2003 International Symposium on Advanced Intelligent Systems
September 25-28, 2003, Jeju, Korea

Character Extraction Algorithm from Scenery Images by Parallel and Local 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: iwakata@soft.ics.keio.ac.jp

***Abstract*— In this paper, we propose an algorithm extracting character regions from scenery images. This algorithm works under a severe constraint: each pixel of a result image must be derived from only information of their neighbor pixels. This constraint is very important for a low cost device like a mobile camera. The proposed algorithm is represented by the local and parallel image processing. It has been tested for 100 scenery images. A result shows that the proposed algorithm can extract character regions at a rate of more than 90%. The result was obtained without learning any template images. the algorithm is very useful.**

I. Introduction

Since semiconductor technology has progressed rapidly, we can integrate more than 100 million transistors on a chip nowadays. As a result, a system LSI has attracted much attention. 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. Although they can communicate with only their four neighbors asynchronously via some physical connections, they can logically communicate with some other neighbors 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.

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

In this paper, we show the algorithm which can extract character regions from scenery images. We explain the proposed algorithm in chapter 2, and show the effectiveness of the proposed algorithm by presenting the results of the computer simulations in chapter 3.

II. Proposed Algorithm

The proposed algorithm is composed of the following steps: smoothing, classification of color, elimination of large region, and extraction of character region. Of course, they are represented by the local and parallel image processing. Figure 2 shows the flow of the proposed algorithm.

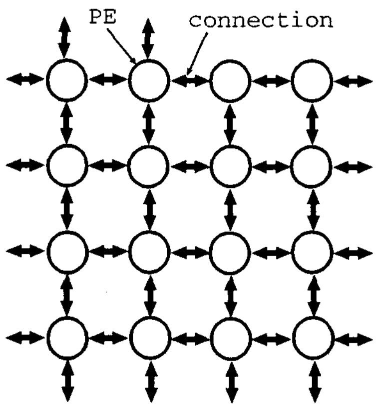


Fig. 1. A vision chip for local and parallel processing

A. Definition of neighbor

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

$$P^q_{i,j,k} = \{(l,m,k) \mid i-r \le l \le i+r, \; j-r \le m \le j+r, (l,m) \ne (i,j)\}. \quad (1)$$

In equation (1), r is neighbor length: the number of neighboring pixels q is expressed as $(2r+1)^2-1$. When neighboring pixels exceed the range of the image, such pixels are substituted for the bordoring pixels.

Neighbor size q is expressed as follows.

$$q = (2r+1)^2 - 1 \quad (2)$$

B. Smoothing

RGB values of inputted image x are smoothed as follows,

$$x^{\mathrm{new}}(i,j,k) = \frac{1}{q} \sum_{(l,m,k)\in P^q_{i,j,k}} x(l,m,k), \quad (3)$$

and

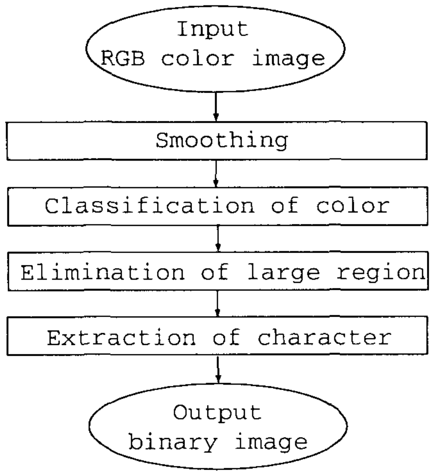


Fig. 2. Procedure of the proposed algorithm

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

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

C. Classification of color

We create multicolor images from inputted images based on relationship of complementary colors. 3 bands of inputted images $k = \{r,g,b\}$ are transformed into multicolor images $k = \{W,B,R,G\}$. Here, W means white, B means black, R means red, and G means green. At (i,j,k) in image x, extraction of multicolors is done by the following equations using q-neighbors.

