Journal of Korea Multimedia Society Vol. 15, No. 10, October 2012(pp. 1247-1256)

파이프라인 구조의 얼굴 검출 하드웨어 설계 및 검증

김신호†, 정용진††

요 약

필터를 기반으로 하는 영상 처리 알고리즘은 많은 연산과 메모리 접근으로 인해 임베디드 환경에서의 실시간 동작이 어렵다. 본 논문에서는 필터 기반의 얼굴 검출 하드웨어 엔진을 임베디드 환경에서 실시간으로 동작시키기 위해 파이프라인 구조로 설계하고 검증하였다. 얼굴 검출 알고리즘은 입력으로 들어온 영상에서 학습된 얼굴의 특징 데이터를 이용하여 얼굴의 위치를 찾는 연산을 수행한다. 이를 하드웨어로 구현하기 위해 알고리즘의 연산을 파악하여 중복되는 연산을 병렬 처리하고 라인 메모리를 이용하여 메모리 접근을 최소화하여, 이것을 기반으로 파이프라인 구조의 하드웨어를 설계하였다. 하드웨어 구조는 Resize, ICT(Improved Census Transform), Find Candidate 등의 3 단계로 나뉘어져 있으며, 총 507KByte의 내부 SRAM을 사용하였다. ARM Cortex A8 프로세서와 Xilinx사의 Virtex5LX330을 이용하여 검증한 결과 9,039 LUTs를 사용하였고 최대 동작 클록은 165MHz로, VGA(640×480) 해상도에서 108 frame/sec의 동작속도로 최대 20명까지 검출이 가능한 것을 확인하였다.

Design and Verification of Pipelined Face Detection Hardware

Shin-Ho Kim†, Yong-Jin Jeong††

ABSTRACT

There are many filter based image processing algorithms and they usually require a huge amount of computations and memory accesses making it hard to attain a real-time performance, expecially in embedded applications. In this paper, we propose a pipelined hardware structure of the filter based face detection algorithm to show that the real time performance can be achieved by hardware design. In our design, the whole computation is divided into three pipeline stages: resizing the image (Resize), Transforming the image (ICT), and finding candidate area (Find Candidate). Each stage is optimized by considering the parallelism of the computation to reduce the number of cycles and utilizing the line memory to minimize the memory accesses. The resulting hardware uses 507 KB internal SRAM and occupies 9,039 LUTs when synthesized and configured on Xilinx Virtex5LX330 FPGA. It can operate at maximum 165MHz clock, giving the performance of 108 frame/sec, while detecting up to 20 faces.

Key words: Face Detection(얼굴 검출), Pipeline(파이프라인), ICT, FPGA, Embedded System(임베디드 시스템)

※ 교신저자(Corresponding Author) : 김신호, 주소 : 서울특별시 노원구 월계1동 광운대학교 참빛관 810호 실시간구조 연구실(139-701), 전화 : 02)940-5551, FAX : 02)942-5517, E-mail : rough83@kw.ac.kr
접수일 : 2012년 5월14일, 수정일 : 2012년 7월7일
완료일 : 2012년 8월 6일
† 정회원, 광운대학교 전자통신공학과
†† 정회원, 광운대학교 전자통신공학과
(E-mail : yjjeong@kw.ac.kr)
※ 본 연구는 2012년 정부(교육과학기술부)의 재원으로 한국연구재단(NRF-2010-0014557)과 지식 경제부가 지원하는 산업융합원천기술개발사업 (10039188, 스마트 자동차용 프로그래머블 융복합 멀티미디어 SoC 플랫폼 개발)을 통해 개발된 결과임을 밝힙니다.

1. 서 론

얼굴 검출은 얼굴 인식과 함께 얼굴 인식의 주요 과정으로써 입력된 영상에서 얼굴을 검출해 내는 기능을 한다. 얼굴 검출에서 검출된 얼굴의 정확도에 따라 얼굴 인식의 결과에 큰 영향을 끼치기 때문에 얼굴 검출은 얼굴 인식 시스템에서 중요한 부분을 차지한다.

