# Thresholding Algorithm for Local and Parallel Processing

Satoshi Iwakata
Keio University
3-14-1 Hiyoshi, Kohoku-ku,
Yokohama 223-8522, Japan
Email: iwakata@soft.ics.keio.ac.jp

Yoshiaki Ajioka
Ecchandes Inc.
12-7 Chuohommachi, Gamagori,
Aichi 443-0057, Japan
Email: ajioka@cyberdoc.co.jp

Masafumi Hagiwara
Keio University
3-14-1 Hiyoshi, Kohoku-ku,
Yokohama 223-8522, Japan
Email: hagiwara@soft.ics.keio.ac.jp

***Abstract*— In this paper, we propose a new thresholding algorithm. This algorithm works under a severe constraint: each pixel in a processed image must be derived from only information of the neighboring pixels. This constraint is very important for a low cost device such as a mobile camera, because it makes possible to process each pixel in parallel. The proposed algorithm deals with gray-scale images, and determines the threshold based on edge information. The proposed algorithm is represented by local and parallel image processing and has been tested using 104 scenery images. The result shows that the proposed algorithm can binarize images.**

## I. Introduction

In recent years, image processing for supporting people with visual impairment is popular[1]-[4]. On the other hand, vision chips attract much attention[5][6]. Especially, a vision chip comprising massive processing elements (PEs) will play an important role for a system LSI in a 3G cellular phone with a mega-pixel mobile camera. We, therefore, have developed an algorithm of image processing for such a vision chip.

Our target vision chip comprises many PEs which carry out processing individually, in parallel, but they can work synchronously for some specific periods of a clock, according to a timeout mechanism[7]-[9]. Although they can communicate with only their four neighbors asynchronously via some physical connections. They can logically communicate with some other neighboring PEs by transmitting their data in order. Therefore, they can successively carry out local processing, synchronizing with each other. We call such processing local and parallel image processing.

Fig.1 shows rough sketch of our target vision chip on which the proposed algorithm works.

Each PE can deal with up to 900 pixels. Each PE is capable of 16 bit shift operation and multiplication.

Features of our target vision chip are parallel processing and low cost. It can downsize systems because it needs little electricity to work. It also can process an image rapidly due to parallel processing.

However, special algorithms composed of local and parallel processing are necessary to make use of this kind of vision chip. Therefore, in this paper, we propose a new thresholding algorithm using local and paralell processing suitable for our target vision chip

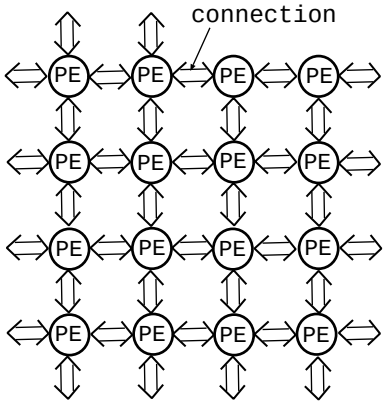


Fig. 1. A vision chip for local and parallel processing

Thresholding is an important process which is often used in automatic extracting[10][11]. Thresholding has much effect on post-processing. As a result, bad thresholding degrades the total performance[12].

The proposed algorithm uses only edge information and is composed of local and parallel processing suitable for an above vision chip.

In this paper, chapter 2 explains general thresholding. Chapter 3 explains the proposed algorithm, chapter 4 shows the effectiveness of the proposed algorithm by presenting the results of the computer simulations. Chapter 5 concludes this paper.

## II. Thresholding

The purpose of the thresholding is separating the target object from the background. Even if the same image, proper threshold is different depending on the target object.

Research of thresholding has a long history. The method focused on concentration distribution[13] and the method focused on edges[10] have been proposed. However, these methods need global information. These methods are not suitable for our target vision chip.

There are many cases which one threshold is calculated for one image. It is difficult to get good results for various images using these methods.

