Extended Abstracts of the 2006 International Conference on Solid State Devices and Materials, Yokohama, 2006, pp. 592-593

P-5-5

An Edge Cache Memory Architecture for Early Visual Processing VLSIs

Övgü Öztürk and Tadashi Shibata

Department of Frontier Informatics, Graduate School of Frontier Sciences, The University of Tokyo
5-1-5 Kashiwanoha, Kashiwa-shi, Chiba, 277-8561, Japan
Phone/Fax: +81-3-5841-6797, ovgu@if.t.u-tokyo.ac.jp, shibata@ee.k.u-tokyo.ac.jp

1. Introduction

Biological systems rely on directional edge information for image perception and understanding. Being inspired by such a biological principle, edge-based image representation algorithms have been developed and successfully applied to robust image recognition, like, medical X-ray analysis [1], face detection [2],identification.

Extracting directional edges from an input image, however, is computationally very expensive, thus not being compatible to real-time responding systems. A dedicated VLSI chip was developed for edge extraction and image vector generation. Although it operates at 100 MHz, it outperforms the software processing running on 2.2GHz CPU by a factor as large as 10^5 [3].

The purpose of the present work is to further enhance its performance by introducing a new architecture. Fig. 2 illustrates the proposed system. Extracted edge-flag bits are temporarily buffered in a cache memory, and edge-based image vectors are formed from arbitrarily selected regions on demand. This allows more efficient perception of images in a larger area of visual scenes as compared to our previous chip.

A prototype chip employing the edge cache memory architecture was designed and sent to the fabrication in 0.18 μm 5-metal-layer CMOS technology. In the following sections, we present the proposed architecture, the prototype chip and programmed FPGA chip results.

2. Edge-based Image Representation

Fig.1 illustrates a typical procedure generating an edge-based feature vector from a 64x64 pixel area [1]. In this procedure, first of all, edge flag bits of an input image of 64x64-pixel are extracted in four directions and edge bit maps are obtained. Next, each edge bit map is projected to an appropriate axis to construct 16 elements, totally 64 elements. For each direction, projection is accomplished by dividing the edge bit map into 16 equal groups and adding edge-flag bits for each group to form one vector element. For example, for vertical edges, edge flag bits in every four columns are accumulated to obtain one vector element. Similar procedures apply to other edge bit maps. Finally, 64 elements coming from four different directions form the 64-element feature vector of the image. Feature vector is used to represent the image and stored in the memory to be used for matching during recognition later.

In our earlier chip implementation [3], when a scene is given, feature vectors for consecutive regions are generated by scanning the target image with the 64x64-pixel window pixel-by-pixel. At each step, edge-flag bits for the 64x64-pixel region are extracted and feature vector is constructed. In this approach, for each region, each time, edge flag bits are extracted, used and thrown away after the operation, causing a waste of computationally very expensive information. Moreover, with this region-by-region approach edge information of the different regions can never be accessible at one time, preventing the coordination of the different regions and further development of the recognition system.

In this paper, we describe a new approach which proposes to store the edge information of the whole image in a cache memory as shown in Fig. 2. It brings the thriftiness and provides an on demand real-time vector generation from arbitrarily accessed regions.

3. Edge Cache Memory Architecture

In the proposed architecture extracted directional edge-flag bits of the entire image are stored in a cache memory and made accessible on demand. There are two key issues: first one is the minimum-latency projection of the directional edge flag bits by reading from the memory; second is to define the borders of the region of interest to be projected for feature vector generation.

Vector Generation Procedure

The basic operation of the system is the summation of the edge flag bits within periodic slots. For vertical case, it can be easily done by consecutive add operations of the data read out from the memory. However, for diagonal case, the addition of the diagonally adjacent edge-flag bits is very complicated. To achieve both vertical and diagonal projections, add and shift algorithm has been introduced as explained below.

Fig. 3 shows the basic steps for diagonal case and the grouping of edge-flag bits. In the proposed algorithm, instead of dealing with each bit separately; edge flag bits are grouped in (2x2) 4-pixel units for the sake of simplicity and efficient processing. When add operation is applied, 4-pixel unit is read from the memory for each column group and added to the existing value. When shift operation is applied, the value in the group is shifted to the right. By this method, for vertical projection, the sum of each column group can be obtained by just consecutive add operations. For +45 degree projection, we only need to add one row group and shift it to the right and add the next row group to obtain the sum of the diagonally adjacent units. And finally it is possible by an accumulator to group these sums according to the period of the slot of the projection type.

In Fig. 3, cache memory containing 12x12-pixel edge-flag bits and 2x2-pixel units are shown. Sums of the diagonally adjacent units are as shown sequentially (from bottom-right corner 'X' to top-left corner 'Y').

Cache Memory and Processing Unit Architecture

Fig. 4. demonstrates the system architecture. To achieve simultaneity, 2-port memory architecture is employed. And a processing unit(PU) is attached to each column group for add/shift operations. While defining the borders of the interest region, INMSK signal is used to define the beginning of the interest region, OUTMSK signal is used to define the end. These control signals can be adjusted easily to achieve arbitrary region selection.

3. Measurement and Results

Fig.4 shows the layout of the prototype chip employing 16x12-bit cache memory with PUs, masking circuits and final accumulator. Due to the spice simulations, it is generating feature vector of M-elements in M clock cycles at 1 GHz. Also, FPGA implementation of the architecture was established. Fig. 5 shows the results of FPGA implementation of the sample values in Fig. 3. "OUPUT6" indicates the values coming from the rightmost PU. Sums of the diagonally adjacent units come out sequentially and go to the final accumulator. "result[7..0]" shows the output of final accumulator, adding the incoming values at each clock cycle(CLK2).

