2000年度 大韓電子工學會 秋季綜合學術大會 論文集 第23卷 第2號 2000/11

윤곽선 검출에 바탕을 둔 움직임 추적 ASIC 구현에 관한 연구

김 희걸, 조 경순
한국외국어대학교 전자공학과
전화 : 031-330-4502 / 핸드폰 : 017-359-1190

A Study on the Implementation of the Motion Tracing ASIC Based on the Edge Detection

Huigirl Kim, Kyeongsoon Cho
Dept. of Electronic Engineering, Hankuk University of Foreign Studies
E-mail : girl@san.hufs.ac.kr

Abstract

This paper describes the algorithm, architecture and design of the circuit implementing motion tracing features based on the edge detection. The Sobel operation was used to compute the edges of moving objects. Motion tracing is performed by searching for the center of the edges for each frame and adding those centers. The edges and the centers of the moving object from camera were displayed in the monitor and verified using Xillinx FPGA.

I. 서론

멀티미디어 시대를 사는 지금 영상처리의 기술은 급속도로 발전해 가고 있다. 현재 영상처리의 응용 분야는 의료, 보안, 영화, 오락 산업 등에 이르기까지 우리 생활 전반에 걸쳐 있다.

영상처리 기술의 한 분야인 윤곽선 검출과 움직임 추적은 영상처리 응용 분야 중 보안시스템과 관련이 있다. 만약 감시용 카메라에서 물체의 움직임이 있을 경우 경보를 울리면, 사람이 모니터 앞을 지켜보아야 하는 일이 없어질 것이다. 또한 물체의 움직임이 있을 경우 그 시점부터 녹화를 한다면, 녹화할 자료의 양을 감소시키는 결과도 가져 올 수 있게 된다.

본 논문은 윤곽선 검출과 움직임 추적에 관한 연구로서 움직임 추적기의 구조 및 기능에 대하여 기술하고 있다. 본 논문에서 기술한 움직임 추적기는 Sobel 연산에 바탕을 두어 움직이는 물체에 대한 윤곽선을 계산하고, 그 중심값을 매 프레임마다 찾아냄으로써 움직임 추적 기능을 구현하였다. Verilog-HDL을 사용하여 회로 설계를 하였으며, 삼성전자㈜ 사의 0.5㎛ Standard Cell Library[1]을 바탕으로 Synopsys 사의 Design Compiler를 사용하여 논리 수준 회로를 합성하였고, Cadence 사의 Verilog-XL을 사용하여 검증하였다. 또한 그 결과를 눈으로 확인하기 위하여 Xillinx FPGA와 카메라, 모니터를 이용, PCB를 제작하여 실제 움직임 검출 결과를 모니터에 출력함으로써 그 결과를 확인하였다.

II. 윤관선 검출과 움직임 추적 이론

2.1 Sobel 연산

[1] 이 논문은 ECT (Enhanced Chip Technology) 사의 지원에 의해 수행된 연구 결과임.

윤곽선(edge)[2]이란 이미지의 밝기 혹은 칼라의 값이 크게 바뀌는 화소(pixel)들을 말한다. 정지 영상의 경우 윤곽선 검출은 물체들 사이의 공간적 구별을 나타낸다. 하지만, 동영상의 경우는 하나의 이미지 데이터를 비교하는 것이 아니라 현재와 과거의 이미지 차이를 비교한다. 따라서 물체의 시간에 따른 변화를 나타낸다. 여기서 이미지의 차이를 더 현저하게 나타내기 위한 작업이 필요하게 된다. Sobel 연산[3]은 위와 같은 작업을 목적으로 사용된다. Sobel 연산은 주어진 mask 행렬의 크기만큼 이미지의 값과 곱셈 연산을 한 후 그 값을 모두 더한다. 이렇게 되면, mask 행렬과 곱셈한 이미지 행렬의 중앙 화소에 대한 Sobel 연산을 실행한 것이 된다. 그림1에서는 mask 행렬이 {-1, 2, -1} 인 Sobel 연산의 예를 나타내었다.

