

3-Line 버퍼를 사용한 실시간 Sobel 윤곽선 추출 블록 FPGA 구현

FPGA Implementation for Real Time Sobel Edge Detector Block Using 3-Line Buffers

박 찬 수*, 김 희 석**

Chan-Su Park*, Hi-Seok Kim**

Abstract

In this Paper, an efficient method of FPGA based design and implementation of Sobel Edge detector block using 3-Line buffers is presented. The FPGA provides the proper and sufficient hardware for image processing algorithms with flexibility to support Sobel edge detection algorithm. A pipe-lined method is used to implement the edge detector. The proposed Sobel edge detection operator is an model using of Finite State Machine(FSM) which executes a matrix mask operation to determine the level of edge intensity through different of pixels on an image. This approach is useful to improve the system performance by taking advantage of efficient look up tables, flip-flop resources on target device. The proposed Sobel detector using 3-line buffers is synthesized with Xilinx ISE 14.2 and implemented on Virtex II xc2vp-30-7-FF896 FPGA device. Using matlab, we show better PSNR performance of proposed design in terms of 3-Line buffers utilization.

요 약

본 논문에서는 3-Line buffers를 사용하여 Sobel 윤곽선 추출 블록을 FPGA로 효율적으로 설계하여 구현하고자 한다. FPGA는 영상처리 알고리즘 중 하나인 Sobel 윤곽선 추출 알고리즘을 처리하기에 적절한 환경을 제공한다. 윤곽선 추출을 위한 방법으로는 파이프라인 방법을 사용하였다. Sobel 윤곽선 연산에서 윤곽선 강도 레벨을 결정하기 위하여 유한 상태 기계로 구현 된 마스크 연산을 이용한 모델을 제안한다. 효율적인 LUT 및 플리플롭의 사용으로 시스템의 성능이 향상됨을 입증하였다. 제안하는 3-line buffers를 이용한 Sobel 추출 연산은 Xilinx 14.2으로 합성하고 Virtex II xc2vp-30-7-FF896 FPGA device으로 구현하였다. Matlab을 이용하여 제안된 3-Line buffers 설계 시 PSNR 성능이 향상됨을 확인하였다.

Key words : edge, Sobel edge, Sobel Mask, 3-Line buffer, FPGA

* Dept. of Electronics Engineering, Cheongju University, dachang57@nate.com, 043-229-8452

★ Corresponding author

Dept. of Electronics Engineering, Cheongju University, khs8391@cju.ac.kr, 043-229-8452

※ Acknowledgment

This research was partially supported by the Cheongju University Research Scholarship Grants in 2013

This work was supported by the Industrial Core Technology Development Program(10049192, Development of a smart automotive ADAS SW-Soc for a self-driving car) funded By the Ministry of Trade, industry & Energy.

Manuscript received Jan. 20, 2015; revised Feb. 23, 2015 ; accepted Mar. 2, 2015

This is an Open-Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/by-nc/3.0>) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

I. 서론

영상 처리 분야에서 카메라의 물체 인식을 위해 사용되는 3차원 입체영상 기술은 고화질 TV와 고성능 컴퓨터 분야에 다양하게 응용되고 있다. 영상 기반 모델링(modelling) 및 렌더링(rendering) 기술은 컴퓨터 그래픽스 기술로 모델링하기 어려운 복잡한 3차원 영상을 쉽게 구현할 수 있어, 3차원 영상을 이용한 콘텐츠(contents)를 쉽게 제작할 수 있도록 보급되고 있으며, 입체 영상의 응용 사례들이 많이 보고되고 있다. 하지만, 실시간 입체 영상에 대한 처리를 위해서는 많은 처리량에 의한 속도 문제로 많은 개선이 필요하다. 입체 영상의 자연스러운 구현을 위해서는 카메라에서 물리적으로 발생하는 색상 및 회전, 이동 등의 차이들을 보정해야만 한다. 이러한 문제점 등을 보정함으로써 소프트웨어와 하드웨어를 통하여 영상 기술이 구현되어 있지만 영상의 고해상도에 따라 많은 처리량의 문제로 기존 기술로 구현하는 것은 어려움이 있다.[1] 이런 이유로 고해상도 영상의 회전 및 이동에 따른 보정을 단순화하기 위해서는 영상을 대표하여 단순하게 처리할 수 있는 윤곽선의 추출이 매우 중요하며[2], Sobel mask는 윤곽선을 추출하기 위해 가장 잘 알려진 방법이다. 이 방법은 잡음에 의한 간섭을 줄이고 윤곽선을 선명하게 추출하여 영상을 단순화 시켜 보정하는데 적합하며 [3-4] 하드웨어로 설계한 방식으로는 Rajesh가 제안한 Line Buffer로 전체 영상 이미지를 저장하는 방법[5]과 Weili가 제안한 2-FIFO (2-First In First Out)을 활용하여[6] 각 이미지의 가로 크기만큼의 각각의 FIFO에 저장한 뒤 스캔하는 방법은 잘 알려져 있다. 하지만 처리 속도가 느리며 이것을 처리하는데 상당한 크기의 메모리를 필요로 한다.

