UNITECH 2019 GABROVO

INTERNATIONAL SCIENTIFIC CONFERENCE
15 – 16 November 2019, GABROVO

MODIFICATION OF ALGORITHM FOR GLOBAL MEDIAN THRESHOLDING

Atanaska Bosakova-Ardenska
dep. of Computer Systems and Technologies
University of Food Technologies – Plovdiv

Angel Danev
dep. of Computer Systems and Technologies
University of Food Technologies - Plovdiv

Abstract

The thresholding is an important part of image processing. The results of thresholding are binary images which are base for object recognition. The thresholding could be global or local when image processing is performed in a space domain. This paper presents one modification of an algorithm for global median thresholding named HisMedian. The modification is proposed to accelerate image processing by this algorithm. Both algorithms- HisMed and its modification are implemented on Java. It is presented a theoretical evaluation for time complexity of both examined algorithms. Test images with different sizes are used for experimental evaluation of examined algorithms. The results show that proposed modification is faster than original algorithm for global median thresholding (HisMed) when the count of zero values in a histogram of processed image is not very big.

Keywords: image processing, thresholding, HisMedian algorithm, histogram, median.

INTRODUCTION

Visual information is very important for the human who accept about 90% of information for environment through visual receptors and thus the image processing is an important part of data processing. Last years computer technologies are rapidly improved and there are developed a lot of applications for images processing [1, 2, 3, 4]. The algorithms for image processing could be grouped according their purpose into following categories:

- algorithms for images compression: there are known two basic types of images compression- with information loss and without information loss;
- algorithms for images enhancement: this group contains i.e. algorithms for filtering, brightness and contrast adjustment. Image processing with specific filter could be executed in a space domain or in a frequency domain;
- algorithms for images segmentation: these algorithms significantly reduce count of colors in processed image. When the result image contains only two colors then the processing is known as thresholding (binarization) and the image is called a binary image;
- algorithms for objects and patterns recognition: these algorithms use some features extracted from segmented images and their aim is to identify some objects as a member of specific group;
- algorithms for images transformation: this group includes algorithms for rotation, resizing, convolution, Fourier transformation and etc.

Thresholding algorithms could be grouped into two main groups – algorithms for global thresholding and algorithms for local thresholding [5]. The main aim of this paper is to present a modification of one algorithm for global thresholding.

THE ALGORITHM FOR GLOBAL MEDIAN THRESHOLDING

The algorithm for global median thresholding (HisMedian) falls in the group of thresholding algorithms which use real color value from the image as a threshold [6]. Figure 1 presents an image before and after processing with HisMedian algorithm.

(a) Original image *(b) Binary image*

Fig. 1. *Thresholding with HisMedian algorithm*

HisMedian algorithm uses histogram data to build sub-histogram and to find threshold value. Figure 2 presents the main steps of the algorithm as a flowchart.

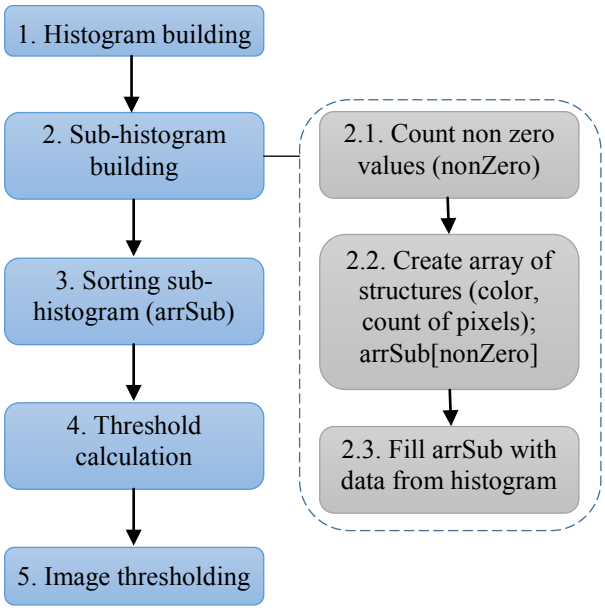

Fig. 2. *HisMedian algorithm*

Histogram building is performed through passing all pixels of the image and incrementing count of color for processed pixels.

The sub-histogram building contains several steps as follows:

Step 2.1. – Count non zero values in the histogram and store this count in variable nonZero;

Step 2.2. – Create array of structures (arrSub[]) with size equal to nonZero. The structure contains two fields- color (the value between 0 and 255) and count (number of pixels with specific color).

