Pictogram Extraction Algorithm from Scenery Images by Parallel and Local Processing

Daisuke Fujita* Satoshi Iwakata* Yoshiaki Ajioka** Masafumi Hagiwara*

*Keio University, 3-14-1 Hiyoshi, Kohoku-ku, Yokohama 223-8522 Japan
**Ecchandes Inc., 12-7 Chuohommachi, Gamagori, Aichi 443-0057 Japan
e–mail:*{fujita,iwakata,hagiwara}@soft.ics.keio.ac.jp, **ajioka@cyberdoc.co.jp

abstract—— In this paper, a pictogram extraction algorithm is proposed. The proposed algorithm aims at supporting visually impaired people and works under a severe constraints: each pixel of a result image must be derived from information of its neighboring pixels. This constraint is very important for a low cost device like a mobile camera. The proposed algorithm is composed of pictogram candidate extraction phase, square region extraction phase, and pictogram extraction phase. In pictogram candidate extraction phase, centers of each region are used. In square region extraction phase, pixel accumulation method, a new algorithm is used for detecting square regions. This method uses edges of images. In the last phase, candidates and square regions are used for pictogram extraction. In computer simulations, the proposed algorithm has been tested for 100 images. The result shows that the proposed algorithm could extract pictogram regions at a rate of 80.7%.

Keyword: graphic, vision chip, artificial sight, extraction, pictogram,visually-impaired person

1 INTRODUCTION

Human beings get much information by our vision[1]. An information board in a station or an underground center is indispensable to lead us in the exact direction. Characters can tell contents to us exactly. However, there are various languages. When we cannot understand the meaning of language, we cannot acquire information from characters. Therefore pictograms are necessary. A pictogram is a "pictorial symbol[2]". It is easy to understand for many people beyond national borders [3]. For example, they are a road sign, a toilet mark, etc. To raise visibility, they are surrounded by the square frame and the area is filled in. In this paper, we propose a pictogram extraction algorithm for local and parallel processing. Section II explains our target hardware. Section III explains pictogram extraction algorithm. In Section IV, the simulation results are shown. Section V concludes the paper.

2 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 clock, according to a timeout mechanism[4][5]. Although they can communicate with only their four neighbors asynchronously via some physical connections, they can logically communicate with some other neighbors by transmitting data in order. Therefore, they can successively carry out local processing, synchronizing with each other. We call such processing parellel and local image processing[6].

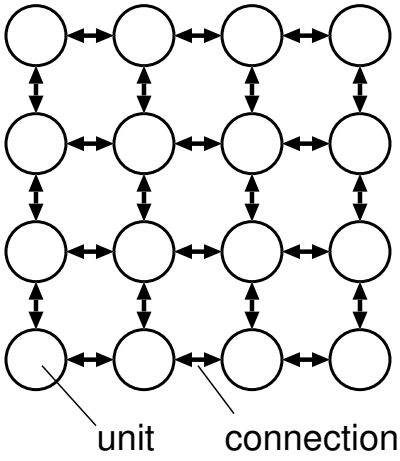

Figure 1: Hardware unit

3 PROPOSED ALGORITHM

3.1 Overview

The proposed algorithm is composed of three phases. They are pictogram candidate extraction phase, suqare region extraction phase, and pictogram extraction phase. In the first phase, it extracts the pictogram candidate from images. In the secoond phase,

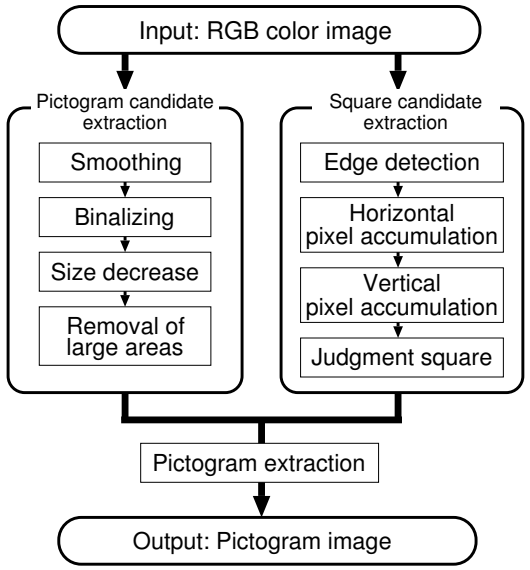

Figure 2: Algorithm

it extract the area surrounded with the square using the pixel accumulating method. In the last phase, it takes out a pictogram correctly by adding the result extracted from the phase 1 and 2. Figure 2 shows the flow of the proposed algorithm.