본 논문에서는 3-Line buffer 기반의 Sobel 윤곽선(Sobel edge) 데이터 추출함으로써 기존의 Sobel mask를 사용한 것보다 동작 속도가 빠르고, 상대적으로 작은 메모리를 할당하는 등의 방법을 제안한다.

3-Line buffer 방법은 크게 3가지의 부분으로 구성된다. 첫 번째는 각 영상의 3x3 크기로 pixel 정보로 추출한 뒤에 추출된 정보를 3개의 pixel (A_Data, B_Data, C_Data)으로 분할 출력하는 영상 데이터 제어, 두 번째는 첫 번째 방법을 통해 순차적으로 출력되는 3개의 pixel line들(A_Data, B_Data, C_Data)을 순차적으로 임시 저장하는 line buffer, 세 번째는 line buffer로부터 3개의 pixel line들(A_Data, B_Data, C_Data)을 순차적으로 제공받아 각 pixel line 별로 Sobel 연산을 수행하여 픽셀 라인의 윤곽선 데이터를

추출하는 Sobel mask filter로 구성된다.

본 논문의 2장에서는 Sobel edge 검출에 관한 이론에 대해서 기술하고 3장에서는 3-Line buffer 방법을 활용한 하드웨어 설계 방식에 관해 기술한다. 4장에서는 3-Line buffer를 이용한 방법과 Rajesh가 제안한 방법에 대하여 동작 속도와 PSNR을 이용한 평가 결과에 대하여 비교하였고, 마지막으로 5장에서 본 논문의 결론을 맺는다.

II. Sobel edge detection

1. Sobel edge

edge는 영상의 밝기가 낮은 부분에서 높은 부분으로 또는 이와 반대로 변하는 지점에 존재하는 부분을 의미한다. 이를 이용하여 영상 안에 있는 pixel 간의 객체의 경계를 찾을 수 있으며 영상 처리 영역에서 기술적으로 간단하고 유용한 방법으로 활용되고 있다. 영상의 상황에 따라 다양한 패턴으로 구성된다. 그 중 그림 1과 같이 edge는 밝기가 급격하게 변화하는 구간을 비롯하여, 광원의 변화, 물체 표면의 반사율 변화, 물체 형상에 의한 가림 등의 요소가 있다. 그림 1은 pixel과 다음 pixel 사이에 변화가 생겼을 때 이 부분의 미분 성분이 edge으로 나타난다.

그림 2와 같이 영상의 기울기를 찾기 위해서는 영상 이미지에서는 데이터가 일정한 간격으로 나열되어 있기 때문에 중심 픽셀을 기준으로 주변 픽셀들과의 차를 취하는 연산인 1차 미분으로 근사치를 구한다.

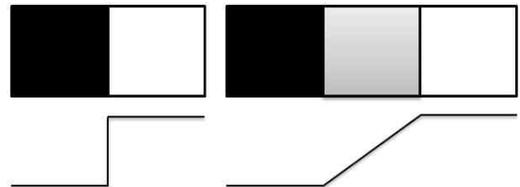


Fig. 1. Edge data about brightness change

그림 1. 휘도 변화에 의한 윤곽선 데이터

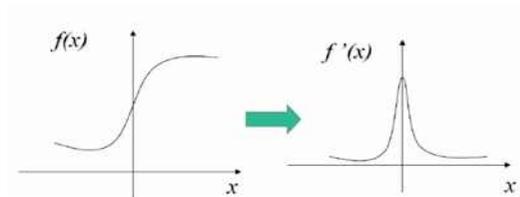


Fig. 2. Edge data of differential ingredient

그림 2. 미분 성분의 윤곽선 데이터

식 1은 edge의 방향성을 의미하고 식 2는 edge의 강도를 의미한다. 1차 미분을 활용하면 edge에 관한 강도를 구할 수 있다. 1차 미분을 영상에 적용하기 위해서는 Sobel mask를 적용한다.[7]