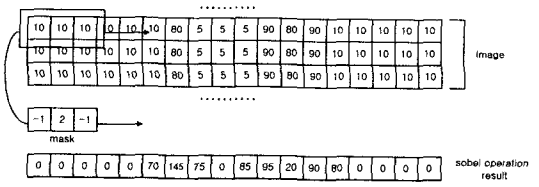

그림 1. Sobel 연산의 예

2.2 윤곽선 검출과 움직임 추적

윤곽선 검출은 Gradient의 크기와 방향을 판단하여 얻어진다. Gradient의 크기는 변화량을 의미하며 Sobel 연산을 통하여 얻어진다. Sobel 연산 결과 즉 그림1에서의 마지막 행의 값이 Gradient의 크기가 된다. Sobel 연산에서 Gradient의 방향(θ)은 $\tan^{-1}(G_y/G_x)$ 이다. 여기서 G_x는 Sobel mask의 수평방향의 Gradient의 값이고, G_y는 Sobel mask의 수직방향의 Gradient의 값이다.

표 1. Gradient의 영역 분류

	1사분면 X>0, Y>0	2사분면 X<0, Y>0	3사분면 X<0, Y>0	4사분면 X>0, Y<0
영역1	$0<\theta<0.414$	$0<\theta<0.414$	$0<\theta<0.414$	$0<\theta<0.414$
영역2	$0.414<\theta<2.414$	-	$0.414<\theta<2.414$	-
영역3	$\theta>2.414$	$\theta>2.414$	$\theta>2.414$	$\theta>2.414$
영역4	-	$0.414<\theta<2.414$	-	$0.414<\theta<2.414$

Gradient의 방향을 계산하기 위해서는 삼각함수 계산을 하여야 하지만, 계산량이 많은 삼각함수 연산 대신 표 1에서와 같이 영역을 분류함으로 하여 삼각함수의 연산을 곱셈과 비교 연산으로 바꾸었다. 그림 2는 표 1를 알기 쉽게 그림으로 나타낸 것이다. 예를 들어 그림 2에서 G_x, G_y가 양수 일 경우 tan67.5 = 2.414이다. Gradient의 방향 성분이 영역3의 값을 가지려면 $(G_y/G_x) > \tan 67.5$ 이여야 한다. 따라서 $(G_y/G_x) > 2.414$ 이고, $1024G_y > 2472G_x$ 이다.

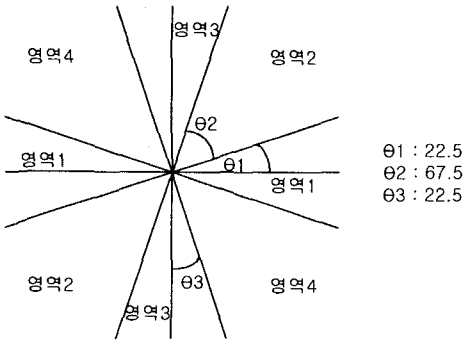

그림 2 Gradient의 영역에 대한 분류

여기서 양변에 1024의 수는 2의 승수이므로 곱셈연산을 시프트 연산으로 바꿀 수 있다. 또한 1024를 곱함으로 해서 소수점의 연산을 정수 연산으로 바꾸었다.

Sobel 연산에서 얻어진 Gradient의 크기가 임계값 이상이 되고, 그에 따른 Gradient의 방향에 수직 영역의 Gradient 크기와 다시 비교하여 그 값이 크면 윤곽선으로 인식이 된다. 여기서 임계값은 실험을 통하여 얻는 값이다. 임계값이 작으면, 이미지 변화에 민감한 반면 잡음이 많이 생기며, 임계값이 크면, 그 반대의 상태가 된다.