3.2 Nearby definition

In the proposed algorithm, only neighboring pixels for processing each pixel are used. We use information of q-neighboring pixels for each pixel. The neighbor set $P_{ij}(q)$ is expressed as follows,

$$P_{ij}(q) = \begin{cases} \text{if } q = 4 \\ \quad p(i+1,j), p(i,j+1), \\ \quad p(i-1,j), p(i,j-1) \\ \\ \text{if } q = 8 \\ \quad p(l,m) \mid i-1 \le l \le i+1, \\ \qquad\quad j-1 \le m \le j+1, \\ \quad p(l,m) \neq p(i,j) \end{cases} \tag{1}$$

.

The pixel value of a position (i,j) to $x(i,j)$ is set first. However, in $P_{ij}(q)$, $x(i,j)$ is substituted for the pixel value $x(l,m)$ of the position (l,m) which protrudes picture size. Thereby, peripherals processing is automated. Therefore, the number of $P_{ij}(q)$ is always fixed. This paper explains functions and operators on the assumption that eight neighboring pixels are used at a maximum.

3.3 Pictogram candidate extraction phase

We explain the preprocessing of the proposed algorithm. It carries out in the following procedure.

- Smoothing
- Binalizing
- Size decrease
- Removal of large areas

3.3.1 Smoothing and Binarizing

We describe the detail of each processing. RGB values of inputted image x are smoothed as follows,

$$x_{ij}^{\text{new}} = \frac{1}{8} \sum_{(l,m) \in \hat{P}_{ij}(8)} x_{lm} \tag{2}$$

$$\hat{P}_{ij}(8) = \begin{cases} (l,m) \in P_{ij}(8) \\ \qquad \text{if } x_{ij} - \mathit{diff} \le x_{lm} \le x_{ij} + \mathit{diff} \\ \\ (i,j) \\ \qquad \text{else} \end{cases} \tag{3}$$

When the difference between the pixel value of the attention pixel and the neighboring pixel exceeds diff, the targeted pixel is substituted. diff is a parameter for smoothing.

Smoothing can obscure the complicated background in an image. It can remove noise components. Binalizing is performed by the specific threshold.

$$x_{ij}^{b} = \begin{cases} 1 & \text{if } x_{ij} > \theta_{\text{binary}} \\ 0 & \text{else} \end{cases} \tag{4}$$

3.3.2 Size decrease

We reduce an image size to one ninth in order to remove huge noises. This can reduce the amount of calculations.

We express binary images before reduction as $\boldsymbol{x}_{ij}^{b}$, and images after reduction as $\boldsymbol{x}_{ij}^{\text{second}}$. Size decrease is expressed as follows,

$$\boldsymbol{x}_{ij}^{b} = \{(l,m) \Big| \begin{array}{l} 3(i-1)+1 \le l \le 3(i-1)+3, \\ 3(j-1)+1 \le m \le 3(j-1)+3, \\ i,j \ge 1 \end{array} \tag{5}$$

$$\boldsymbol{x}_{ij}^{\text{second}} = \begin{cases} 1 & \text{if } \sum x_{lm}^{b} \ge 5 \\ 0 & \text{else} \end{cases} \tag{6}$$

3.3.3 Removal of large areas

(1)Estimation of area

Small noises are removed by size decrease in the previous step, However, too large areas which are not pictograms remain. Here, the size of each area is calculated using local information[6].

In [6], pixel values are moved in the direction of the center of the areas. This algorithm is shown in Figure 3. Thereby, the size of the area can be detected as u_{ij}.