$$\theta = \tan^{-1}\left(\frac{\partial I}{\partial y} / \frac{\partial I}{\partial x}\right) \quad (1)$$

$$\nabla I = \sqrt{Gx^2 + Gy^2} \approx |x| + |y| \quad (2)$$

그림 3과 같이 영상에서 3x3 block에 대한 중심 pixel을 (x, y)로 기준으로 식 3과 같이 수평 방향 edge, 식 4와 같이 수직 방향 edge를 추출한다. 식 5와 같이 추출한 수평 edge와 수직 edge에 절대 값을 취한 뒤에 연산하게 되면 1차 미분에 의한 edge를 추출할 수 있다.[8]

$$G(x) = f(x-1, y) - f(x+1, y) \quad (3)$$

$$G(y) = f(x, y-1) - f(x, y+1) \quad (4)$$

$$Edge = |G(x)| + |G(y)| \quad (5)$$

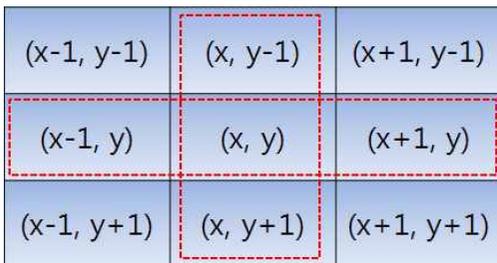


Fig. 3. 3x3 block matching to replace the derivative
그림 3. 미분을 대체할 3x3 블록 매칭

1차 미분을 이용하여 마스크로 바뀌 edge 추출을 위해서는 edge에 해당하는 주변 pixel 값을 검색하는 과정이 필요하다. 이에 따라 필요한 마스크 패턴으로 잡음을 제거하는 전처리 과정을 진행해야한다.

2. Sobel Mask

edge 추출을 위한 Mask pattern 중 가장 대표적인 Sobel mask는 그림 4와 같이 두 가지의 Sobel mask pattern을 활용하는데 Vertical mask는 가로 방향 x축 벡터 성분을 가지게 되고 Horizontal mask는 세로 방향 y축 벡

터 성분을 가지고 된다. 가운데 중심 pixel의 주변 음수와 양수 Mask pattern 값에 따라 해당 pixel data를 곱한 후 전체를 더하면 Sobel edge가 추출 되게 된다.



Fig. 4. Sobel mask pattern
그림 4. Sobel mask 패턴

Sobel mask는 모든 방향에 따라 x, y축 edge data를 추출하고 식 5에 따라 절대 값을 취한 후 합치기 때문에 잡음에 강한 편이다. 또한 대각선 방향의 edge에 분포된 성분이 다른 마스크 패턴보다 높기 때문에 활용 범위가 넓다.

III. 3-Line Buffer Method

1. Image data control

Image data control(영상 정보 제어)는 그림 5와 같이 영상 정보 데이터를 각각의 Line-buffer에 저장하도록 어드레스(Write_addr) 및 Sobel mask filter의 Sobel 연산 동작 신호를 제공하는 것으로 설계되었다.

영상 정보 제어는 설계된 3개의 Line-buffer (A_Line_buffer, B_Line_buffer, C_Line_buffer)로 입력 순서에 맞추어 차례대로 A, B, C Line buffer에 어드레스와 저장 데이터 신호를 활성화하며 3번째 Line buffer의 저장 공간이 설정되는 상태가 될 때에 Sobel mask 로 Sobel 연산 동작 신호(C_Write_enable)를 출력하는 것을 특징으로 한다.

2. Line buffer

영상 정보 제어에서 순차적으로 출력되는 3개의 pixel line들을 순차적으로 임시 저장하는 기능을 수행한다. 그림 6의 상태도와 같이 영상 정보 제어로부터 입력 받는 어드레스 신호의 활성화에 따라 3개의 pixel line들을 각각 저장하며 3개의 pixel line들 중 첫 번째 pixel line을 임시 저장하는 Line buffer, 두 번째 pixel line을 임시 저장하는 Line buffer, 세 번째 pixel line을 임시 저장하는 Line buffer를 포함하여 설계된다. 각 Line buffer의 크기는 영상 크기의 M x N 에서 8bit (Gray scale) x depth M이 된다.