## III. Proposed Algorithm

The proposed algorithm uses edge information. It is composed of the following steps: smoothing, edge detection, threshold detection, and thresholding. Of course, they are represented by the local and parallel image processing. Fig.2 shows the flow of the proposed algorithm.

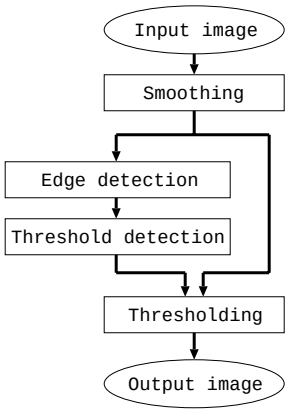


Fig. 2. Procedure of proposed algorithm

### A. Definition of neighbors

Local and parallel image processing is realized using information of neighboring pixels. In the image $X$, the pixel value at $(i, j)$ is expressed as $x(i, j)$. $P^q_{i,j}$, the neighboring set of $(i, j)$, is expressed as follows,

$$P^q_{i,j} = \{(l,m) \,\Big|\, i-r \le l \le i+r, \quad j-r \le m \le j+r, (l,m) \ne (i,j)\}. \tag{1}$$

In equation (1), $r$ is neighbor length: the number of neighboring pixels $q$ is expressed as follows,

$$q = (2r+1)^2 - 1. \tag{2}$$

In other words, $q = 8, 24, 48 \cdots$, when $r = 1, 2, 3, \cdots$. When neighboring pixels exceed the range of the image, such pixels are substituted for the bordoring pixels. This indicates that all pixels have the same number of neighboring pixels.

### B. Smoothing

Pixel values of inputted image $X$ are smoothed as follows,

$$x^{\text{new}}(i,j) = \frac{1}{8} \sum_{(l,m) \in \hat{P}^8_{i,j}} x(l,m) \tag{3}$$

and

$$\hat{P}^q_{i,j} = \begin{cases} (l,m) & \text{if} \quad |x(i,j) - x(l,m)| \le D \\ (i,j) & \text{else.} \end{cases} \tag{4}$$

When difference in pixel value between targeted pixel $(i, j)$ and the neighboring pixel $(l, m)$ is more than $D$, $x(i, j)$ is substituted for $x(l, m)$. This processing makes an image smooth preserving edges.

The target objects often have high contrast with their background in images. This smoothing can hold down unevenness of color depth in one region highlighting necessary edges. The proposed algorithm uses Laplacian filter to detect edges. This kind of spatial filter is affected by the noise if it is used without modification. If this smoothing is not used, many unnecessary edges are detected at the next step. For example, changes in color depth which human beings don't sense are often detected by computers. It divides one region into plural regions.

This smoothing can combine the regions which have gradual changes into one region. The smoothing is operated $N$ times.

### C. Edge detection

After the smoothing, edges are detected. Laplacian filter is used for edge detection. Fig.3 shows Laplacian operator.

| | | |
|---|---|---|
| -1 | -1 | -1 |
| -1 | 8 | -1 |
| -1 | -1 | -1 |

Fig. 3. Laplacian filter

Using the Laplacian filter, an edge image $E$ is created from a grayscale image $X$. Each pixel of the edge image $E$ is presented as follows,

$$e(i,j) = \begin{cases} 1 & \text{if} \quad \left[8 \cdot x(i,j) - \sum_{(l,m) \in P^8_{i,j}} x(l,m)\right] > x(i,j) \\ 0 & \text{else.} \end{cases} \tag{5}$$

Using Laplacian filter, the part which has great changes in pixel value is detected in equation (5). Then, the result is compared with the target pixel. Pixels having high pixel value, namely bright pixels are detected as edges.

### D. Threshold detection

After edge detection, candidate values of the threshold $c(i, j)$ are calculated at edge pixels. Threshold candidate $C$ is calculated as follows,

$$c(i,j) = \begin{cases} \dfrac{x(i,j) + \min\limits_{(l,m) \in P^8_{i,j}} x(l,m)}{2} & \text{if} \quad e(i,j) = 1 \\ 0 & \text{else.} \end{cases} \tag{6}$$

In equation (6), min[·] means minimum in [ ].