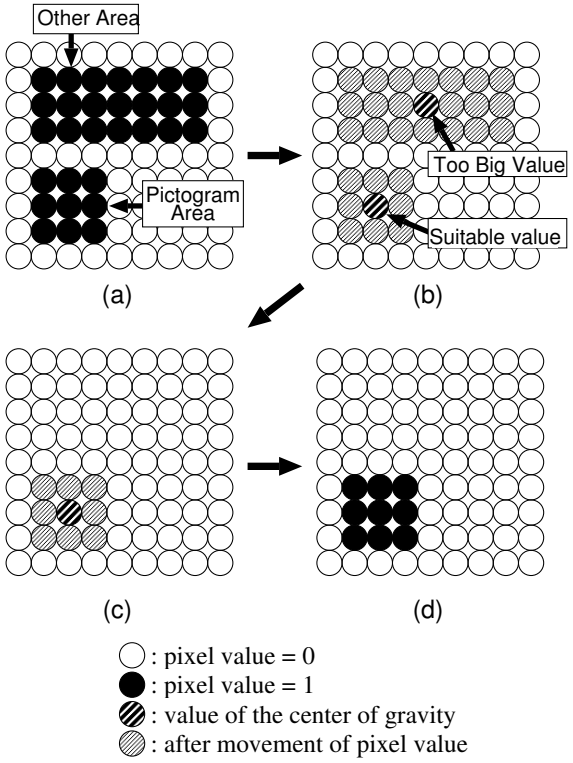

Figure 3: Procedure of judging size

(2)Judgment of size

We set threshold θ_{del} for judgment of area size. The processing element compares the size of area u_{ij} and θ_{del}. Consequently, when $u_{ij} \geq \theta_{\text{del}}$, the operation unit sends a signal that makes pixel value 0 to neighboring PEs. Therefore, too large as a pictogram are removed. The output image which doesn't include unsuitable area is expressed as $\boldsymbol{x}_{\text{c}}$.

Figure 4 shows the effect of judgment of size.

3.4 Square candidate extraction phase

3.4.1 Preprocessing

The proposed algorithm creates the edge image $\boldsymbol{x}_{\text{edge}}$ from inputted RGB image $\boldsymbol{x}$, in order to extract a square area. Edge is made from the difference of an attention pixel and about four pixels.

$$\boldsymbol{x}^{\text{edge}} = \begin{cases} 1 & \text{if } x_{ij} - x_{lm} = 1 \\ 0 & \text{else} \end{cases} \tag{7}$$

3.4.2 The pixel accumulating method

By the pixel accumulating method, horizontal and vertical length of each area in images can be calculated. Moreover, discontinuous lines with inclination are interpolated and the length is calculated as the number of pixels. Each pixel keeps the number of accumulated pixels in its buffer.

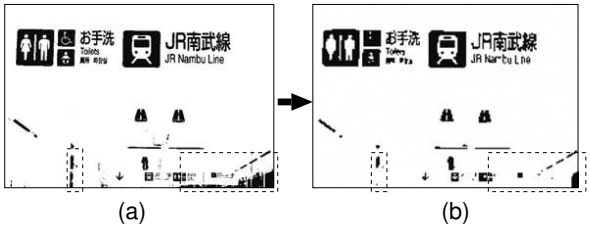

Figure 4: Effect of judgment of size

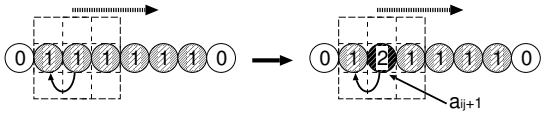

Figure 5: Horizontal pixel accumulation algorithm

3.4.3 Horizontal pixel accumulation

First of all, all pixel values are substituted for accumulation buffers as follows,

$$(a_{ij}, b_{ij}) = (x_{ij}, x_{ij}) \tag{8}$$

We use the buffer of a_{ij} for horizontal pixel accumulation. The buffer of b_{ij} is used for vertical pixel accumulation.

Step1 Accumulation start judging

In horizontal pixel accumulation, if a_{ij} is the pixel value 1, it progresses to Step 2. When there is no pixel value in a_{ij}, the neighboring of the target pixel is referred. When a pixel value exists in neighbors, it is regarded as noise pixel value to 1. Therefore target is set. When there is no pixel value in a_{ij} and there is no pixel value also in neighbors, it proceeds shifts to of Step 3.

Step2 Horizontal pixel accumulation

Left pixel of each pixel is referred. Horizontal pixel accumulation is expressed as follows,

$$a_{ij} = \begin{cases} a_{ij} & \text{if } a_{i-1,j} < a_{ij} \\ a_{ij} + 1 & \text{else} \end{cases} \tag{9}$$

Figure 5 shows the image of horizontal pixel accumulation algorithm. Practically, pixel accumulation algorithm is carried out in by parallel like as shown in Figure 6. This step is repeated.