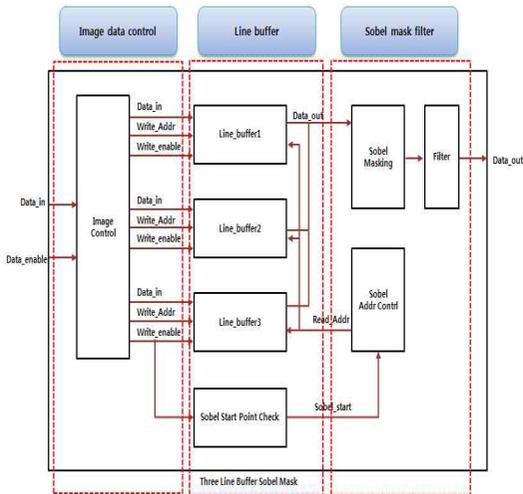


Fig. 5 3-Line buffer Sobel mask Block
그림 5. 3-Line buffer Sobel mask 블록도

3. Sobel mask filter

Sobel mask filter는 Line buffer로 부터 3개의 pixel line 들을 시계 방향 순으로 데이터를 입력 받아 pixel line 별로 Sobel 연산을 수행하여 Sobel edge data를 추출하는 기능을 한다.

하드웨어 설계 시 기존의 Rajesh 알고리즘 같이 직렬화 하여 Line buffer에 전체 이미지를 저장하고 읽게 되면 그림 6과 같이 한 영상을 담은 메모리에 Sobel mask를 처음 부터 끝까지 mask pattern의 변화 없이 적용 하는 방식이므로 mask pattern에 따른 추가적인 조정이 필요 없다. 하지만 전체 영상을 저장하기 때문에 많은 Clock(CLK)이 소모되고 메모리의 크기가 증가하게 되어 추가적인 보정 블록들을 설계하는데 FPGA(field-programmable gate array)의 메모리 크기의 제약이 생기게 된다.

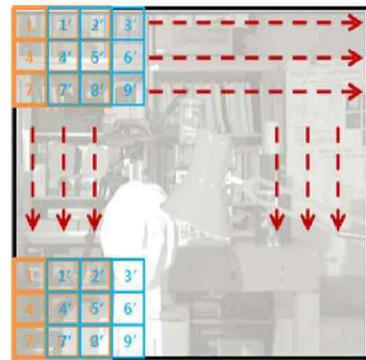


Fig. 6 scan process of Sobel mask
그림 6. 소벨 마스크의 스캔 과정

본 논문의 실시간 Sobel edge 추출을 위하여 고속으로 수행하는 하드웨어 설계 시, 메모리 크기를 감소시키고 동작 속도를 증가시키기 위하여 3-Line*(Image Width)으로 설계된다. 그림 8과 같이 3-Line에 영상의 data가

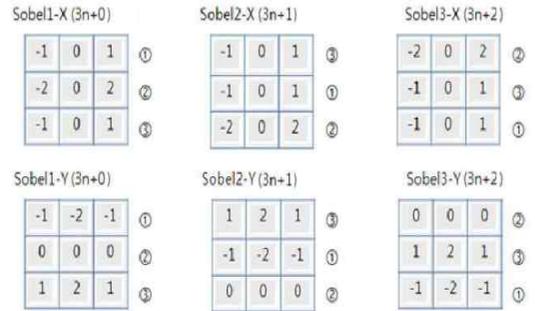


Fig. 7 3-Line buffer Sobel mask Pattern
그림 7. 3-line buffer 소벨 마스크 패턴



Fig. 8 Date input pattern for 3-Line buffer
그림 8. 3-Line buffer으로 데이터 입력 패턴

1->2->3->1->.. 순으로 한 줄씩 Line buffer에 저장되기 때문에 영상 정보를 읽어올 때에 Sobel mask pattern이 바뀌게 되고 3가지 종류의 순서에 맞게 변경되어야 한다.

그림 7에서는 그림 8처럼 영상의 정보가 바뀔 때마다 Vertical mask와 Horizontal mask pattern이 변경되게 되고 3n+0, 3n+1, 3n+2번째로 Sobel mask 가 Line별 실행이 끝날 때마다 변경되게 된다. 따라 Line buffer에 영상 정보를 저장하고 Sobel mask filter에서 영상 정보를 동기화 되어 수행된다. 그러므로 Line buffer의 정보를 읽기, 쓰기가 별도로 동작하기 때문에 실시간으로 동기화된 Sobel edge 추출이 가능하다.

