A 22.4 mW Competitive Fuzzy Edge Detection Processor for Volume Rendering

Joonsoo Kwon, Minsu Kim, Jinwook Oh, and Hoi-Jun Yoo
Department of Electrical Engineering
KAIST
Daejeon, Republic of Korea
js1616@eeinfo.kaist.ac.kr

***Abstract*— A low power competitive fuzzy edge detection (C-FED) processor is proposed for gradient calculations in volume rendering. Its linearized fuzzy membership function reduces overall power by 35.1% and the proposed hardware sharing between computation stages reduces power consumption by 18%. Threshold adaptive bit control scheme is proposed to pre-determine background pixel with simple operation which results in 13% power reduction. Overall power consumption is reduced by 53.8%. Its power consumption and energy per pixel is 22.4 mW and 0.14nJ/pixel, respectively, at 1.8-V supply. The fabricated processor occupying 450 μm x 450 μm in a 0.18 μm CMOS process achieves 1821.5fps for the input image of 300 x 300 pixels at 200 MHz operating frequency.**

I. INTRODUCTION

Edge detection is the process of identifying edge points out of an input image whose gradient values are maxima in pixel brightness. Since it is a basic pre-processing of various image processing applications, it has been actively researched for several years [1-3]. Among many edge detection algorithms, Sobel and Canny[4] are most widely used. Recently, edge detection is applied to a gradient calculation stage of volume rendering [5-6]. For real-time operation of volume rendering, hardware implementation of edge detection is needed, and only Sobel algorithm has been mapped to hardware from entire edge detection algorithms [7]. However, edge results of Sobel can be unsatisfactory because of its weakness at noise robustness, edge thickness and lack of adaptability of sensitivity.

In this paper, we propose a low power edge detection processor for volume rendering based on competitive fuzzy edge detection (C-FED) algorithm [8]. The C-FED algorithm is suitable for volume rendering because it obtains sharp and noise robust edge outputs than Sobel algorithm with adaptive edge sensitivity control process.

Since C-FED can obtain sharp and noise-robust edge, it is employed for low power edge detection processor. In order to reduce power consumption of C-FED processor, software-hardware co-optimization is applied. Firstly, we modified the C-FED algorithm with linearized membership function. By linearizing membership function, we can eliminate complex multiplication and division in operation. Second, by sharing hardware components between operation stages, it is employed for low power edge detection processor. Third, proposed threshold adaptive bit control scheme avoids unneeded calculations by pre-determining some of background (BGND) pixels. With proposed three schemes, the overall power consumption can be reduced by 53.8%. Finally, implemented C-FED processor in a 0.18μm CMOS process consumes only 22.4 mW.

II. ALGORITHM

A. Conventional C-FED

Fig. 1 shows stages of C-FED algorithm. The first step is to calculate the gradient in 4 directions. The next step is to classify gradient vectors into one of 6 classes. Each class is represented by its threshold vector c_i ($0 \leq i \leq 6$). The Epanechinicov fuzzy set membership function which is shown in Eq.(1) decides the closest threshold vector by the comparison with the gradient vectors .

