Journal of Engineering and Applied Sciences 13 (Special Issue 11): 8686-8689, 2018
ISSN: 1816-949X

Binary Median Filter Speed Optimization Using An Integral Image

[1]Sang-Ha Lee, [1]Jisang Yoo and [2]Soon-Chul Kwon
[1]Department of Electronic Engineering,
[2]Department of Smart Systems, Kwangwoon University, Seoul, Korea

Abstract: In the computer vision, there are various post-processing algorithms to improve the performance of image processing algorithms. In this study, our goal is a speed optimization for the median filter for binary images. To output a median value, a sorting algorithm is used. The sorting algorithm has a common feature that the throughput exponentially increases as the amount of data to be sorted increases. we proposed a speed optimization method of a median filter for a binary image using an integral image. The feature of a binary image is that its pixels consist of 0 and 1 values. Using this feature, instead of using a sorting algorithm to output a median value, the median value can be calculated by calculating the number of 1's in the mask area using the integral image.

Key words: Image enhancement, filter operation, optimization, sorting algorithm, integral image, binary image

INTRODUCTION

In the computer vision, there are various post-processing algorithms to improve the performance of image processing algorithms. Examples of such algorithms are morphology filters such as open and close operations, hole filling filters to fill holes caused by occlusion in stereo vision and median filters for an image enhancement. In this study, our goal is a speed optimization for the median filter for binary images. In general an object detection algorithm such as circle, ellipse and human outputs a binary image of a target object. In order to improve the quality of the output image, a morphology filter and a median filter are applied to the binary image which is the output of the object detection algorithm. Figure 1 shows an example of image enhancement through median filter a median filter uses a sliding window method and outputs a middle value among the values in the mask. This filter is mainly used to fill the noise or holes in the image. In order to improve the performance of the filter, various filters based on this filter have been proposed (Huang *et al*., 1979; Glover, 1992; Ze and Zhong, 1996; Ataman *et al*., 1980; Perreault and Patrick, 2007; Yang *et al*., 2015). To output the median value, a sorting algorithm is used. Types of sorting algorithms include bubble sorting, insertion sorting and quick sorting. These sorting algorithms have a common feature that the throughput exponentially increases as the amount of data to be sorted increases. There are many studies to solve this problem. In this study, we proposed a speed optimization method of a median filter for a binary image using an integral image. The feature of a binary image is that its pixels consist of 0 and 1 values. Using this feature, instead of using a sorting algorithm to output

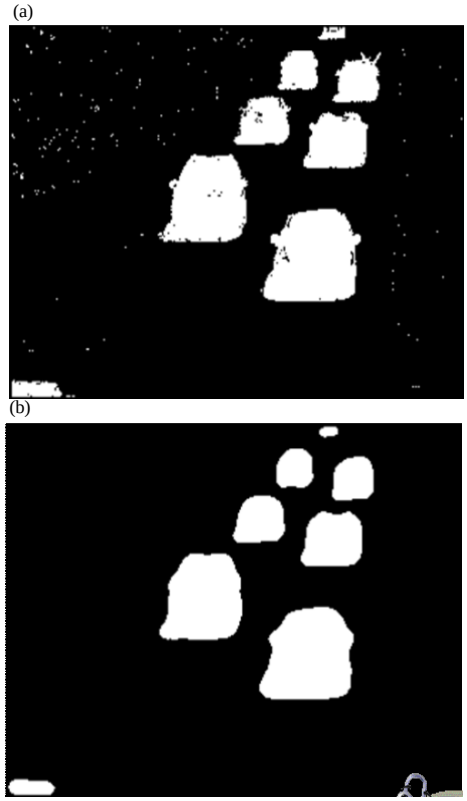

Fig. 1: a) Input binary image and b) Output binary image for apply 9×9 median filter

Corresponding Author: Soon-Chul Kwon, Department of Smart Systems, Kwangwoon University, Seoul, Korea

a median value, the median value can be calculated by calculating the number of 1's in the mask area using the integral image.

MATERIALS AND METHODS

Design and simulation: Figure 2 shows a flow chart of the proposed algorithm. The proposed algorithm consists of two modules. The integral image generator is a module that generates an integral image of the input binary image. The comparator is a module that outputs the median value by comparing the total number of 1s calculated using the integral image with the total area of the mask. Details of the integral image generator module are described and details of the comparator module are described in this study.