$$C^q_{i,j,W} = \max_k \{q \cdot x(i,j,k) - \sum_{x(l,m,k) \in P^q_{i,j,k}} x(l,m,k)\} \tag{5}$$

$$C^q_{i,j,B} = -\max_k \{q \cdot x(i,j,k) - \sum_{x(l,m,k) \in P^q_{i,j,k}} x(l,m,k)\} \tag{6}$$

$$C^q_{i,j,R} = q\{x(i,j,r) - x(i,j,g)\} \tag{7}$$

$$C^q_{i,j,G} = -q\{x(i,j,r) - x(i,j,g)\} \tag{8}$$

Equation (5), (6),(7), and (8) are used to detect white, black, red, and green component, respectively.

The pixel having the largest value are selected for each color. Then binary images transformed to 4 bands are obtained as follows.

$$y(i,j,W) = \begin{cases} 1 & \text{if } C^q_{i,j,W} = \max_k\{C^q_{i,j,k}\}, \\ & C^q_{i,j,W} > f_1 \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

$$y(i,j,B) = \begin{cases} 1 & \text{if } C^q_{i,j,B} = \max_k\{C^q_{i,j,k}\}, \\ & C^q_{i,j,B} > f_2, \; C^q_{i,j,B} \neq C^q_{i,j,W} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

$$y(i,j,R) = \begin{cases} 1 & \text{if } C^q_{i,j,R} = \max_k\{C^q_{i,j,k}\}, \\ & C^q_{i,j,R} > f_3, C^q_{i,j,R} \neq C^q_{i,j,W}, \\ & C^q_{i,j,R} \neq C^q_{i,j,G} \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

$$y(i,j,G) = \begin{cases} 1 & \text{if } C^q_{i,j,G} = \max_k\{C^q_{i,j,k}\}, \\ & C^q_{i,j,G} > f_4, \; C^q_{i,j,G} \neq C^q_{i,j,W}, \\ & C^q_{i,j,G} \neq C^q_{i,j,B}, \\ & C^q_{i,j,G} \neq C^q_{i,j,R} \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

For example, pixels having white-like color are selected by equation (9).

D. Elimination of large region

At the stage of elimination of large region, too large regions are removed in the binary images y obtained from the previous section. First, the sum of q neighboring pixel values are calculated as follows,

$$R^q_{i,j,k}(x) = \sum_{y(l,m,k) \in P^q_{i,j,k}} y(l,m,k). \tag{13}$$

Then, too large regions are removed as follows,

$$U_{i,j,k} = \begin{cases} 1 & \text{if } x(i,j,k) = 1 \\ & R^q_{i,j,k}(x) \leq th \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

E. Extraction of character region

At the stage of extraction of character region, tiny texture is removed. In order to eliminate such texutures, it is necessary to move pixel values toward the center of the region and to detect size of the region.

1) Moving toward the center of the region: In order to move pixel values toward the center of the region, position/size detection algorithm[2] is used. This algorithm is to move pixel values toward the center of the region which the target pixel belongs to. As a result, pixels which locate center of the regions are detected and the size of each region is obtained. The procedure is expressed as follows.

- Step1 Detect the derection toward the center of the region locally using the information of 8-neighboring pixels.
- Step2 Move a pixel value to one of 8-neighboring pixels toward the center of the region and add the pixel value.
- Step3 Finish if all the pixels get to the center of each region, else return to Step 1.