At the neighbor of edge pixels, the part having the biggest gradient is detected. Intermediate value between the bright pixel and the dark pixel is threshold candidate. Therefore, equation (6) calculates intermediate values between edge pixel and the minimum value of neighbor of the edge pixels.

### E. Thresholding

Based on threshold candidates, threshold $t(i,j)$ is calculated at each pixel. This is represented as follows,

$$t(i,j) = \begin{cases} \mathrm{Ave}[F^q_{i,j}] & \text{if} \quad \sum_{(l,m)\in P^q_{i,j}} c(l,m) > 0 \\ 0 & \text{else} \end{cases} \tag{7}$$

and

$$F_{i,j} = \{c(i,j) | c(i,j) > 0\}. \tag{8}$$

In equation (7), Ave$[\cdot]$ means average. In equation (8), $F_{i,j}$ is the set of c(i,j) which is more than 0.

After the threshold set $T$ is calclated, thresholding is done using each $t(i,j)$. This operateion is represented as follows,

$$y(i,j) = \begin{cases} 1 & \text{if} \quad x(i,j) < t(i,j) \\ 0 & \text{else.} \end{cases} \tag{9}$$

Here, $y(i,j)$ is a pixel value of binary image $Y$. In equation (9), adjusting neighbor length, the object which has specific size can be extracted.

## IV. Computer Simulation

In order to show the usefulness of the proposed algorithm, we carried out computer simulations.

### A. Effect of smoothing

First we verified the effect of smoothing. We tested the proposed algorithm changing the number of smoothing $N$. Fig.4 shows edge images, Fig.5 shows results of thresholding.

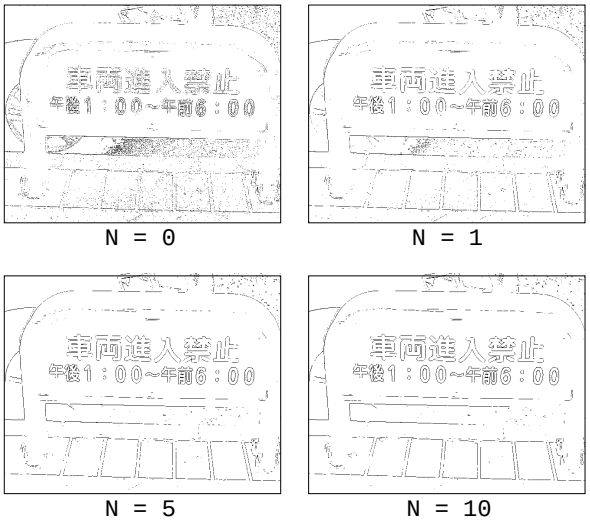


Fig. 4. Differences in edge

In Fig.4 and Fig.5, $N$ means the number of smoothing. From these figures, we can see that unnecessary edges are detected when the number of smoothing is too small. This constricts proper thresholding. Since the proposed algorithm uses edge information, smoothing is necessary.

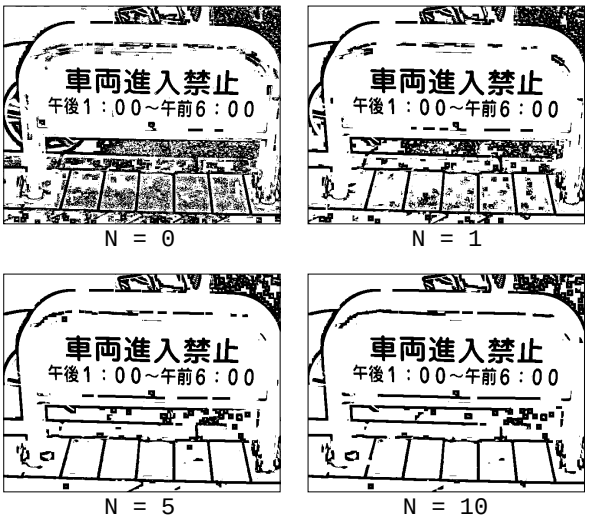


Fig. 5. Differences in thresholding

### B. Application to character extraction

Research of character extraction from scenery images is popular[14]-[16]. Characters and figures have much information in the shapes rather than light and shade. Therefore, thresholding is often used as preprocessing.