Step 2.3. – Filling the array of sub-histogram is performed through passing histogram of the image and copying information for color and count of pixels only for colors with non-zero count of pixels.

The array arrSub is sorted according to the values of field for count of pixels.

Threshold value is set to be equal to color of median element into sorted array of sub-histogram.

The binary image is produced by using the calculated threshold value.

Time complexity for threshold calculation by HisMedian algorithm could be calculated by formula 1.

$$T = \sum_{}^{m.n} t_{inc} + \sum_{}^{256} t_{cmp} + k.t_{inc} + \sum_{}^{k} t_{store} + t_{sort} \quad (1)$$

where m and n are sizes of the image in pixels, t_{inc} is time for performance of operation increment, t_{cmp} is time for performance of operation comparison, k is the number of non-zero values in the histogram (this value is between 0 and 256), t_{store} is time for performance of store operation for one element of the structure and t_{sort} is time for sub-histogram sorting.

THE MODIFIED ALGORITHM FOR GLOBAL MEDIAN THRESHOLDING

The main idea of the proposed modification is that a lot of real situation are source of grayscale images with many colors (above 200) and because of this the building of sub-histogram could be replaced with building array of structures (color, count of pixels) sized 256. This array will contains full information of the histogram, i.e. colors with zero count of pixels will be also presented. The array could be used for the threshold calculation after sorting. Figure 3 presents the main steps of the proposed algorithm as a flowchart.

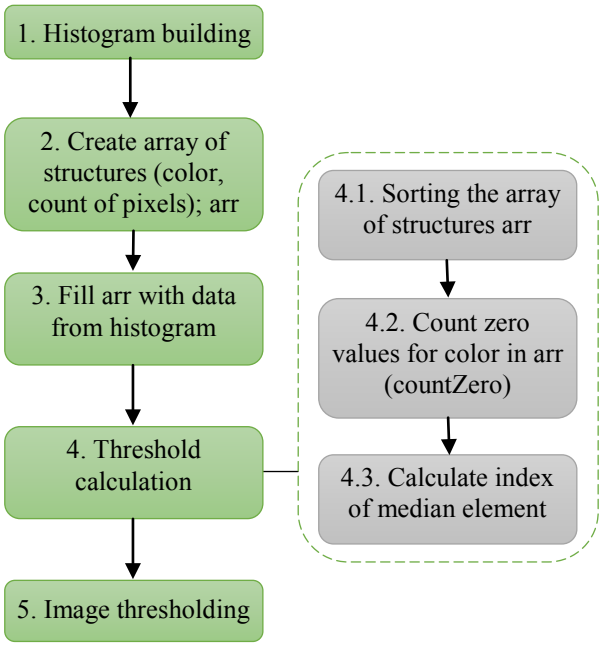

Fig. 3. *Modified HisMedian algorithm*

Histogram building is performed through passing all pixels of the image and incrementing count of color for processed pixels.

It is created array of structures (color, count of pixels) with size 256.

Filling the array is performed through passing histogram of the image and copying information for color and count of pixels.

Threshold calculation contains three steps:

Step 4.1. – Sorting the array of structures (color, count of pixels) arr according to the values of field for count of pixels.

Step 4.2. – Count the zero values for color (countZero) through passing sorted array arr.

Step 4.3. – Calculation of index of median element in a sorted array by formula:

$$idx = countZero + (256-countZero)/2 \quad (2)$$

The threshold value is equal to color of median element in sorted array of structures (arr[idx].color).

The last step of the algorithm is image thresholding, i.e. the image is binarized using calculated threshold.

Time complexity for threshold calculation by modified HisMedian algorithm could be calculated by formula 3.

$$T = \sum^{m.n} t_{inc} + \sum^{256} t_{store} + t_{sort} + \sum^{p} (t_{cmp} + t_{inc}) + t_{idx} \quad (3)$$

where m and n are sizes of the image in pixels, t_{store} is time for performance of store operation for one element of the structure, t_{sort} is time for sorting array of structures arr, t_{cmp} is time for performance of operation comparison, t_{inc} is time for performance of operation increment, p is number of elements with value zero for pixels count in arr, t_{idx} is time for calculation of index by formula 2.