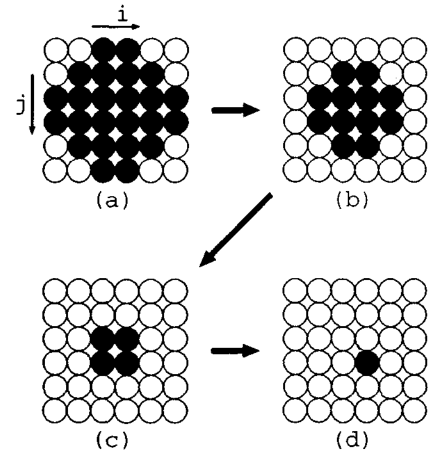


Fig. 3. An example of pixel move

The move-vector $v(i,j)$ which means amount of move toward the center at (i,j) of binary image y is expressed as follows,

$$v(i,j) = (r,s) \tag{15}$$

$$r = \sum_{y(l,m)\in P_{i,j}} (l-i)\cdot y(l,m) \tag{16}$$

$$s = \sum_{y(l,m)\in P_{i,j}} (m-j)\cdot y(l,m). \tag{17}$$

Then, pixels are moved toward $\Theta(v(i,j))$ and added their values after the move-vector is normarized in order to move to one of 8-neighboring pixels as follows,

$$\Theta(v(i,j)) = \begin{cases} (1,0) & \text{if } r>0,\ 2|s|<|r| \\ (1,1) & \text{if } r>0,\ s>0\ \ |r|/2 \le |s| \le 2|r| \\ (1,-1) & \text{if } r>0,\ s<0\ \ |r|/2 \le |s| \le 2|r| \\ (-1,0) & \text{if } r<0,\ 2|s|<|r| \\ (-1,1) & \text{if } r<0,\ s>0,\ |r|/2 \le |s| \le 2|r| \\ (-1,-1) & \text{if } r<0,\ s<0,\ |r|/2 \le |s| \le 2|r| \\ (0,1) & \text{if } s>0,\ 2|r|<|s| \\ (0,-1) & \text{if } s<0,\ 2|r| \le |s| \\ (0,0) & \text{otherwise} \end{cases} \tag{18}$$

This move is repeated until all the pixels get to the center of each region. Then, the size of region is detected at the pixel which is the center of region. This value at (i,j) is defined as $w(i,j)$. Figure 3 shows an example of pixel move.

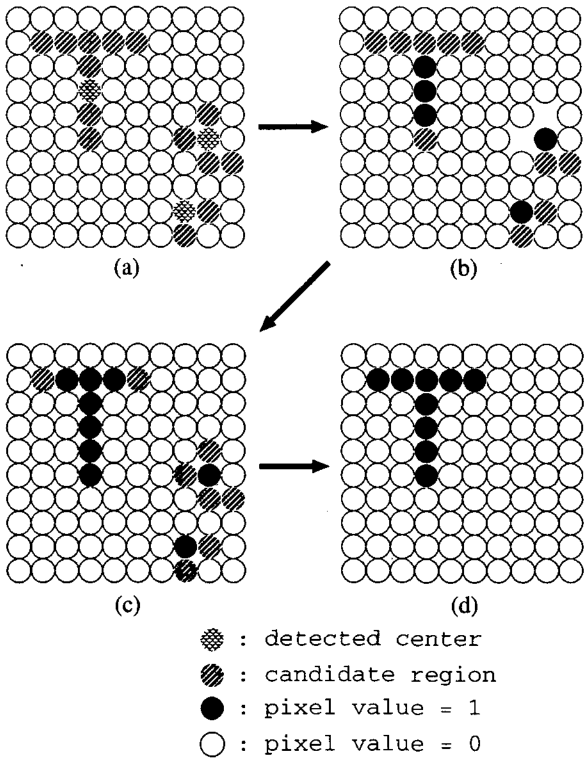


Fig. 4. Procedure of judging size

2) Judgement of size: We set thresholds Th_{big} and Th_{small} for judgement of region size. At the centers of regions, region size $w(i,j)$ is compared with these thresholds. If $Th_{\text{small}} \le w(i,j) \le Th_{\text{big}}$ and $x(i,j)=1$, value of the pixel is 1, else 0. These pixels send signal to 8-neighbors. At the pixels which have received this signal, if $x(i,j)=1$, the pixel value is 1. Else the pixel value is 0. By repeating this operation, the signal spread, and the region which has irrelevant size for characters are removed. Figure 4 shows the procedure of judgement of size.