We used 104 senery images taken by usual digital camera. We compared the proposed algorithm and discriminant analisys method.

Discriminant analysis is the method to calculate threshold statistically. It divides pixels into two classes based on density histogram. It is known as a good method to binarize[17].

Input images are grayscale and $640 \times 480$ pixels. 20 images were used for setting parameter, and the other 84 images were used for the evaluation.

Here, we define the extraction rate as follows,

[Extraction rate]

$$= \frac{\text{number of extracted character}}{\text{number of characters in image}} \times 100[\%]. \tag{10}$$

Fig.6 and Fig.7 show some examples of input and the output images. In image1 - image8 of Fig.6 and Fig.7, top is original image, middle is a result of discriminant analysis, and the bottom is a result of the proposed algorithm.

From image4 in Fig.6 and image8 in Fig.7, there are some cases that discriminant analysis can't binarize images properly. On the other hand, the proposed algorithm can binarize low contrast images properly.

Table I shows conditions of computer simulations. The values of parameter were determined based on preliminary simulations. Table II summarizes the result of computer simulations.

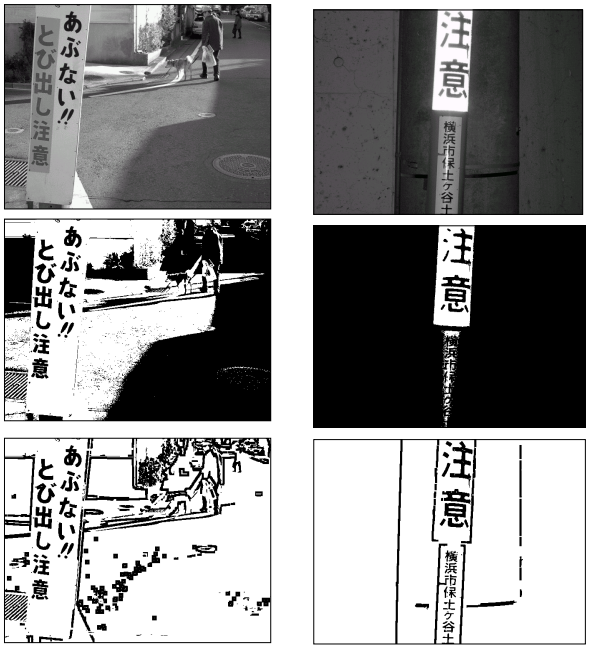


image 1 image 2


image 3 image 4

Top : input
Middle : discriminant analisys
Bottom : proposed

Fig. 6. Examples of computer simulations

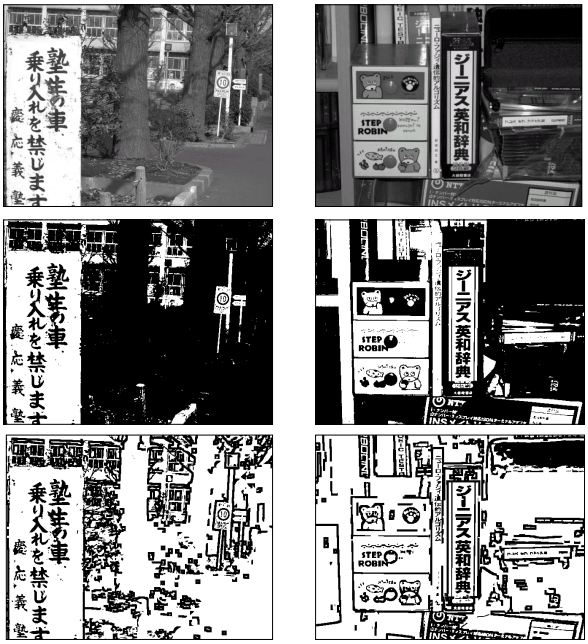


image 5 image 6

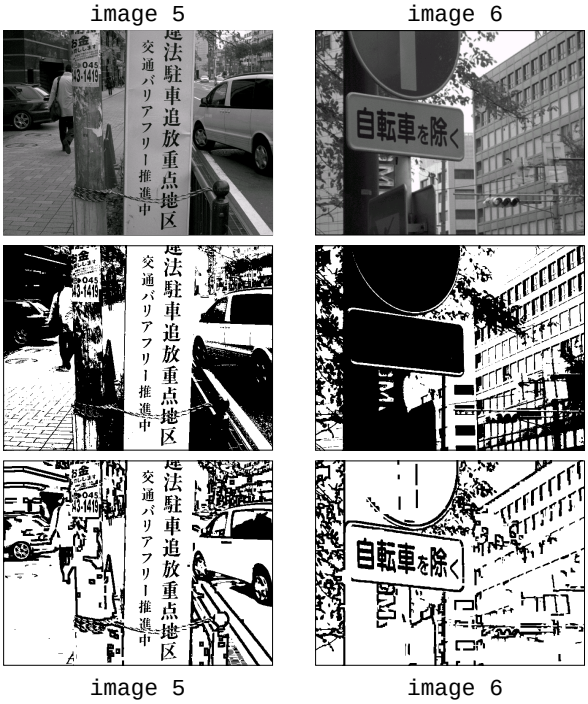


image 5 image 6