움직임 추적[4][5]은 윤곽선 자료를 바탕으로 구하여진다. 움직임 추적은 전체 이미지에 대하여 윤곽선들에 대한 행의 누적좌표 값, 열의 누적좌표 값, 윤곽선의 개수를 구하는 것으로부터 시작한다. 행의 누적좌표 값과 열의 누적좌표 값을 윤곽선의 개수로 각각 나눈다. 이렇게 되면, 행과 열 좌표에 대한 평균값이 된다. 또한 이 좌표의 평균값은 윤곽선으로 나타나는 물체의 중심 좌표이다. 정지 영상이 아닌 동영상이기 때문에 윤곽선의 중심의 좌표를 연속된 프레임마다 계산하여 화면에 표시한다. 화면에서 물체가 움직이게 되는 것에 따라서 그에 대한 중심 함께 움직이기 때문에 물체의 이동을 추적할 수 있게 된다. 또한 화면을 여러개의 영역으로 구분한 다음 중심 값이 어느 영역에 속하는지를 판단하여 해당 영역에 대한 중심 값을 출력함으로 해서 물체의 미세한 움직임에는 둔감하게 처

리하였다. 본 논문의 실험에서는 화면 전체를 240(16×15)개의 영역으로 나누었다.

III. 움직임 추적기 회로 구현

움직임 추적기는 크게 영상 입력 제어기, 영상 출력 제어기, 움직임 추적기, SDRAM 제어기로 구성되어 있으며, 카메라에서 영상을 입력받아서 모니터에 움직임 추적의 결과를 보여주도록 설계하였다.

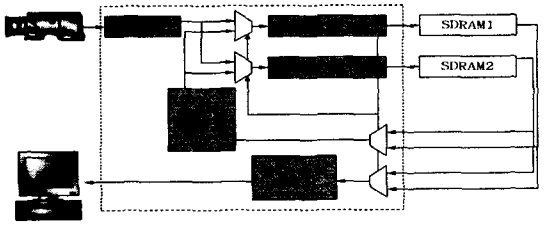

그림 3. 움직임 추적기의 전체 블록 다이어그램

3.1 SDRAM 제어기

윤곽선 검출과 움직임 추적에 필요한 이미지나 중간 계산 결과 또는 최종 결과를 저장하기 위한 자료 저장 장소가 필요하다. 칩 내에 SRAM과 같은 블록들을 사용하는 것이 가능하지만 칩의 크기가 너무 커지게 된다. 이 문제를 해결하기 위해 본 논문에서는 외부의 SDRAM[6]을 사용하는 방법을 채택하였다. 여기에 사용된 DRAM 모델은 KM416S1120D라는 512K x 16bit x 2 Banks Synchronous DRAM이다.

SDRAM제어기는 SDRAM의 접근을 용이하게 하기위해서 설계되었다. SDRAM제어기의 다른 기능은 데이터의 흐름 제어이다. 그림 3에서 영상 입력 제어기에서 들어오는 영상 데이터를 SDRAM1과 SDRAM2에 저장하게 된다. 이는 하나의 SDRAM에 자료의 입출력을 동시에 할 수 없기 때문이다. 카메라에서 영상입력 제어기를 거쳐 들어오는 데이터는 끊김없이 연속적으로 들어 오기 때문이다. 이런 이유로 움직임 추적기에서 SDRAM1에 데이터를 저장하고 있는 동안 영상입력 제어기에서 들어 오는 데이터는 SDRAM1에는 저장 할수 없게 되고 SDRAM2를 사용하여야 한다. 이와 마찬가지로 움직임 추적기에서 SDRAM1에 데이터를 읽고 있는 동안 영상출력 제어기에는 SDRAM1의 데이터에는 접근 할 수 없으므로 SDRAM2의 데이터를 사용하게 된다.