이러한 얼굴 검출 알고리즘은 크게 지식 기반 방법[1], 특징 기반 방법[2], 템플릿 매칭 방법[3], 학습된 필터를 이용하는 외형 기반 방법[4]으로 나누어진다[5]. 학습된 필터를 이용한 얼굴 검출은 효율적인 얼굴 검출을 위해 입력 영상의 전처리 및 변환을 통하여 다양한 필터를 순차적으로 적용하는 구조를 가지고 있다. 이런 구조상 메모리 접근 횟수와 복잡한 연산이 많기 때문에 실시간 동작에 어려움이 있다.

본 논문에서는 3단계의 파이프라인 구조를 이용한 얼굴 검출 하드웨어를 설계 및 검증하고 기존의 얼굴 검출 하드웨어와 비교 분석 하였다. 논문의 구성은 다음과 같다. Ⅱ장에서는 사용된 얼굴 검출 알고리즘에 대해서 설명하고, Ⅲ장에서는 구현한 하드웨어 구조를 알아본다. Ⅳ장에서는 제안한 하드웨어의 성능을 비교 분석하여 결과를 알아보고 Ⅴ장에서는 결론 및 향후 연구 방향에 대하여 논한다.

2. 알고리즘

얼굴을 검출하기 위한 방법에는 다양한 것들이 있지만, 본 논문에서는 미리 학습된 얼굴 검출 필터를 사용하는 외형 기반 방식의 알고리즘을 사용하였다. 이러한 외형 기반 방식에서 얼굴을 검출해 내기 위한 필터를 학습 시키는 것은 가장 중요하지만, 실시간으로 이루어질 필요가 없다. 또한 얼굴 검출 알고리즘은 입력된 영상의 밝기 변화에 따라 픽셀 값이 변화되기 때문에 검출 성능의 한계를 보이고 있다. 따라서 여러 가지 조명 상황에 따라 효율적으로 얼굴을 검출하기 위해서는 입력 영상의 밝기 변화에도 검출 성능에 영향이 없는 구조 정보만을 포함한 영상으로의 변환이 필요하다. 본 논문에서는 Bernhard Fröva가 제안한 MCT(Modified Census Transform)[6][7]을 하드웨어 설계에 적합하게 수정한 ICT(Improved Census Transform)[8]를 이용하여 얼굴 검출 필터를 제작하고 이것을 이용한 얼굴 검출 하드웨어를 구현하였다.

얼굴 검출 알고리즘의 흐름은 그림 1과 같다. 전체 흐름은 VGA(640×480) 크기의 그레이 스케일 이미지를 입력 받은 후, 얼굴을 검출하기 위한 필터 정보(Filter set)와 어떤 비율로 영상의 크기를 축소(Resize set) 시킬지 정한다. 영상 크기 변환 과정은 영상에 존재하는 다양한 크기의 얼굴을 검출하기 위해 입력되는 영상을 크기 변환 비율(Ratio)에 따라 피라미드 구조로 축소(Resize)시킨다.

축소된 영상에 대하여 밝기 변화에 강인하도록 구조 정보로만 영상이 구성되도록 ICT 변환을 한 뒤, 변환된 영상에 대하여 얼굴의 후보 영역 검출(Find Candidate) 연산을 거친다. ICT 변환은 원본 이미지 픽셀을 주변 픽셀과의 관계값으로 변환하기 때문에 기변화에 안정 적인 특징이 있다. 즉, 밝기 변화에 의해 절대적인 픽셀 값은 변화되지만 크고 작은 상관관계만은 변하지 않으므로 밝기 변화에 안정적인 영상을 얻을 수 있다.