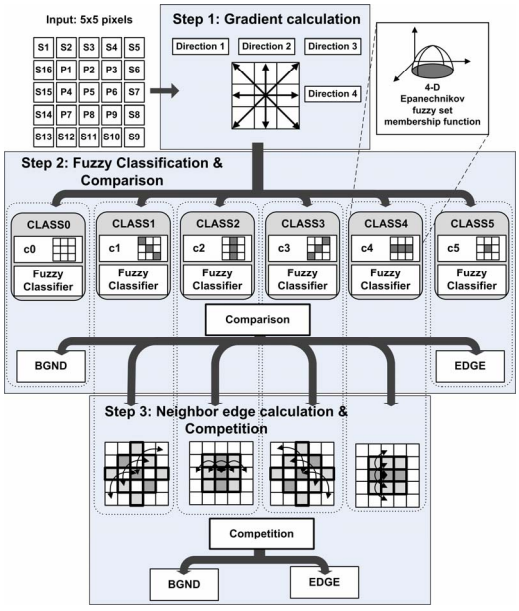


Figure 1. Overview of C-FED algorithm

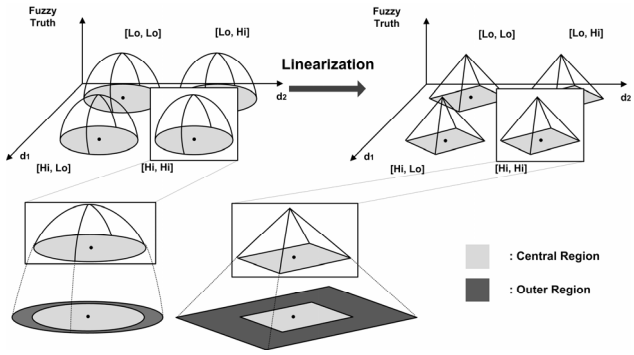


Figure 2. 2-D Conventional membership function and linearized function

$$u(x) = \max\{0,\ 1 - \|x-c\|^2 / w^2\} \qquad (1)$$

If the pixel is classified as an edge (class 1-4), then competition between neighboring pixels occurs to decide which pixel is the strongest edge. Otherwise, the pixel is classified as a BGND pixel (class 0) or a speckle pixel (class 5) without competition.

B. Membership function linearization(MFL)

Linearization of fuzzy membership function in Eq.(1) is proposed in order to avoid multiplication and division. Since pixel is detected by comparing fuzzy value of each class, Eq.(1) can be multiplied by same factor w^2 on both 0 and second order term without any differences in result decision.

$$u(x) = \max\{0,\ w^2 - \|x-c\|^2\} \qquad (2)$$

The linearization result of Eq. (2) is expressed as following Eq. (3).

$$u(x) = \begin{cases} \max\{0,\ (w-c)x + c^2\} & (x_i < c) \\ \max\{0,\ -(w-c)x + 2w^2 - c^2\} & (x_i > c) \end{cases} \qquad (3)$$

Additional parameters such as s, t_0 and t_1 are introduced so that multiplications can be implemented using simple shift operations as shown in Eq.(4) ($s = \log_2[w\text{-}c]$, $t_0 = c^2$, $t_1 = 2w^2 - c^2$)

$$u(x) = \begin{cases} \max\{0,\ x << s + t_0\} & (x_i < c) \\ \max\{0,\ -x << s + t_1\} & (x_i > c) \end{cases} \qquad (4)$$

Fig.2 shows 2-D graph of a conventional membership function and a linearized membership function. Since class decision is made by comparing relative magnitude of each membership function and only central region near threshold vector which reduces error of approximation is concerned, MFL has negligible effect on the edge output result while reducing power consumption by 35%.

III. System Architecture

Fig.3 shows the overall architecture of proposed C-FED processor. The processor consists of controllers, line memories, and a window processor. Controller unit manages overall operations of the entire processor. Memory controller reads input pixels from memories and distributes to the proper line memories.