4. 3-Line buffer 방법

그림 9는 설계된 Sobel mask 순차회로의 상태도이다. B, C, D와 같이 각 Line의 영상 정보를 저장하는 것에 대한 상태부터 시작하여 D 상태에서 E 상태로 넘어갈 때에 주소가 세 번째 이상 입력이 되게 되면 영상 입력에 맞추어 Sobel mask filter가 동작하게 된다. 다음으로 G, H I와 같이 3-Line buffer Sobel mask pattern이 동작하게 되었으므로 Line buffer의 동작 순서는 B, C, D 상태 별 입력과 G, H, I 상태 출력이 동시에 실행된다. 각 상태에 대한 동작은 표 1과 같다.[9]

Table 1. Lists of States

표 1. 상태에 따른 리스트

No.	States	Remark
1	A	Idle
2	B	Write A_Line Address & Data
3	C	Write B_Line Address & Data
4	D	Write C_Line Address & Data
5	E	if(C_Line Address>3) Masking Start
		else if(C_Line_full) B State Check
6	F	Masking mode check, G, H, I Sobel calculation
7	G	Sobel Pattern calculation of 3N+ 0
8	H	Sobel Pattern calculation of 3N+ 1
9	I	Sobel Pattern calculation of 3N+ 2
10	J	Sobel X, Y calculation
11	K	Sobel Data absolute

IV. 실험

본 논문에서 제안한 실시간 Sobel edge 추출에 대한 하드웨어 영역의 동작 속도와 합성결과는 Virtex II xc2vp-30-7-FF896 FPGA 기반의 xilinx 14.2를 사용하여 검증되었다. 또한 영상에서 추출된 Sobel edge 정보의 손실과 영상 차이의 성능을 검증하기 위해서 Rajesh가 제안한 방법들과 PSNR 결과를 비교하였다.

표 2는 Rajesh가 제안한 Multiplier 방법으로 메모리가 감소하도록 설계된 하드웨어와 제안하는 3-Line buffer 방법으로 설계한 하드웨어 결과를 비교하였다. 합성 결과는 기존 방법보다 제안한 F/F(Flip-flop)는 34%, 4-input LUTs는 80% 감소되었다. Rajesh가 제안한 일부분이 직렬화 구조로 Multiplier 설계된 메모리 설계 방식보다 일부 감소되었음을 확인 가능하다.

표 3은 Rajesh와 제안하는 방법으로 하드웨어 설계된 합성 결과의 동작 속도이다. 최소값 주기가 기존 방법에서는 6.751ns 이었을 때, 3-Line buffer를 적용한 방법에서는 4.190ns로써 62% 감소됨을 확인하였다. Data pass delay가 감소되었기 때문에 전체적인 동작 구간 차이도 성능이 증가되었으므로 확인하였다.

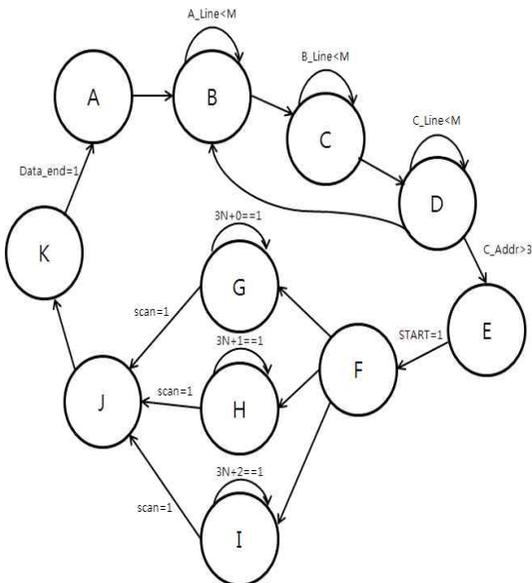


Fig. 9 Finite State Machine of 3-Line buffer Sobel
그림 9. 3-Line buffer Sobel의 상태도

