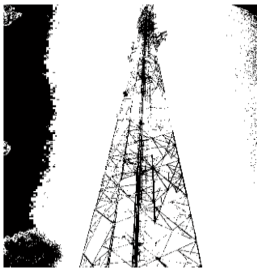
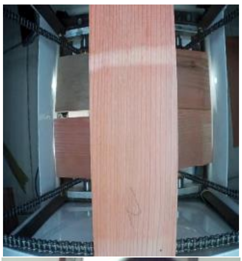
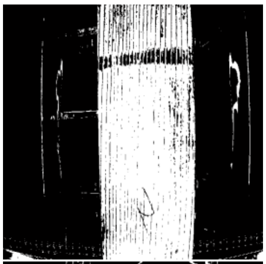
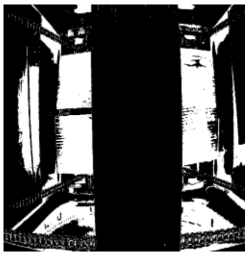
3. Results and Analysis

To see performance of the proposed algorithm, the experiments are done with some *statisticparameter* values. Table 1 is the *statisticparameter* and the number of generated binary image, while Table 2 is the original image and generated binary image.

Table 1. *Statisticparameter* and the Number of Generated Binary Image

No	Image	*Statisticparameter* value	The number of generated binary image	Is the desired object still exists on one of the binary image?
1	Image A	$\bar{X}$	4	Yes
		$\bar{X}$ + 0.5 σ	9	Yes
		$\bar{X}$ + σ	9	Yes
		$\bar{X}$ + 1.5 σ	8	No
		$\bar{X}$ + 2 σ	6	No
		$\bar{X}$ + 2.5 σ	4	No
		$\bar{X}$ + 3 σ	2	No
2	Image B	$\bar{X}$	11	Yes
		$\bar{X}$ + 0.5 σ	3	Yes
		$\bar{X}$ + σ	3	Yes
		$\bar{X}$ + 1.5 σ	1	Yes
		$\bar{X}$ + 2 σ	1	Yes
		$\bar{X}$ + 2.5 σ	1	Yes
		$\bar{X}$ + 3 σ	1	No
3	Image C	$\bar{X}$	4	No
		$\bar{X}$ + 0.5 σ	5	Yes
		$\bar{X}$ + σ	3	Yes
		$\bar{X}$ + 1.5 σ	3	Yes
		$\bar{X}$ + 2 σ	3	No
		$\bar{X}$ + 2.5 σ	3	No
		$\bar{X}$ + 3 σ	2	No
4	Image D	$\bar{X}$	9	Yes
		$\bar{X}$ + 0.5 σ	8	Yes
		$\bar{X}$ + σ	7	No
		$\bar{X}$ + 1.5 σ	3	No
		$\bar{X}$ + 2 σ	2	No
		$\bar{X}$ + 2.5 σ	1	No
		$\bar{X}$ + 3 σ	1	No

Table 2. The original image and generated binary image

Image	Original Image	One of the generated binary image that contains the desired object	Another generated binary image
A			
B			
C			
D			

4. Conclusion

The algorithm still generates more than one binary image is the weakness of this research. However, each sample has one of the results that contains the desired object. It is important in next research to automatically select only one binary image that contains the desired object. So this algorithm can be developed and can be used to convert RGB image to binary as needed.

References

[1] Nobuyuki Otsu. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*. 1979; 9(1): 62-66.

[2] Hetal J Vala, Astha Baxi. A Review on Otsu Image Segmentation Algorithm. *International Journal of Advanced Research in Computer Engineering & Technology*. 2013; 2(2): 387-389.

[3] Ratri Dwi Atmaja, et al. The Detection of Straight and Slant Wood Fiber Through slop angle fiber feature. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2015; 14(2): 318-322.

[4] Kwang Baek Kim. Image Binarization using Intensity Range of Grayscale Images. *International Journal of Multimedia and Ubiquitous Engineering*. 2015; 10(7): 139-144.

[5] Allan G Bluman. Seventh Edition of Elementary Statistics: A Step by Step Approach. McGraw-Hill. 2009.
[6] Ratri Dwi Atmaja. Wood image real-time segmentation algorithm based on video processing. *International Journal of Imaging and Robotics*. 2015; 15(1): 12-18.
[7] Ratri Dwi Atmaja, et al. One Layer Object Separation Algorithm in Binary Image. *Indonesian Journal of Electrical Engineering and Computer Science*. 2016; 1(1): 215-220.