3.2 영상 입력 제어기

영상입력 제어기는 카메라에서 들어오는 영상 자료를 SDRAM에 저장하는 역할을 한다. 움직임 추적기의 계산량을 줄이기 위해서 256 × 240 개의 영상 데이터만을 저장하게 된다. 인접한 영상데이터의 값은 거의 변화가 없기 때문에 홀수 혹은 짝수 번째의 영상 데이터만을 가지고 계산을 하여도 결과에는 크게 영향을 주지 않는다. 움직임 추적기의 계산은 현재와 과거의 영상 데이터를 필요로 함으로 연속된 2장의 이미지를 저장하게 된다. 또한 움직임 추적기의 계산 시간에 비해 영상데이터를 저장하는 시간이 더 적으므로 카메라에서 들어오는 홀수 필드의 데이터와 짝수 필드의 데이터 중에서 홀수 필드의 데이터만을 저장하게 된다.

3.3 영상 출력 제어기

영상출력 제어기는 SDRAM에 저장되어 있는 움직임 추적 데이터를 읽어 들여 모니터에 출력하는 역할을 한다. 카메라에서 입력 되는 데이터는 흑백 이미지 데이터이며 8bit이다. 움직임 추적 결과 데이터는 흑(0)과 백(255) 두 가지의 값으로 표시된다. 그러므로 SDRAM에 저장되는 움직임 추적 결과는 저장공간과 저장시간을 줄이기 위해 1bit의 값 '1'과 '0'으로 표시하였다. 따라서 Burst Length=8 모드로 SDRAM에 움직임 추적 결과를 저장 할 때 한번에 128bit의 자료를 보낼 수 있으므로, 128개의 윤곽선 자료를 저장하게 된다. 영상 출력 제어기는 SDRAM에서 한번에 128개의 윤곽선 자료를 읽어 들여 1bit씩 그 값을 검사한다. 검사한 값에 따라 '0'의 값은 8bit "10000000", '1'의 값은 "11111111"로 출력하게 된다.

3.4 움직임 추적기

움직임 추적기는 총10개의 블록으로 구성되어 있다. 움직임 추적기는 SDRAM에 저장된 이미지 데이터를 읽어 들여 Gradient의 크기와 방향을 계산하고 그 결과 값을 다시 SDRAM에 저장한다. 그 후 다시 SDRAM에서 Gradient의 값을 읽어 들여 윤곽선의 판단과 윤곽선의 좌표, 윤곽선의 개수를 계산하여 그 결과값을 다시 SDRAM에 저장한다. 마지막으로 계산된 윤곽선의 좌표와 개수로 윤곽선의 중심을 계산하고 중심의 영역을 판단한다. 중심을 모니터에 나타내기 위해서 5×5개의 흰색 화소로 구성된 사각형 이미지를 사용하였다. 움직임 추적기의 내부 블록은 구동 제어부, 입력 버퍼, 윤곽선 검출 연산부, 출력 버퍼, 움직임 추적 연산부, 나눗셈 연산부, 메인 제어부, 주소 제어기, 쓰기 주소 제어기, Refresh 제어부로 구성 된다. 그

림 4는 움직임 추적기를 합성한 결과이다. 계층적으로 합성한 결과이기 때문에 하위 블록들은 사각형으로 표시 되었다.