Table 2. Synthesis Result of the previous methods and our proposed method

표 2. 기존 방법과 제안하는 방법의 합성결과

Resources		Used	Available	Utilization
Number of Slice Flip Flops	Rajesh	482	27392	1%
	Proposed	167		1%
Number of occupied Slices	Rajesh	371	13696	2%
	Proposed	369		2%
Total Number of 4 input LUTs	Rajesh	353	27392	1%
	Proposed	283		1%
Number of bonded IOBs	Rajesh	14	556	2%
	Proposed	20		3%
Number of BUFGM UXs	Rajesh	1	16	6%
	Proposed	1		6%

Table 3. Speed Performance of the previous methods and our proposed method

표 3. 기존 방법과 제안하는 방법의 속도 성능

Parameter		Value
Minimum Period	Rajesh	6.751ns
	Proposed	4.190ns
Maximum Frequency	Rajesh	148.133MHz
	Proposed	238.687MHz

그림 10은 제안된 3-Line buffer의 시뮬레이션 결과이다. 위의 그림은 전체적인 시뮬레이션 흐름이며 A부분은 입력되어지는 영상 pixel 값과 Line buffer로 출력되어지는 정보 주소를 의미하며, 영상 정보 값이 표현된 영상 정보 제어

Table 4. PSNR results of the previous methods and our proposed method

표 4. 기존 방법과 제안하는 방법의 PSNR 결과

Image	Size	Rajesh [dB]	propose d [dB]
Baby	640x480	30.38	32.99
Doll	640x480	29.04	29.43
Lena	640x480	28.31	33.00
Tsukuba	640x480	31.86	31.98

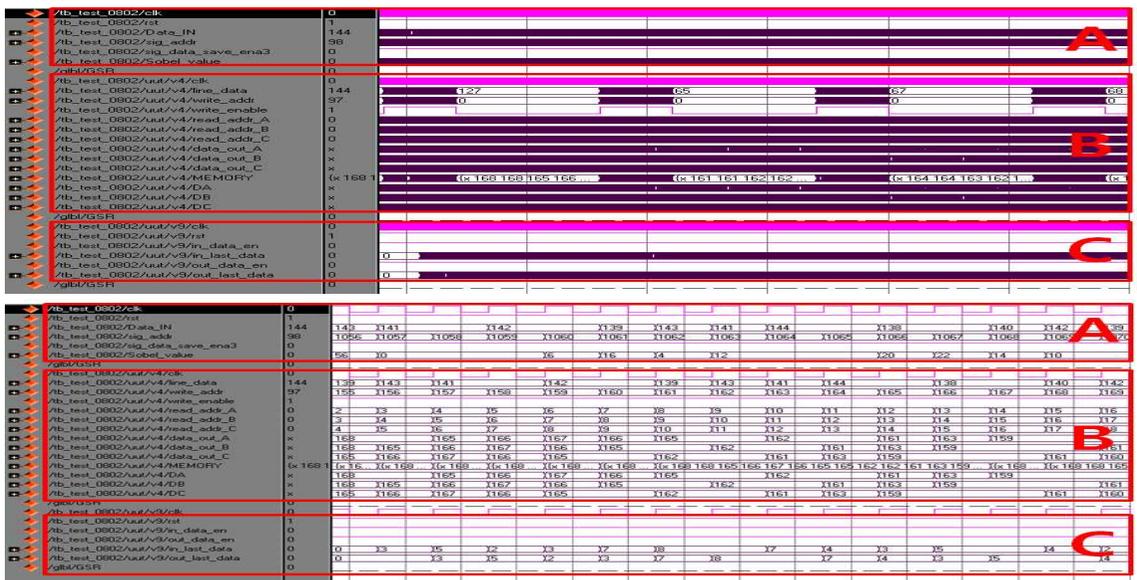
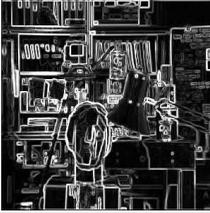


Fig. 10 3-Line buffer design timing chart

그림 10. 3-Line buffer 설계 타이밍도

Table 5. Comparison results of the previous methods and out proposed method

표 5. 기존 방법과 제안하는 방법의 PSNR 결과

	Reference	MATLAB (S/W)	Rajesh's Method (H/W)	Proposed Method (H/W)
Lena Image				
Tsukuba Image				

고 B부분은 3-Line buffer로 구성된 Line buffer 중 1개의 Line buffer의 시물레이션 결과이다. 마지막으로 C부분은 전체적으로 입력되어 지는 데이터와 출력되는 데이터를 나타낸다.