Step3 End judging

The end judging of horizontal pixel accumulation is shown in Figure7. When a pixel value does not exist in the right-hand of target pixel, it's the end pixel. At the end pixel, the value of buffer a_{ij} is substituted in buffer b_{ij}, and shifts to vertical pixel accumulation.

3.4.4 Vertical pixel accumulation

Vertical pixel accumulation is similar to horizontal

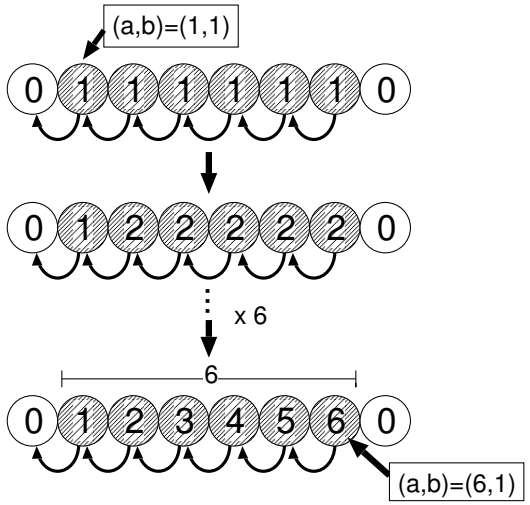

Figure 6: Parallel processing

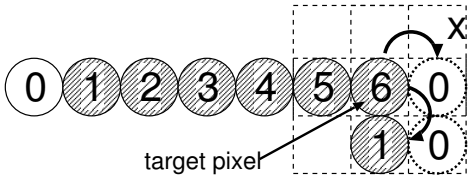

Figure 7: Horizontal to Vertical pixel accumulation

pixel accumulation.

Step1 Vertical pixel accumulation
Upper pixel of each pixel is referred. Vertical pixel accumulation expressed as follows,

$$b_{ij} = \begin{cases} b_{ij} & \text{if } b_{i,j-1} < b_{ij} \\ b_{ij}+1 & \text{else} \end{cases} \quad (10)$$

This step is also performed repeatedly like Horizontal pixel accumulation.

Step2 Vertical pixel accumulation end judging
Pixel accumulation terminates after N_{count} times repetition.

3.4.5 Square area judging

After the pixel accumulation, a square area judging is performed. The difference of a square area judging is calculated from the buffer (a_{ij}, b_{ij}) accumulated by the pixel accumulation method.

$$\delta = |a_{ij} - b_{ij}| \quad (11)$$

Figure 8 shows the restoration of square area. If δ doesn't exceed the square judging threshold θ_{square}, the contents of an accumulation buffer is outputted to a square area restoration phase as a square area. Pixel accumulation is performed toward the lower right. From the pixel of the lower right judged to be a square, the quadrangle of a basis is shortly restored toward

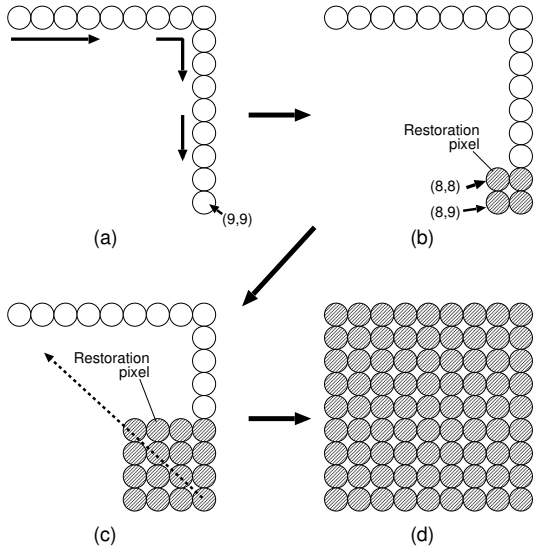

Figure 8: Restoration of square area

the upper left. Restoration reverts reducing every one contents of the buffer(a_{ij}, b_{ij}) which each pixel holds. This restoration processing is stopped when one of the buffer values is set to 0. An accumulation buffer is changed as follows,

$$x(i-l, j-m) = \begin{cases} 1 & \text{if } 0 \le l \le \alpha - 1 \\ & \quad 0 \le m \le \beta - 1 \\ 0 & \text{else} \end{cases} \quad (12)$$

3.5 Pictgram extraction phase

As mentioned above, a square can be restored only from the number information of pixels which is the side of edge, and the vertical length. It is not regarded as a square, and a square domain judging is finished without restoring.