III. Computer Simulations

In order to show the usefulness of the proposed algorithm, we carried out simulations using scenery images.

A. Conditions of simulation

We used 100 scenery images which are 640 × 480 pixels taken by a normal digital camera, 20 images were for setting parameter, and the other 80 images were for evaluation. We defined the detection rate as follows,

[Detection rate]

$$= \frac{\text{number of extracted character}}{\text{number of characters in image}} \times 100[\%]. \tag{19}$$

TABLE I

PARAMETER OF COMPUTER SIMULATION

parameter name	value
diff	30
times of smoothing	20
r at smoothing	1
r at classification of color	5
r at elimination of large region	10
f_1	4000
f_2	1000
f_3	4000
Th	380
Th_{small}	15
Th_{big}	40

TABLE II

A RESULT OF COMPUTER SIMULATIONS

Detection rate(%)	92.1
False detection rate(%)	3.5

In equation (19), number of characters in image means the number of characters expected to extract. Characters which could not be read are not included.

It is difficult to evaluate false detection rate. In this paper, we define the false detection rate as follows,

[False detection rate]

$$= \frac{\text{extracted non-character region}}{\text{number of pixels in images}} \times 100[\%]. \qquad (20)$$

Table I shows conditions of computer simulations. The values of parameter were determined based on preliminary simulations. In the simulation, we used 3 bands; $H_{i,j,1}$, $H_{i,j,2}$, $H_{i,j,3}$ at the stage of classification of color.

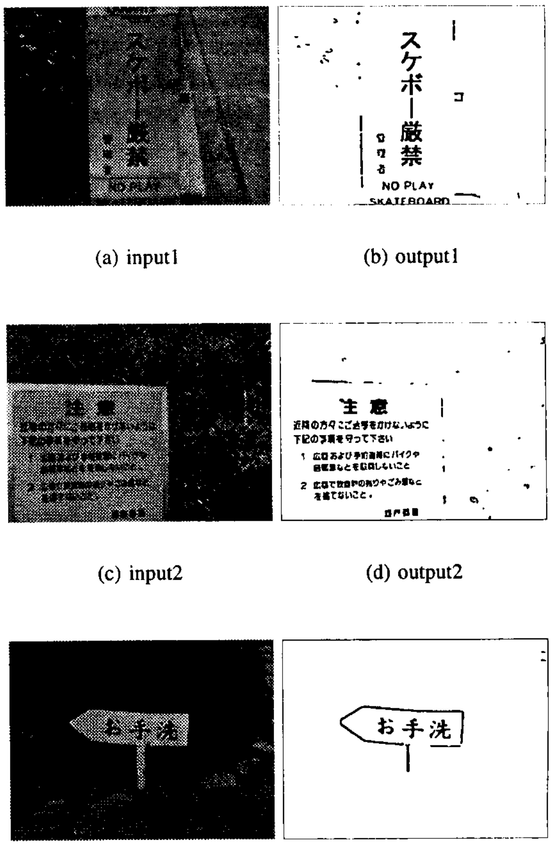
B. Result of computer simulation

Table II summarizes the result of computer simulations. Figure 5 shows some examples of input and the output images.

IV. CONCLUSION

In this paper, we proposed an algorithm which extract character regions from scenery images. 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 100 scenery images. The result shows that the algorithm can extract some character domains in a rate of more than 90%, and false detection is in a rate of less than 5%.



(a) input1 (b) output1

(c) input2 (d) output2

(e) input3 (f) output3

Fig. 5. Examples of input and the output

REFERENCES

[1] Yoshiaki Ajioka: "VISUAL DEVICE", PCT, W000/16259

[2] Yoshiaki Ajioka, Masahiro Shimada, Hideharu Amano: "Preliminary evaluation of The Vision System", Technical report, IEICE, Vol.101, No.696, 2002.