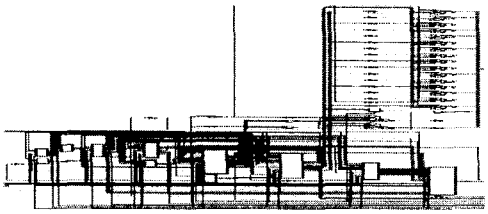

그림 4 움직임 추적기

Ⅳ. 실험 결과

본 논문에 실험에서 윤곽선 검출과 움직임 추적기 구조를 Verilog-HDL로 구현하였다. HDL로 기술한 회로는 삼성전자㈜ 사의 0.5㎛ Standard Cell Library를 바탕으로 Synopsys 사의 Design Compiler를 사용하여 논리 수준 회로로 합성하였고, Cadence 사의 Verilog-XL을 사용하여 검증하였다. 합성된 회로는 움직임 추적기, 영상 입력 제어부, 영상 출력 제어부와 기타 회로를 합하여 총 27,495개의 게이트로 구성었으며 전체 회로의 최대 지연 시간은 57.28ns이었다. 실험에서 회로 검증 방법은 동일한 입력을 C프로그램을 실행하여 얻은 결과와 회로에서의 결과를 비교하는 방식으로 검증하였다. Maximum, typical, minimum 지연 시간에 대한 게이트 레벨 simulation을 통하여 수행하였다. 또한 결과를 모니터에 직접 나타내게 하기 위하여 Xilinx FPGA와 카메라, 모니터, ADC, DAC 등의 회로를 PCB에 구현하여 그 결과를 모니터로 확인하였다. Xilinx FPGA을 이용한 PCB에서는 256 x 240 크기의 흑백 영상을 10MHz 클럭을 사용하여 카메라의 입력으로부터 모니터에 출력까지의 데이터를 처리하는데 66.1ms 시간이 소요되었다. 과거와 현재 두 장의 이미지를 가지고 계산하게 되므로 1초에 총 30 프레임의 이미지를 받아들여 모두 계산이 끝나면, 15프레임의 결과가 남게 된다. 표 2는 합성된 게이트 수를 모듈별로 나타낸 것이다.

표 2. 모듈별 게이트 수

블록 이름	합성한 게이트 수
움직임 추적	20675
SDRAM 제어기	430
영상 입력 제어기	2099
영상 출력 제어기	1478
기타 회로	2813
합 계	27495

Ⅴ. 결론

본 논문에서는 윤곽선 검출에 바탕을 둔 움직임 추적 ASIC구현에 관한 연구에 대해 기술하였다. Sobel 연산에 바탕을 두어 움직이는 물체에 대한 윤곽선을 계산하고, 그 중심값을 매 프레임마다 찾아냄으로써 움직임 추적 기능을 구현하였다. PCB에서 결과를 모니터로 출력하기 위해서 10MHz 클럭을 사용하였고, 두 프레임의 이미지를 받아들여 윤곽선과 움직임을 추적하는데 66.1ms 시간이 소요되었다.

움직임 추적기 회로는 총 27,495개의 게이트로 합성되었다. 움직임 추적기의 내부 회로 중 입력 버퍼와 이미지 버퍼의 크기가 전체 칩 크기의 40%를 차지하였으며, 향후 개선할 계획이다. 또한 움직이는 물체를 추적하는 기능을 갖는 팬/틸트 카메라에 본 논문에서 기술한 칩을 적용하는 작업을 진행 중이다.

참고문헌

[1] Samsung Electronics Co., Ltd., 0.5㎛ Hihg Density CMOS Standard Cell Library Data Book, 1997.

[2] 이진성, 이훈철, 김성대, "적외선 영상에서 움직임 영역 검출을 이용한 목표물 검출 및 추적 기법" http://sdvision.kaist.ac.kr/~jslee/research/kspc99/kspc_ppt /sld003.htm

[3] Ramesh Jain, Rangachar Kasturi, Brian G. Schunck, "Machine Vision", McGraw-Hill, pp.147, 1995.

[4] Li RX, Zeng B, Liou ML, "Reliable motion detection/compensation for interlaced sequences and its", IEEE Transactions on Circuits & Systems for Video Technology, V.10 N.1, 23-29, 2000.

[5] Morimoto CH, Koons D, Amir A, Flickner M, "Pupil detection and tracking using multiple light sources", Image & Vision Computing, V.18 N.4, 331-335, 2000.

[6] Samsung Electronics, KM416S1120D:1M x 16 SDRAM, Revision 1.4, June 1999.