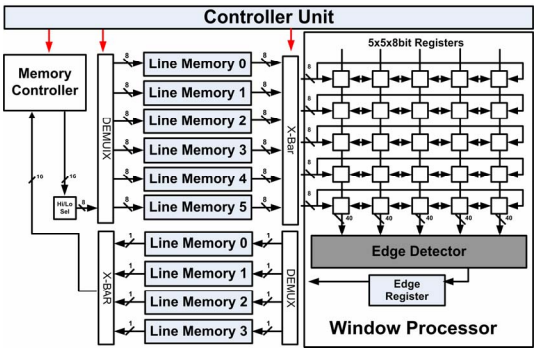


Figure 3. Overall system architecture of C-FED processor

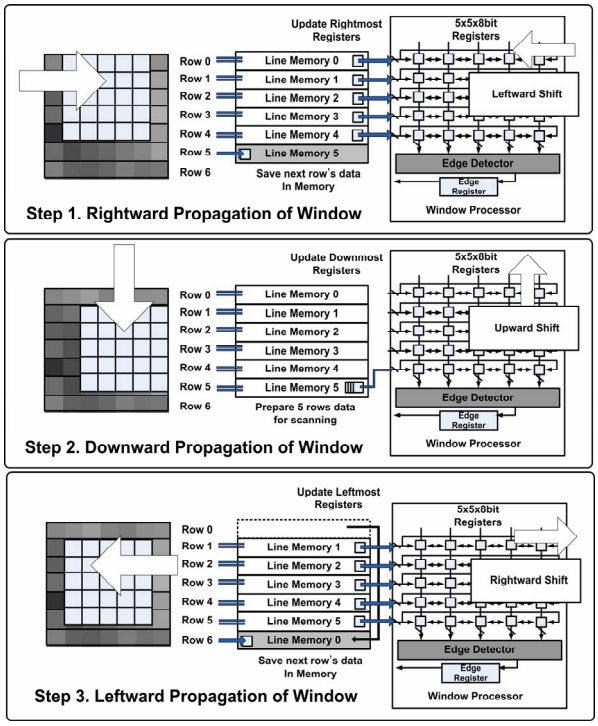


Figure 4. Meander propagation sequence for time-hiding memory

The processor contains 6 input line memories and 4 output line memories. Each line memory stores all pixels of one row of input image. To calculate sub- direction edges, 5x5 pixels are required to calculate class of one center pixel. In output generation stage, neighboring 3 pixels are also candidates for edge decision. Therefore 5 input memories and 3 output memories are required. However, 1 additional memory for each input and output memories are required respectively to hide delay of reading and writing process of memory in meander propagation. The detailed operation of time-hiding memory is shown in Fig. 4.

IV. Window Processor Design

Fig. 5 shows the proposed algorithm and architecture of window processor which can be largely divided into the finite state machine (FSM) part and the datapath. The datapath contains sub blocks including gradient calculation unit(GCU)s, fuzzy classifiers, and max decision units.

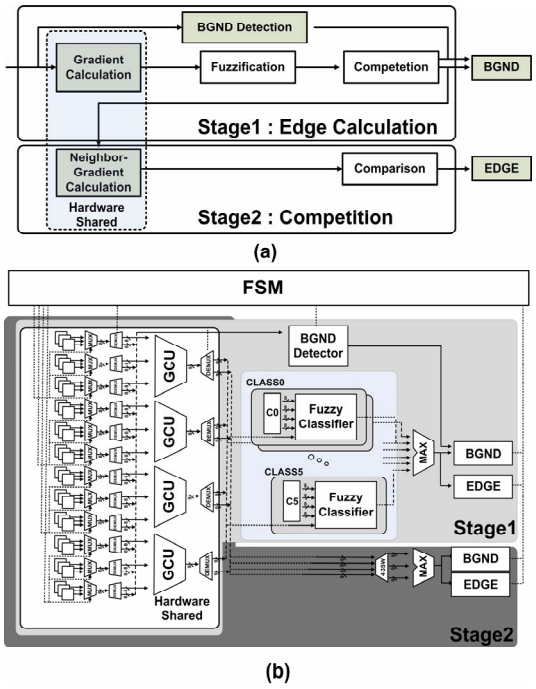


Figure 5. (a) Proposed edge detection algorithm and (b) window processor

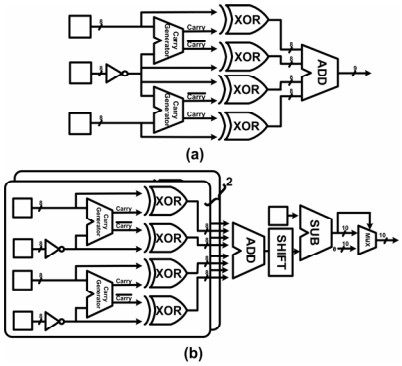


Figure 6. (a) Gradient calculation unit (b) fuzzy classifier

GCU in Fig.6 (a) calculates sum of absolute differences (SAD) between 3 input pixels. Fuzzy classifier in Fig.6 (b) also performs SAD operation in order to calculate differences between input pixels and corresponding threshold vector for each class. Due to linearization of fuzzy membership function, it operates without multiplication and division.

A. Hardware sharing(HS)

4 GCUs are shared in window processor as shown in Fig. 5 (b). In order to reduce power and area consumption, gradient calculation is shared between *Stage 1: Edge Calculation Stage* and *Stage 2: Competition Stage*. As shown in Fig. 5 (a), BGND class pixels require only Stage 1 for decision while edge class pixels need both stages. Since ratio of BGND class pixels is larger than that of edge class pixels in usual cases, the processor operates without considerable degradation of throughput due to HS while power consumption is reduced by 18%.