The proposed modification of the algorithm HisMedian will be faster than HisMedian algorithm if next inequality is true:

$$\sum^{256} t_{cmp} + k.t_{inc} + \sum^{k} t_{store} > \sum^{256} t_{store} + \sum^{p} (t_{cmp} + t_{inc}) + t_{idx} \quad (4)$$

If value of k is near to 256, i.e. zero values in histogram are small number, then inequality (4) could be transformed to inequality (5).

$$\sum^{256} (t_{cmp} + t_{inc}) > \sum^{p} (t_{cmp} + t_{inc}) + t_{idx} \quad (5)$$

Time for index calculation (t_{idx}) is small because it includes only three arithmetic operations (one addition, one subtraction and one division) because of this it could be ignored. The value of p is small because it is equal to 256-k and also it could be ignored. Finally inequality (4) could be transformed to inequality (6) when value of k is near to 256 (this value corresponds to a lot of real images).

$$\sum^{256} (t_{cmp} + t_{inc}) > 0 \quad (6)$$

The proposed modification of HisMedian algorithm is faster than original HisMedian algorithm when the number of zero values in histogram is small, which is proved by inequality (6).

RESULTS AND DISCUSSION

The proposed modification of the HisMedian algorithm and the original HisMedian algorithm are implemented on Java. Figure 4 presents graphical user interface (GUI) of developed application.

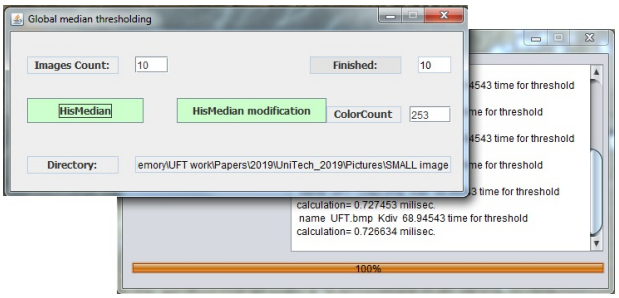

Fig. 4. *Application for thresholding with HisMedian algorithm and its modification*

The application has two windows- one for settings and second one for results. The settings window has two buttons- one for thresholding with HisMedian algorithm and second for thresholding with modified HisMedian algorithm. The results window presents information about name and location of processed images and time for thresholding calculation without time for histogram building. Developed application processes images in 24 BMP format. Every image is converted to grayscale using formula (7) which uses popular gamma correction for luminosity [7].

$$gray = 0.21*red + 0.71*green + 0.07*blue \quad (7)$$

Six images with different sizes and different count of non-zero values in histogram are used for experiments. Table 1 presents information about these images.

Table 1. Details for experimental images

	Size [pixels]	Count of non-zero values in histogram
IMG1	1417x970	253
IMG2	2048x1536	251
IMG3	3000x2250	253
IMG4	1420x939	86
IMG5	2048x1355	86
IMG6	3000x1984	86

Time evaluation is done using Java method nanoTime() and computer system with Intel Celeron E3300 Dual Core 2,5GHz processor and RAM-3GB. Figures 5 and 6 present fragment of source code where is performed time measuring. After the time measuring a conversion form nanoseconds to milliseconds is performed.

```
start=System.nanoTime();
int nonZeroColors=0;

for(int counter=0; counter<256; counter++)
  if(histogram[counter]!=0) nonZeroColors++;

HistogramMed arrSub[]=new
HistogramMed[nonZeroColors];
for(int c=0; c<nonZeroColors; c++)
  arrSub[c]=new HistogramMed();
int index=0;
for(int counter=0; counter<256; counter++)
{
  if(histogram[counter]!=0)
  {
    arrSub[index].color=counter;
    arrSub[index].count=histogram[counter];
    index++;
  }
}
Comparator<HistogramMed> comp =
Collections.reverseOrder();
Arrays.sort(arrSub,comp);
myThreshold=arrSub[nonZeroColors/2].color;
end=System.nanoTime();
diff=end-start;
```

Fig. 5. *Time measuring for algorithm HisMedian (source code- fragment)*

Class HistogramMed contains data variables for color and count of pixels and it is defined to implements java interfaces Comparator and Comparable. The methods compare() and compareTo() are implemented to performe comparison between two objects (instances of HistogramMed class) by their values for count of pixels. The array, of objects instanced of class HistogramMed, is named arrSub in figure 5 and it presents sub-histogram (fig. 2). The sub-histogram is sorted using method sort() which is member of java class Arrays.

```
start=System.nanoTime();
HistogramMed arr[]=new HistogramMed[256];
for(int counter=0; counter<256; counter++)
  arr[counter]=new HistogramMed();
for(int counter=0; counter<256; counter++)
{
  arr[counter].color=counter;
  arr[counter].count=histogram[counter];
}
Comparator<HistogramMed> comp =
Collections.reverseOrder();
Arrays.sort(arr,comp);
int idx=0;
for(int counter=0; counter<256; counter++)
  if(arr[counter].count==0)
    idx=counter;
  else
    break;
myThreshold=arr[idx+((256-idx)/2)].color;
end=System.nanoTime();
diff=end-start;
```

Fig. 6. *Time measuring for modified algorithm HisMedian (source code- fragment)*

Every image is processed 20 times with implemented algorithms and average time results are presented in graphics. Figures 7 and 8 present time for thresholding calculation (without time for histogram building) using HisMedian algorithm and its modification.