Top : input
Middle : discriminant analisys
Bottom : proposed

Fig. 7. Examples of computer simulations

TABLE I

PARAMETERS OF PROPOSED ALGORITHM

| parameter | value |
|---|---|
| N at smoothing (%) | 5 |
| D at smoothing (%) | 25 |
| r at thresholding (%) | 5 |

TABLE II

A RESULT OF COMPUTER SIMULATION

| | proposed | discriminant analysis |
|---|---|---|
| extraction rate (%) | 89.3 | 85.1 |

Next, we applied the proposed algorithm to images changing parameters. The results of this simulation is summarized in Fig. 8.

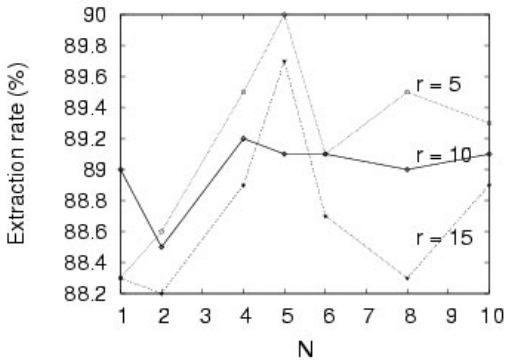


Fig. 8. Extraction rate of proposed algorithm

From this figure, increase of smoothing times improves extraction rate. However, when smoothing is repeated too many times, extraction rate degrades.

The extraction rate of the proposed algorithm was up to 90.0%, and 87.5% at minimum. The extraction rate of discriminant analysis was 85.1%. The proposed algorithm worked well despite the severe constraint: each pixel of a result image must be derived from only information of their neighbor pixels.

## V. CONCLUSION

In this paper, we proposed a new thresholding algorithm for local and parallel processing. This algorithm works under a severe constraint: each pixel in a result image must be derived from only information of their neighboring pixels. This constraint is very important for a low cost device like a mobile camera.

In simulations, the proposed algorithm has been tested for 104 scenery images. The result shows that the algorithm can binarize images with proper threshold.