4. Conclusions

A real-time on demand feature vector generation from arbitrarily selected regions for image representation was achieved successfully. A prototype chip design (just received from fabrication, under measurement and evaluation process), FPGA implementation were done.

Acknowledgements

The VLSI chip was fabricated in the fabrication program of VLSI Design and Education Center, The University of Tokyo with Hitachi Ltd. and Dai Nippon Printing Corporation.

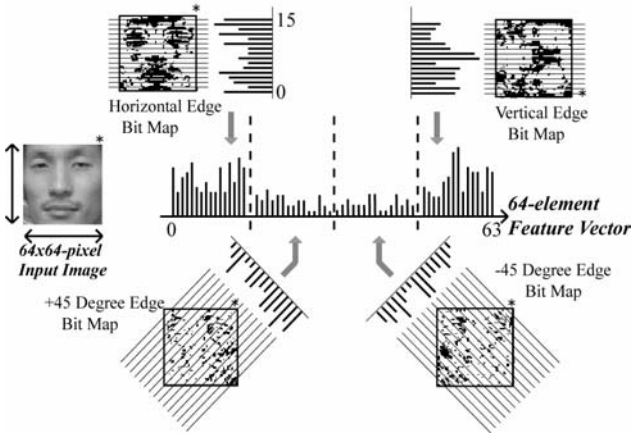

Fig. 1 64-element feature vector generation based on PPED[1].

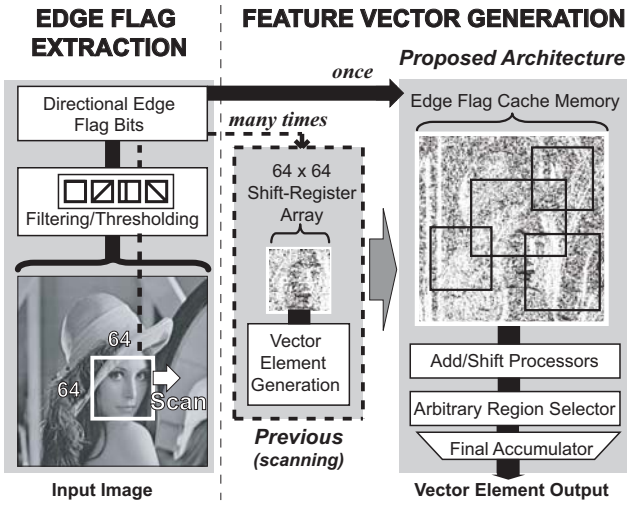

Fig. 2. Proposed Feature Vector Generation Approach.

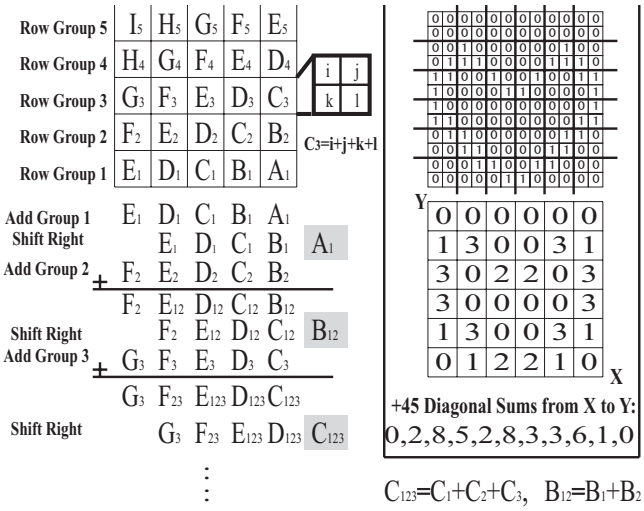

Fig. 3. +45 Degree projection steps and a 12x12 sample memory

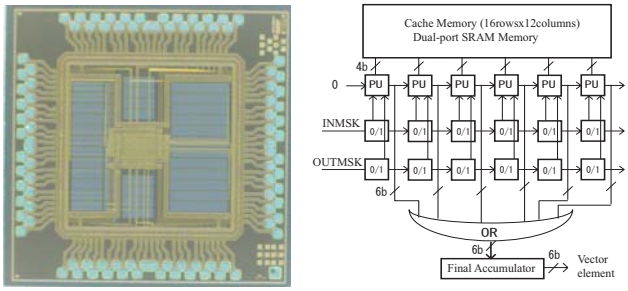

Fig. 4. Prototype Chip Photo, General Architecture

References

[1] M. Yagi and T. Shibata, *Trans. Neural Networks*, Sep. 2003.

[2] Y. Suzuki and T. Shibata, Proc. EUSIPCO, Sep. 2004.

[3] H. Yamasaki and T. Shibata, ESSCIRC, Sep. 2005

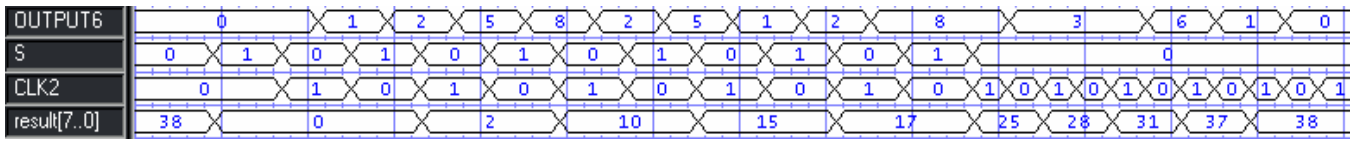

Fig. 5. Results of the FPGA simulation, showing the +45 Degree diagonal sums.