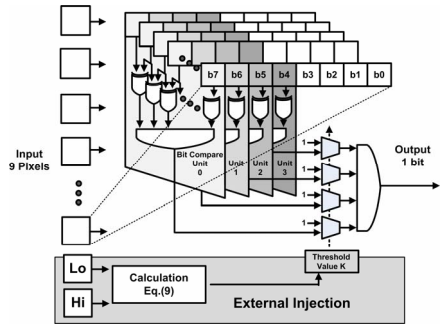


Figure 7. BGND detector with TABC

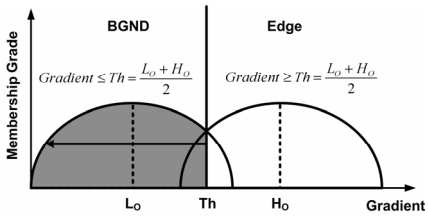


Figure 8. 1-D Fuzzy membership function versus gradient input

B. BGND detector with threshold adaptive bit control (TABC) scheme

To reduce the power consumption, TABC scheme is proposed to pre-determine the BGND pixel without utilization of following GCUs and fuzzy classifiers. Fig. 7 shows the BGND pixel detector with TABC scheme implemented by simple XOR gates and AND gates. Output bit sets to high when the center pixel is classified as a BGND class.

The main principle of BGND detector is that BGND pixel and other neighboring pixels have similar value since BGND pixel has small gradient value on each direction. In hardware realization, BGND is decided if several MSBs have the same value through all input pixels.

To effectively measure similarity, the TABC scheme adaptively determines the number of MSBs for comparison corresponding to the threshold value. Therefore, only essential MSBs are activated while the redundant transitions in the datapath for LSBs are inhibited. In addition, false detection of BGND never occurs because TABC uses the sufficient condition for classifying a pixel as BGND.

In the proposed architecture, the maximum number of MSBs "K" according to the threshold input values (Lo, Hi) is determined by

$$K = 9 - \left\lfloor \log_2 \left| (Lo + \mathrm{Hi})/2 \right| \right\rfloor. \quad (5)$$

If the pixel has determined to be a BGND class by TABC scheme, then the absolute difference between center pixel(p_5) and arbitrary pixel(p_i) is less than 8- K bits number since upper K bits are already compared to be the same , which can be described as follows.

$$\left\| p_5 - p_i \right\| < 2^{8-K} \quad (6)$$

A gradient magnitude (g_i) of one direction, represented by sum of two Eq.(6) s, results in Eq.(7).

$$g_i = \| p_5 - p_i \| + \| p_5 - p_j \| < 2^{(8-K)+1} = (Hi + Lo)/2 \quad (7)$$

The fuzzy membership function is symmetrical with respect to the threshold vector value in both conventional and linearized cases as shown in Fig. 8. If gradient value in one direction is less than the mean value of threshold, then the gradient is classified as "Lo" class**.** All the gradients of target pixel of TABC scheme are classified as "Lo" classes. Since threshold vector of BGND class is [Lo Lo Lo Lo], TABC guarantees that the pixel is BGND.

V. IMPLEMENTATION RESULTS

The test edge detection images are shown in Fig.9 to show the proposed processor's ability to produce thin lines in correspondence with object contours. We adopt the test noised image shown in Fig.9 (a). The result processed by Sobel edge detection is represented in Fig.9 (b). Fig.9 (c) is the result of the proposed processor. The better performance edge detection is obviously distinguished in the result given by the proposed processor.

The proposed competitive fuzzy edge detection window processor is fabricated in a 0.18μm CMOS technology. Fig.10 shows a chip photograph and implementation summary of proposed window processor. It occupies 0.2025mm^2. The frame rate is 1821.5fps for image of 300x300 pixels. The average dissipated power is 22.4mW while the processor operates competitive fuzzy edge detection at 200MHz frequency. The energy consumption is 0.14nJ/pixel.

Fig. 11 shows the measured waveforms of input image bit and output edge bit at 200 MHz clock. In case of BGND (B) pixel, output bit occurs after 1 cycle. Otherwise if edge pixel is an edge, operation takes 2 cycles.

Fig.12 summarizes total power reduction. As a result, MFL, HS, and TABC achieve 35.1%, 18%, 13% power reduction respectively. Finally, total 53.8% power reduction is achieved.

VI. CONCLUSION

A 22.4 mW low power C-FED processor is designed and implemented in 0.18 μm CMOS process. Fuzzy membership function is linearized for efficient hardware mapping. The C-FED processor shares hardware between computation stages to reduce power. TABC scheme is proposed to pre-determine BGND pixel with simple operation. Proposed schemes reduce overall power consumption by 53.8%. As a result, 22.4mW power consumption is achieved while the frame rate is 1821.5 fps for image of 300 x 300 pixels at 200 MHz operating frequency.