## REFERENCES


[1] H.Hiraiwa, Y.Takeuchi, T.Matsumoto, H.Kudo, Y.Liu,and N.Ohnishi: “A System Assisting the Visually Impaired People to Understand Character Information in the Environment”, IEICE Technical Report, PRMU2001-269, pp.195-202 (2002-3)(in Japanese)

[2] T.Nagai, T.Kagehiro, M.Kaneko, and A.Kurematsu: “Text and Signboard Detection in Scene Images”, IEICE Technical Report, CS2000-145, pp.103-108 (2001-3)(in Japanese)

[3] K.Kiyota, T.Sakurai, and S.Yamamoto: “On-Line Character Recognition for the Visually Disabled Person Based on the Relative Position of Stroke Representative Points”, IEICE Trans, Vol.J80-D-II, No.3, pp.715-723(1997-3)(in Japanese)

[4] F.Hang, T.Nagai, M.Kaneko, and A.Kurematsu: “Extraction of Characters on Signboards from Various Scene Images”, IEICE Technical Report, IE2000-157, pp.13-18 (2001-1)(in Japanese)

[5] Y.Hirai, T.Ochiai, and M.Yasunaga: “A Neural Network System Composed of 1000 Neurons and One Million 7-bit Synapses”, T.IEICE Japan, Vol.J84-D-II, No.6, pp.1185-1193(2001-6)(in Japanese)

[6] M.Ishikawa, T.Komuro, and S.Kagami: “New Deployment of Digital Vision Chip”, IEICE Technical Report, ICD2002-07, pp.23-28, (2002-07)(in Japanese)

[7] Y.Ajioka: “Visual Device”, PCT, W000/16259

[8] Y.Ajioka, Y.Shimada, and H.Amano: “Size/Position detection algorithm on The Visual Device”, IEICE Technical Report, VLD2001-135, CPSY2001-94, pp.23-30(2002-1)(in Japanese)

[9] S.Iwakata, Y.Ajioka, and M.Hagiwara: “Character Extraction Algorithm by Local and Parallel Processing”, IEEJ Trans, EIS, Vol.124-C, No.4, pp.959-965(2004-4)(in Japanese)

[10] F.Saitoh: “A Thresholding Technique for Character Images Based on Gray-levels of Plane-edges”, IEEJ Trans. EIS, Vol.123, No.10, pp.1760-1767(2003-10)(in Japanese)

[11] Shung-Shing Lee, Shi-Jimm Horng, and Horng-Ren Tsai: ”Entropy Thresholding and Its Parallel Algorithm on the Reconfigurable Array of Processors with Wider Bus Networks”, IEEE Trans. Image Processing, Vol.8, No.9, pp.1229-1242(1999-9)

[12] Sumei Guo and Shinji Ozawa: “A Gray-Level Image Thresholding Method”, IEICE Trans, Vol.J80-D-II, No.1, pp.183-189(1997-1)(in Japanese)

[13] Hideki Goto, Masatsugu Hirayama, and Hirotomo Aso: “Character Pattern Extraction from Grayscale Document Images Using Local Multilevel Thresholding”, IEICE Trans, Vol.J82-D-II, No.11, pp.2188-2192(1999-11)(in Japanese)

[14] Hau Yang, Shinji Ozawa: “Extraction of Text Region From Color Image Sequence of A Book Cover Containing Gilt Characters”, T.IEE Japan, Vol.122-C, No.4, pp.630-637(2002)

[15] K.Matsuo, K.Ueda, and M.Umeda: “Extraction of Character String from Scenery Image by Binarizing Local Target Area”, T.IEE Japan, Vol.122-C, No.2, pp.232-241(2002-2)(in Japanese)

[16] Y.Liu, T.Yamamura, N.Ohnishi,and N.Sugie: “Extraction of Character String Regions from a Scene Image”, IEICE Trans, Vol.J81-D-II, No.4, pp.641-650(1998-3)(in Japanese)

[17] N.Otsu: “A Threshold Selection Method from Gray-Scale Histograms”, IEEE Trans. Systems, Man, and Cybernetics, Vol.SMC-9, No.1, pp.62-66(1979-1)