4 COMPUTER SIMULATION

We carried out the computer simulations to extract pictograms from images. We used the pictograms which are used in the stations of Tokyu Lines. We tested 100 scenery images. They were 640×480 pixels taken by a normal digital camera, 20 images were used for setting parameter, and the other 80 images were used for evaluation.

Table 1 shows conditions of computer simulations. Table 2 shows the parameters of computer simulations. The values of parameter were determined based on preliminary simulations.

We defined the detection rate as follows,

$$\text{extraction rate}(\%) = \frac{\text{number of extracted pictograms}}{\text{number of contained in a picture}} \times 100. \quad (13)$$

The extraction rate was 80.7% as a result of the simulation.

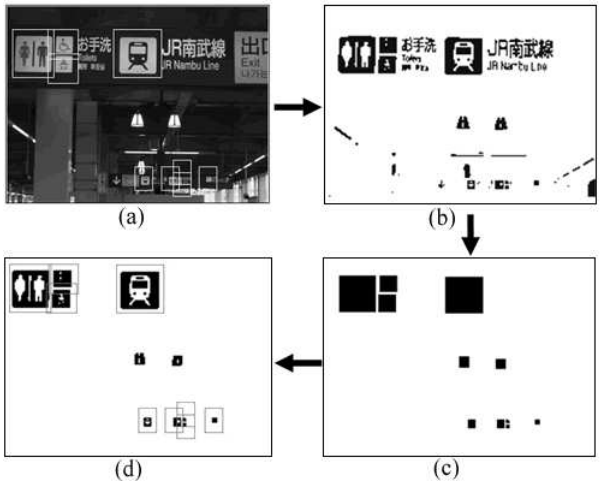

Figure 9: Result

Table 1: Conditions of computer simulation

number of pixels	640×480
number of used images	100
number of the images used for parameter setting	20
number of the images for evaluation	80

The proposed algorithm can extract not only large pictograms but also small pictograms by using the same parameters.

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

$$\text{False detection rate}(\%) = \frac{\text{extracted non-pictogram pixels}}{\text{number of pixels in images}} \times 100. \tag{14}$$

As a result of the simulation, the rate of false detection was 4.5%.

Figure 10 shows an example which failed in extraction. Since this algorithm specializes in extraction of square areas, it is difficult to extract excessively inclined pictograms.

5 CONCLUSION

In this paper, we proposed the algorithm which extracts pictogram areas. This algorithm works under a severe constraint: each pixel of a result image must be derived from information of its neighboring pixels. This constraint is important for a low cost device like a mobile camera. In computer simulations, the proposed algorithm could extract pictogram areas in a rate of 80.7%.

Table 2: Parameter of computer simulation

parameter	value
θ_{binary}	100
θ_{del}	100
θ_{square}	5
θ_{count}	70
N_{count}	70

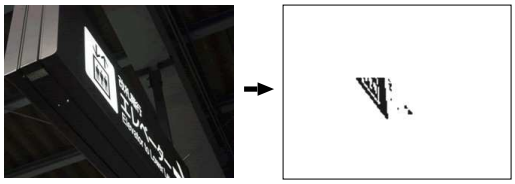

Figure 10: The example of extraction failure

References

[1] Hiroyasu Hiraiwa, Yoshinori Takeuchi, Tetsuya Matsumoto, Hiroaki Kudo, Noboru Oonishi: "The text understanding support system within the environment for a visually impaired person", IEICE Report, PRMU2001-269, pp.195-202, 2002.

[2] Yukiko Shimizu: "Grammar structure of a pictogram", Musashi Institute of Technology Environment information faculty, Information media center journal Vol.3, pp58-63, 2002.4.

[3] Daisuke Matsuura, Hitoshi Yamauchi, Hiromitsu Takahashi: "Extraction of the circular road sign using specific color distinction and area limitation", IEICE Trans.(D-II), Vol.J85-D-II, No.6, pp.1075-1083, 2002.

[4] Yoshiyuki Ajioka: "VISUAL DEVICE", PCT, W000/16259.

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

[6] Satoshi Iwakata, Yoshiaki Ajioka, Masafumi Hagiwara: "Character Region Extraction Algorithm from Scenery Images by Local and Parallel", IEEJ Trans., Electronics, Information and Systems, Vol124, No.4, pp.959-966, 2004.