얼굴 후보 영역 검출 연산은 8×8 크기의 얼굴 검출 필터를 적용하여 모든 얼굴 신뢰도 값을 더하여 그 합이 미리 정해진 임계값 이하일 경우에 얼굴의 후보

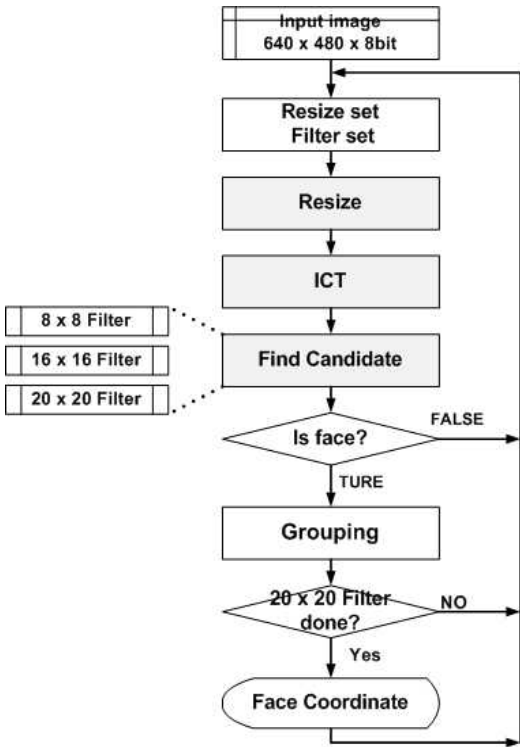


그림 1. 얼굴 검출 알고리즘 흐름도

영역으로 검출하게 된다. 얼굴 후보 영역이 검출되면 그룹화(Grouping)과정을 거쳐서 첫 번째 필터 연산의 얼굴 영역을 검출한다. 이렇게 검출된 영역에 대하여 20×20 크기의 얼굴 검출 필터를 적용 시킬 때까지 동일한 과정을 반복하여 최종 얼굴 영역을 검출하고 그 좌표를 결과로 출력시켜 준다. 그룹화 과정은 검출된 각각의 얼굴 후보 영역들의 중심 좌표와 거리를 비교하여 가까운 거리에 있고 크기가 비슷한 얼굴 후보 영역들을 하나의 얼굴로 만드는 과정을 거친다.

2.1 ICT (Improved Census Transform)

얼굴 검출에 있어 가장 큰 문제는 빛의 밝기 변화이다. MCT[7]는 이러한 빛의 밝기 변화에 강인하도록 영상을 구조 정보로만 갖도록 변환 하지만, 변환 후 9비트의 비트스트림(bit-stream) 결과이므로, 하드웨어로 구현할 시 메모리의 낭비가 심하다. 따라서 이것을 개선하여 하드웨어 구현에 적합한 형태로 만든 것이 ICT 변환[8]이며, 이 성능은 MCT와 유사하다고 확인된 바 있다.

$$\Gamma_i(x) = U_i \times 27 + D_i \times 9 + L_i \times 3 + R_i \qquad (1)$$

ICT 변환을 하게 되면 그림 2와 같이 3×3 픽셀 크기 영역 내에서 중심 픽셀(C)과 상(U), 하(D), 좌(L), 우(R)의 픽셀 값을 비교하여 변환한다. 변환 결과 중심점을 기준으로 좌, 우, 상, 하 네 개의 점이 0, 1, 2 세 개의 값을 가지므로, 식 (1)을 이용하여 각각의 위치에 대한 변환에 각각의 위치 정보를 유지하도록 가중치를 주어 7비트로 표현되도록 최종 변환을 하게 되면 총 81개의 특징이 나온다. 이 특징들은 원본 이미지에서 주위 화소와의 관계에 의한 결과로 얼굴 검출 필터의 얼굴 신뢰도 값의 인덱스

$$U_i = \begin{cases} 0 & \text{if } U-8 \geq C \\ 2 & \text{if } U+8 < C \\ 1 & else \end{cases} \qquad D_i = \begin{cases} 0 & \text{if } D-8 \geq C \\ 2 & \text{if } D+8 < C \\ 1 & else \end{cases}$$

$$R_i = \begin{cases} 0 & \text{if } R-8 \geq C \\ 2 & \text{if } R+8 < C \\ 1 & else \end{cases} \qquad L_i = \begin{cases} 0 & \text{if } L-8 \geq C \\ 2 & \text{if } L+8 < C \\ 1 & else \end{cases}$$