Integral image generator: This module calculates the integral image computed from the input binary image. In computer vision it was popularized by Lewis (1995) and then given the name "integral image" and used within the Viola Jones object detection framework. An integral image is a data structure for quickly and efficiently calculating the sum of values in a rectangular area. The formula for calculating the integral image is shown in Eq. 1:

$$\text{Integral image: } I(x, y) = \sum_{\substack{x' \le x \\ y' \le y}} i(x', y') \quad (1)$$

Where:

x,y = The pixel coordinates of the image
$I(x', y')$ = The input image
I(x, y) = The output image

The integral image is formed by summing all the values in the rectangular area formed by the current coordinates (x, y) based on the image coordinates (0, 0). Figure 3 shows the input binary image and the integral image computed through it. Using an integral image, we can quickly find the sum of pixels within a specific rectangular area:

$$\text{Sum}_{\text{rect}}\left((x_1, y_1),(x_2, y_2)\right) = \sum_{y'=y_1}^{y_2} \sum_{x'=x_1}^{x_2} i(x', y') \quad (2)$$

$$\text{Sum}_{\text{rect}}\left((x_1, y_1),(x_2, y_2)\right) = I(x_2, y_2) - I(x_2, y_1-1) - I(x_1-1, y_2) + I(x_1-1, y_1-1) \quad (3)$$

As shown in Eq. 2 it is common to approach all the pixels within a rectangular area to obtain a sum. This method has a disadvantage that the calculation amount increases as the rectangular area to be calculated is wider. Using an integral image, we can simply calculate the pixels

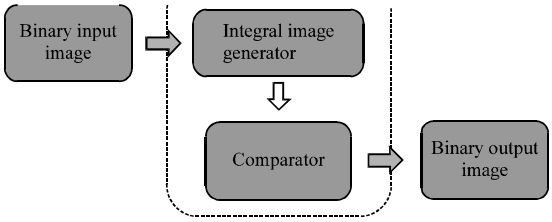

Fig. 2: Flow chart showing the proposed algorithm

(a)

1	0	1	1	1	1	1	0
0	0	1	1	1	1	1	0
0	0	0	0	0	0	1	1
0	0	0	0	0	1	1	1
0	0	1	1	1	1	1	1
0	0	1	1	1	1	1	1
0	0	0	1	1	0	0	0
0	0	0	1	1	0	0	1

(b)

1	1	2	3	4	5	6	7
1	1	3	5	7	9	11	11
1	1	3	5	7	9	12	13
1	1	3	5	7	10	14	16
1	1	4	7	10	14	19	22
1	1	5	9	13	18	24	28
1	1	5	10	15	20	26	30
1	1	5	11	17	22	28	33

Fig. 3: a) The input binary image and b) Integral image computed through it

in the rectangular region through Eq. 3. Figure 4 shows the example. Only four values (yellow, green, purple, blue) are used to obtain the pixel sum in any rectangular area. Once the integral image has been computed, evaluating the sum of intensities over any rectangular area requires exactly four array references regardless of the area size.

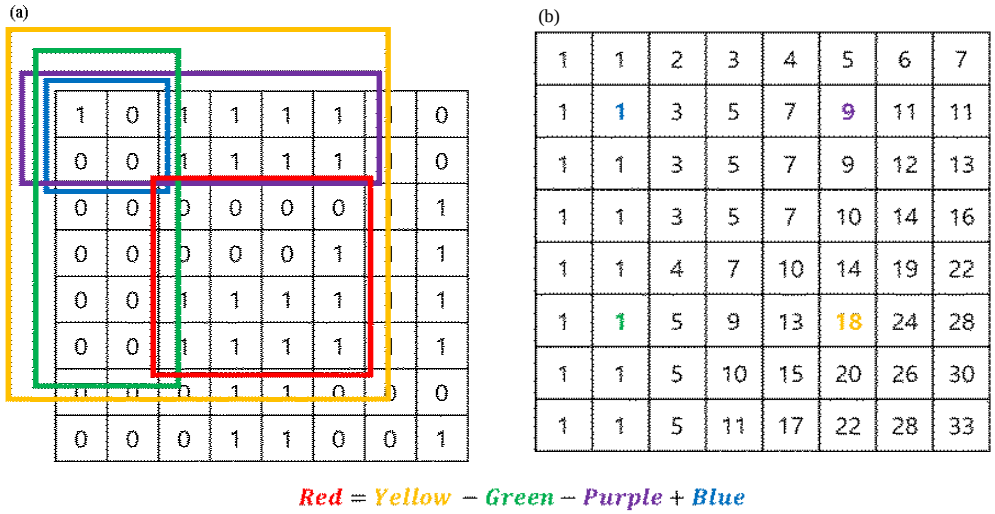

Fig. 4: a, b) The example of obtaining the pixel sum of a specific rectangular area using an integral image