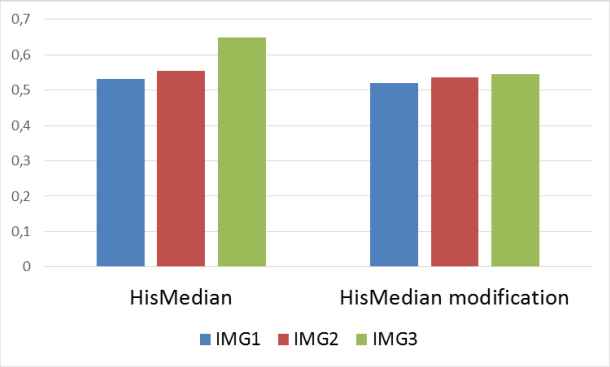

Fig. 7. *Time [ms] for processing images with many colors with algorithms HisMedian and its modification*

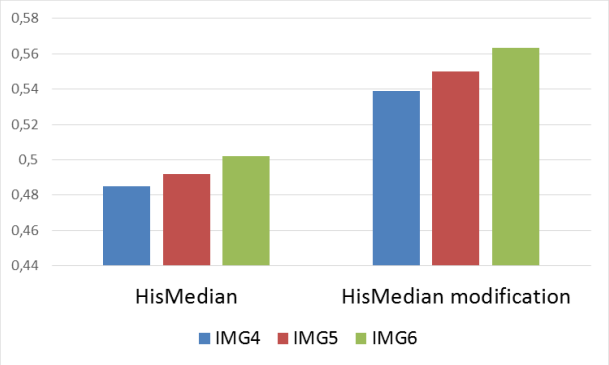

Fig. 8. *Time [ms] for processing images with small number of colors with algorithms HisMedian and its modification*

It is observed that modified algorithm HisMedian is faster than original algorithm by processing images IMG1, IMG2 and IMG3. These images contain small number of zero values in histogram (3 to 5). When the processed images have a lot of zero values in histogram (170) then the modified algorithm is slower than original HisMedian algorithm. It is observed also that time for processing increases when size of images increases too.

CONCLUSION

One modification of global median thresholding algorithm HisMedian is presented in this paper. The algorithm HisMedian and its modification are implemented on Java in application with GUI. There are presented theoretical and experimental evaluations of the HisMedian algorithm and the proposed modified HisMedian algorithm. The results show that theoretical analysis is correct and the proposed modification is faster than original HisMedian algorithm when the processed image has small number of zero values in the histogram.

REFERENCE

[1] S. Naseera, G.K. Rajini, B. Venkateswarlu, J. Priyadarsini, A Review on Image Processing Applications in Medical Field, Research Journal of Pharmacy and Technology, 2017, vol. 10, pp 3456-3460, DOI: 10.5958/0974-360X.2017.00644.8

[2] P. Mihova, G. Petrov, F. Andonov, Applications of open source frameworks for advanced medical image processing, Bulgarian Journal of Public Helth, 2015, vol VII, issue 1, ISSN: 1313-860X, pp 69-77

[3] Hema L. Chavan, Santosh A. Shinde, A Review on Application of Image Processing for Automatic Inspection, International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), 2015, vol. 4, Issue 11, ISSN: 2278 – 1323, pp 4073-4075

[4] I. Nikolova, G. Zapryanov, K. Alexiev, Detecting of Unique Image Features by Using Camera with Controllable Parameters, Proceedings of the Fourth International Bulgarian-Greek Conference Computer Science, 2018, Vol. 3, pp. 18-19

[5] Gonzalez R., R. Woods, Digital Image Processing, Pearson Education, 2009, ISBN 978-81-317-2695-2

[6] A. Bosakova-Ardenska, A. Danev, An Algorithm for Histogram Median Thresholding, CompSysTech 2018, ISBN: 978-1-4503-6425-6, pp 62-67, DOI: 10.1145/3274005.3274019

[7] Kanan C, Cottrell GW, Color-to-grayscale: does the method matter in image recognition?, PLoS ONE, 2012, vol. 7, issue 1, DOI: 10.1371/journal.pone.0029740.