

P14 Fast Square-area Detection Algorithm using Automata for VLSI Implementation

Junichi AKITA, Kazuhiro MAEDA, Akio KITAGAWA and Masakuni SUZUKI

Dept. of Electrical and Comp. Eng., Kanazawa University

2-40-20 Kodatsuno, Kanazawa, Japan.

TEL:+81-76-234-4863 FAX:+81-76-234-4870 e-mail:akita@ec.t.kanazawa-u.ac.jp

1 Introduction

Conventional image sensors, including CCD sensors and smart sensors are considered for just acquiring the image as the matrix of dots, not for recognizing the meaning of image.

In this paper, we propose a novel real-time algorithm to detect the square area in an object using the structure of node automata, and discuss its implementation as a CMOS image sensor, where the pixels and the node automata are integrated in one chip. We also discuss the search algorithm of their position in the pixel plain using area dividing methodology.

2 Area Detection Algorithm

First, we consider the one-dimensional array of pixels and node automata, as shown in Fig.1. Here the squares at the highest level are pixels, whose colors represents whether they belong to the target object (gray, '1') or not (white, '0'). The circles at the lower levels represents the node automata, which have the value of logical AND of the two pixels or nodes at the previous level, as the following procedures. (Here d_j is the value of pixel at the j -th place, and $S_{i,j}$ is the value of node at the i -th level, and j -th place.)

1. The nodes at the first level, $i = 1$, have the logical AND of value of two neighbor pixels;
 $S_{1,j} = d_j \cdot d_{j+1}$.
2. The nodes at the second level, $i = 2$, have the logical AND of values of two neighbor nodes at the first level; $S_{2,j} = S_{1,j-1} \cdot S_{1,j+1}$.
3. The similar procedures are processed for the lower level, until all of the nodes become '0';
 $S_{i+1,j} = S_{i,j-1} \cdot S_{i,j+1}$.

The node value of '1' at $i = 1$ represents the "block" of two pixels, and the node value of '1' at $i = 2$ represents the "block" of two nodes of $i = 1$, or the "block" of four pixels in other words. It is easily derived that

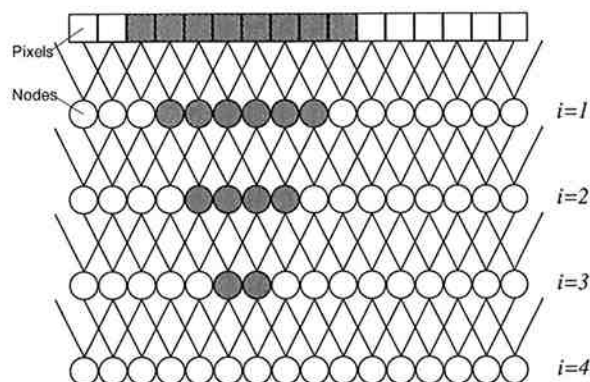


Figure 1: Detection process in one dimensional case

the node value of '1' at i -th level represents the block of 2^i pixels, and all nodes will be '0' at the k -th step if the size of the largest pixel block is $2k$. We can detect the position and size of the block of pixels, by detecting where and when all nodes have become '0.'

It is notable that the nodes determine their values just according to the values of neighbor nodes at the previous step, and they can be easily implemented as the finite state automata connected to only the neighbor nodes.

The above structure and the processes can be extended for the two-dimensional case, by placing matrix pixels with the node automata among the pixels, as shown in Fig.2. In this case, the number of steps needed for detecting the square area whose size is $m \times n$ ($m \leq n$) is $n/2$ at maximum, in other words, the number of detection steps is proportional to $O(n)$, which is fast enough for practical image data.

Here, we describe the simulation results of this algorithm for real images. The original image shown in Fig.3(a) are digitized according to color information as shown in Fig.3(b). The square area detection discussed above is processed for the digitized image, and the processes are shown in Fig.4(a)-(f).

All the pixels are going to eliminated at the 7th step, just after the Fig.4(f), and the all of detection