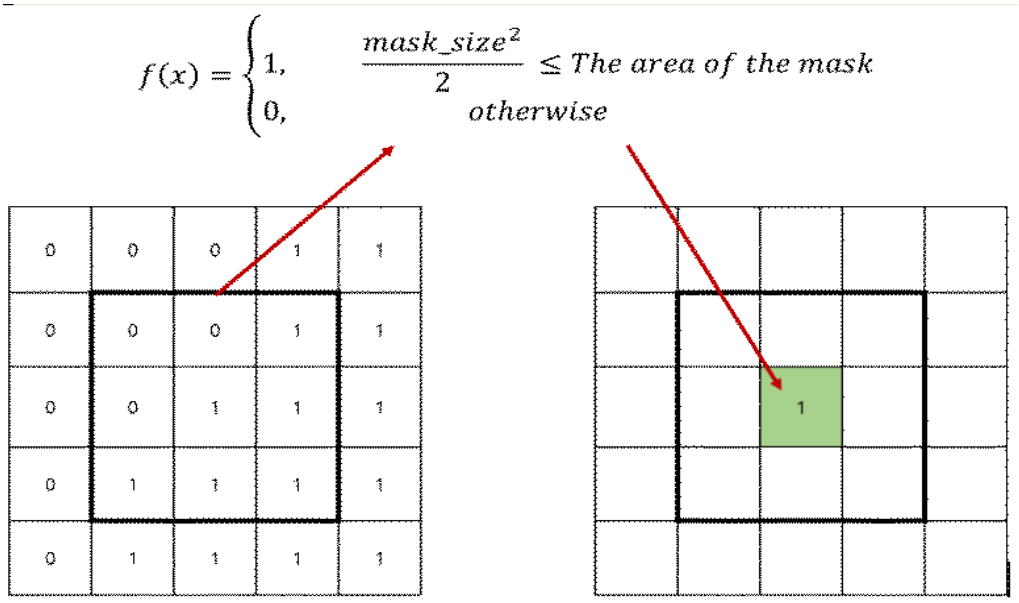

Fig. 5: A method of outputting a median value using an integral image

Comparator: The comparator module performs median filtering using the integral image calculated by the Integral image generator module. The original median filtering is shown in Fig. 5. The original method shows that the values in the mask area are aligned and the median value is outputted. The sorting algorithm has the feature in that the amount of computation exponentially increases as the array size increases. The proposed method is shown in Fig. 5. The proposed method uses the feature that a pixel of the binary image consists of only 0 and 1. Due to this feature, the number of 1's in the mask is equal to the sum of the pixel values in the mask. Therefore, when the number of 1's is larger than half of the mask area, the median value is determined as 1 and the median value is determined as 0 when the number is < half. This method can calculate median value quickly regardless of mask size.

RESULTS AND DISCUSSION

In this study, we evaluated the proposed algorithm by comparing the processing speed with the original method and the mask size. The proposed algorithm consists of C++ language and open CV library. Testing for processing

Table 1: The processing speed according to mask size according to VGA resolution for the proposed algorithm and the original method

Variables	Original method (msec)	Proposed method (msec)
Mask size		
7×7	10	1
9×9	11	1
11×11	13	1
13×13	15	1

Table 2: The processing speed according to mask size according to HD resolution for the proposed algorithm and the original method

Variables	Original method (msec)	Proposed method (msec)
Mask size		
7×7	29	4
9×9	33	4
11×11	39	4
13×13	45	4

speed can measure FPS at VGA, HD resolution. The test environment consists of the following. It ran on Intel® Core™ i7 at 3.60 GHz and Windows 10. Table 1 and 2 show the processing speed according to mask size according to VGA and HD resolution for the proposed algorithm and the original method. As shown in Table 1 and 2 as the mask size increases, the amount of computation increases in the original method but the proposed method can be processed fast regardless of the mask size.

CONCLUSION

In this study, we proposed a median filtering method for binary image using integral image instead of median filtering method which needs to arrange data array every pixel for binary image. Using the pre-computed integral image, the proposed algorithm shows that the median filtering operation is performed quickly regardless of the mask size. With this algorithm, it is possible to perform fast processing even if a large size mask is used.

REFERENCES

Ataman, E., V. Aatre and K. Wong, 1980. A fast method for real-time median filtering. IEEE Trans. Acoustics Speech Signal Process., 28: 415-421.

Glover, J.H., 1992. Fast median filter. US Patent No. 5,144,568. Honeywell International Inc., Washington, D.C. USA.

Huang, T., G. Yang and G. Tang, 1979. A fast two-dimensional median filtering algorithm. IEEE. Trans. Acoust. Speech Signal Process., 27: 13-18.

Lewis, J.P., 1995. Fast template matching. Fast Normaliz Ed Cross Correlation, 95: 120-123.

Perreault, S. and P. Hebert, 2007. Median filtering in constant time. IEEE. Trans. Image Process., 16: 2389-2394.

Yang, Q., N. Ahuja and K.H. Tan, 2015. Constant time median and bilateral filtering. Intl. J. Comput. Vision, 112: 307-318.

Ze, T. and L. Zhong, 1996. Fast median filtering. J. Northwest Inst. Text. Sci. Technol., 10: 381-384.