밑의 그림에서는 위의 그림을 확대한 것이다. 다음과 같이 A, B, C 모든 부분에서 입력과 출력이 동일한 CLK에 따라 동기화 되는 것을 확인 가능하다.

표 4은 Matlab을 활용하여 edge를 추출하는 함수로 출력된 Reference 이미지를 Rajesh가 제안한 Line Buffer의 전체 영상 이미지와 제안하는 3-Line buffer의 이미지를 각각 비교한 PSNR 결과이다. Lena이미지 같은 경우에 Line buffer의 전체 영상 이미지보다 16%정도 증가한 결과를 보였고 평균적으로는 6.5% 증가한 결과를 보였다. 제안하는 3-Line buffer 방법으로 추출한 이미지가 Rajesh와 비교하여 비슷한 성능을 보였다.

Rajesh이 제안한 방법과 본 논문에서 제안하는 3 Line buffer 방법의 Sobel edge 추출 결과는 Reference Image에 따라 edge 추출 전후의 예로 표 5에 제시하였다.

III 결론

본 논문에서는 실시간 영상을 처리하기 위한 기반으로 3-Line buffer를 이용한 하드웨어 보정 방법을 제안 하였다. 순차적으로 입력하는 영상 정보 제어를

설계하고 각 이미지의 Line별로 저장할 수 있는 Line buffer들을 크기에 맞추어 설계한 뒤 pixel line별로 지정된 mask pattern으로 Sobel 연산을 수행하여 윤곽선 데이터를 추출하는 Sobel mask filter을 설계하는 것을 기술하고 평가하였다. 또한 실시간으로 모든 Block들이 CLK 동기화 되어 수행됨에 있어 발생할 수 있는 CLK delay 현상을 방지하기 위하여 Sobel mask 순차회로 알고리즘의 상태도(FSM)를 적용하였다. 실험결과 이미지에서 추출된 Sobel edge 정보들을 합성, 연산 속도 및 PSNR 결과에서 비교하여 우수한 결과를 얻었다.

향후 연구 과제로는 실제로 고속 edge 추출을 이용한 회전, 이동 영상 보정 시스템의 알고리즘 및 하드웨어 설계를 연구 하고자 한다.

References

- [1] Yo-Sung Ho, "3DTV Stereo-scopc Image Information Processing," Dooyang Company, pp.9-17, 2010
- [2] Won-Ki Go, "Hardware Block Design of the Image Pre-processing Algorithm for Recognition of License Plates," 2013
- [3] Hi-Seok Kim, Young-Hwan Kim, Sea-Ho Kim, Choong-Mo Youn, "FPGA Implementation of Disparity Estimation Processing Architecture for Stereo Camera System," CENICS, 2013

- [4] Sung-Whan Chung, Moon-Ho Lee, "Visual C++ Digital Image Processing using Open source CxImage," Hongrung Publishing Company, pp.457-482, 2006
- [5] Rajesh Mehra, Rupinder Verma, "Area Efficient FPGA Implementation of Sobel Edge Detector for Image Processing Applications," International Journal of Computer Applications, Volume 56-No.16, 2012
- [6] Weili Kong, Pengfei Chang, Zhuo Bil, "REAL-TIME SOBEL EDGE DETECTOR," International Conference on Engineering and Technology, T.4-3.4, pp.1-4, 2013
- [7] X. Zhai, F. Bensaali, S. Ramalingam, "Real-Time License Plate Localisation on FPGA," Computer Society Conference on Computer Vision Pattern Recognition Workshops, pp.14-19, 2011
- [8] O. R. Vincent, O. Folorunso, A, "A Descriptive Algorithm for Sobel Image Edge Detection," Proceedings of Informing Science & IT Education Conference, pp.97-107, 2009
- [9] I.Yasri, N.H.Hamid, V.V.Yap, "Performance Analysis of FPGA Based Sobel Edge Detection Operator," 2008 International Conference on Electronic Design, 2008

HiSeok Kim (Member)



1977 : BS degree in Electronics Engineering, Hanyang University.
 1980 : MS degree in Electronics Engineering, Hanyang University.
 1985 : PhD degree in Electronics Engineering, Hanyang University.
 1987~current : Professor, Dept. of Electronics Engineering, Cheongju University

BIOGRAPHY

ChanSu Park (Student Member)



2013 : BS degree in Electronics Engineering, Cheongju University.
 2015 : MS degree in Electronics Engineering, Cheongju University.
 2015 : PhD course in Electronics Engineering, Cheongju University.