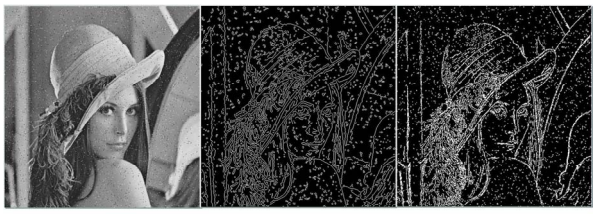
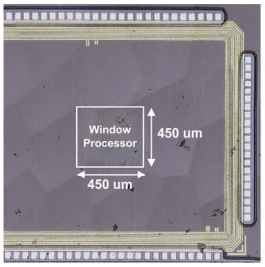


Figure 9. (a) Input image (b) Sobel edge detection result (c) C-FED result



Process Technology	1-poly 4-metal 0.18μm CMOS technology
Area	450 um X 450 um
Power	22.4mW @ 1.8V
Operating Frequency	200MHz
Frame Rate	1821.5 fps
Input Image	300x300 pixels
Energy/pixel	0.14nJ/pixel

Figure 10. Chip photograph and Features

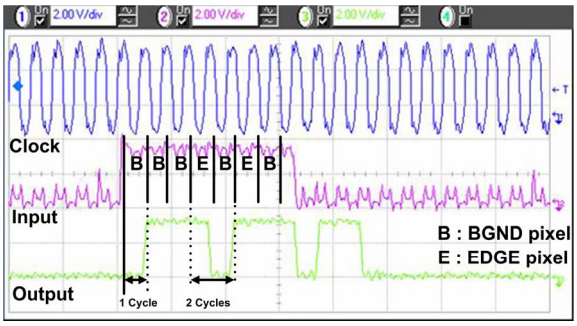


Figure 11. Measured waveform

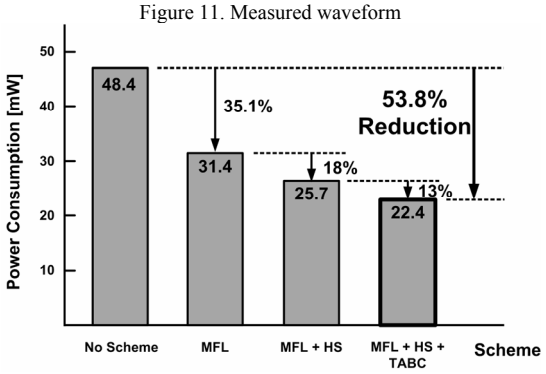


Figure 12. Power reduction results by proposed schemes.

REFERENCES

[1] Q. Ying-Dong, C. Cheng-Song, C. San-Ben, L. Jin-Quan"A fast subpixel edge detection method using Sobel–Zernike moments operator,"Image and Vision Computing, vol.23, pp. 11–17, 2005.

[2] D. Heric, D. Zazula, "Combined edge detection using wavelet transform and signal registration," Image and Vision Computing, vol.25, pp.652–662, 2007.

[3] G. Grassi, P. Vecchio, E. Di Sciascio, L. A. Grieco and D. Cafagna, "Neural networks for image processing: new edge detection algorithm," 2007 IEEE International Conference on Electro/Information Technology, pp. 498-502, 2007.

[4] J. Canny, "A Computational Approach to Edge Detection," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 8, pp. 679-698, 1986.

[5] J. Kniss, G. Kindlemann, C. Hansen, "Interactive Volume Rendering Using Multi-Dimensional Transfer Functions and Direct Manipulation Widgets," IEEE Visualization Conference, pp. 255 - 262, 2001.

[6] M.P. Persoon, I.W.O. Serlie, F.H. Post, R. Truyen, F.M. Vos, "Visualization of Noisy and Biased Volume Data Using First and Second Order Derivative Techniques," IEEE Visualization Conference, pp. 379 - 385, 2003.

[7] N. Kazakova, M. Margala, and N. G. Durdle, "Sobel edge detection processor for a real-time volume rendering,", IEEE International Symposium on Circuits and Systems , vol.2, pp. 271–350, 2004.

[8] L.R. Liang, C.G. Looney, "Competitive fuzzy edge detection," Applied Soft Computing, vol.3, pp. 123–137, 2003.