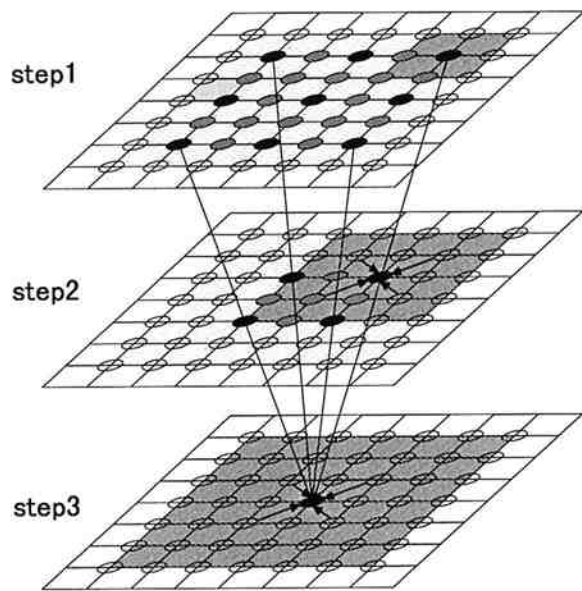


Figure 2: Detection process in two dimensional case

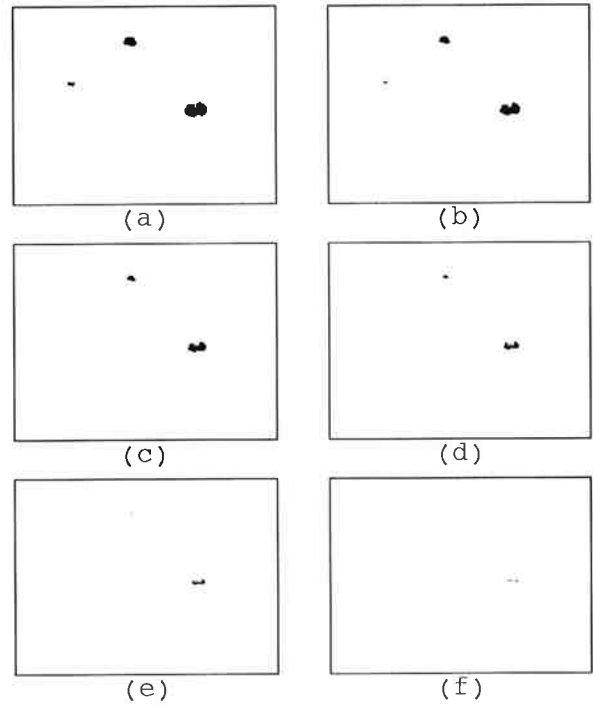


Figure 4: Detection steps

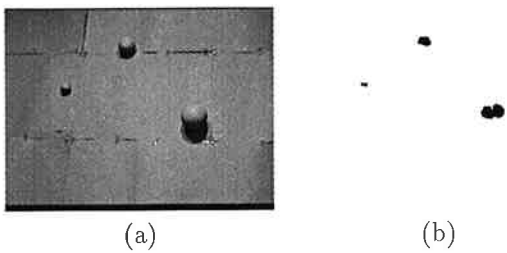


Figure 3: Original sample image(a) and digitized image according to color information(b)

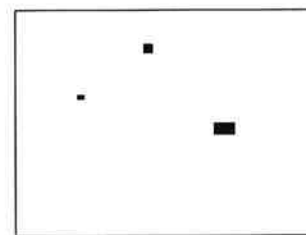


Figure 5: Detected results of each square area

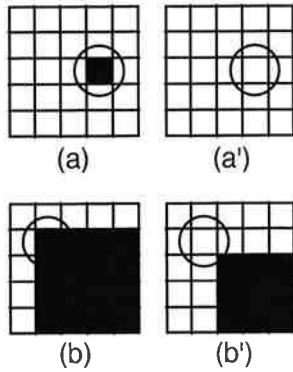


Figure 6: Definition of 'eliminating' pixel. (a) 'eliminating' pixel, (b) pixel does not 'eliminate.'

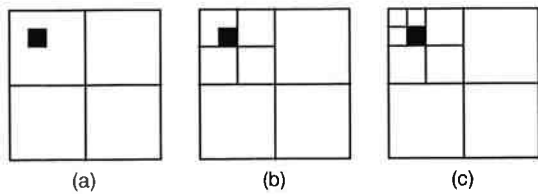


Figure 7: Area division process to detect black ('eliminating') pixel

procedures are finished at this step.

3 Position Search Algorithm

In the process discussed above, the 'eliminating' pixel represents the position where the square area exists, where the word of 'eliminating' represents the node automaton making transition from '1' to '0', whose neighbor pixels also make the same transition, as shown in Fig.6(a), while it does not represent the square area if the neighbor pixels do not make transition to '0', shown in Fig.6(b). Thus the flag of 'eliminating' pixel, $f_{i,j}$ is expressed as follows,

$$f_{i,j} = S_{i,j}^{n-1} \cdot \overline{S_{i,j}^n} \cdot (\overline{S_{i-1,j-1}^n} \cdots \overline{S_{i+1,j+1}^n}) \quad (1)$$

where $S_{i,j}^n$ is the state of node automaton at (i, j) in n -th transition step.

It is unreasonable to scan all pixels to detect the eliminating pixel, since the number of scanning step of all pixel is very large. Figure 7 shows the reasonable process to detect the eliminating pixel.

Here we assume that the logical-OR ('1' for eliminating pixel) of the selected subarea can be directly read out. At the first step(a), the whole area is divided into four subareas, and the logical-OR of upper-left subarea is '1', that implies that there is an eliminating pixel in this subarea. At the following step(b),

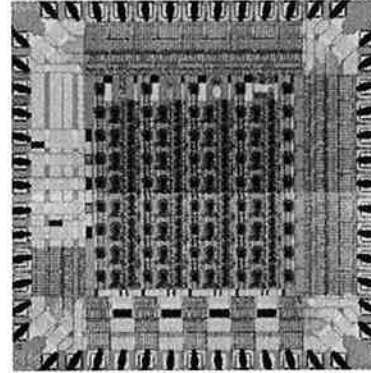


Figure 8: Chip photograph of the first evaluation circuit

the upper-left subarea is divided into four smaller subareas, and then the upper-left smaller subarea should be divided into four subareas at the step(c).

The above scan procedure of image plain generates a kind of encoded image, and it is reported that it is more effective especially in case of less number of eliminating pixels than to scan all pixels[1].

The all of the detection and search procedures are summarized as follows.

1. Read in the digitized image.
2. Make transition of all node automata.
3. If there are any eliminating node automata, scan the position of them by executing area division steps.
4. Repeat the 2. and 3. steps until all node automata transits to '0'.

4 Implementation of Fast Area Detection Algorithm

4.1 Design of state transition evaluation circuit

We have designed the evaluation circuit of processing just the detection procedures. The original image is fed into the circuit serially, and the state transitions of node automata are occurred until all the nodes goes to '0'. The state of each node is read out as the logical-OR of nodes in both the vertical and the horizontal line.

Figure 8 shows the designed circuit using CMOS $1.2\mu\text{m}$, 2 layers metal process¹. The chip size is

¹The VLSI chip in this study has been fabricated in the chip fabrication program of VLSI Design and Education Cen-