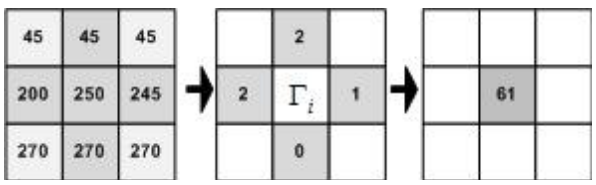


그림 2. ICT 변환의 예

역할을 하게 되며 이는 MCT와 비교하여 성능에는 영향을 주지 않고[8] 결과가 7비트의 비트스트림으로 나타낼 수 있으므로 하드웨어 구현 시 효율적인 메모리 사용이 가능하다.

2.2 얼굴 검출 필터 제작

본 논문에서 사용한 얼굴 검출 필터는 ICT변환을 한 영상에 대하여 Bernhard Fröva[7]가 제안한 총 4 단계의 캐스케이드 구조의 학습 알고리즘을 이용하여 만들어 졌다. 필터 학습 과정은 얼굴인 영상과 얼굴이 아닌 영상에 대하여 얼굴이 아니라고 판단되는 부분을 1 단계 검출 단계에서 제거하고, 다음 2, 3, 4 단계에서는 1 단계보다 상대적으로 변별력이 떨어지지만 얼굴 구분이 가능한 여러 가지 특징들을 선형으로 결합하여 검출 성능을 높이는 방법이다.

하지만 임베디드 환경에서 모든 캐스케이드 단계를 거친 필터를 사용하기에는 메모리를 너무 많이 차지하는 단점이 있다. 따라서 각 캐스케이드 단계별로 중복되는 특징 위치의 얼굴 신뢰도 값을 하나로 합쳐 캐스케이드 단계를 제거한 단일 구조의 필터를 제작하였다. 필터는 얼굴과 얼굴이 아닌 영상에 대하여 비교 될 수 있는 모든 특징 위치마다 얼굴을 분류할 수 있는 얼굴 신뢰도 값을 룩업 테이블 형태로 필터의 크기만큼 가지고 있다.

표 1. 얼굴 검출 필터 구조

X_1, Y_1 좌표	0000000	신뢰도 값	81 개
	0000001	신뢰도 값	
	…	신뢰도 값	
	1001111	신뢰도 값	
	1010000	신뢰도 값	
X_2, Y_1 좌표	0000000	신뢰도 값	81 개
	0000001	신뢰도 값	
	…	신뢰도 값	
	1001111	신뢰도 값	
	1010000	신뢰도 값	
…	…	…	81 개
…	…	…	81 개
$X_{filter\ size}$ $Y_{filter\ size}$ 좌표	0000000	신뢰도 값	81 개
	0000001	신뢰도 값	
		신뢰도 값	
	1001111	신뢰도 값	
	1010000	신뢰도 값	

3. 하드웨어 구조 설계

설계된 전체 하드웨어의 구조를 그림 3에 나타내었다. 3개의 파이프라인 스테이지로 구성되며 입력 영상과 필터 데이터(8×8, 16×16, 20×20), 각 연산 모듈의 결과값을 INT.SRAM 에 저장하였다. 또한 Filter set과 Resize set의 결과를 레지스터에 세팅해 놓아 메모리의 접근 시간을 감소 시켰다. Resize 모듈은 미리 정해진 일정 비율에 따라 원하는 픽셀의 좌표를 계산하여 그 부분의 픽셀만 가져와 영상을 축소하는 역할을 수행한다.

ICT 모듈은 축소된 영상에 대하여 ICT 변환을 수행하며 Find Candidate 모듈은 얼굴 검출 필터를 적용하여 입력된 영상에서 얼굴의 후보 영역을 찾는 역할을 한다. Grouping 모듈은 각각의 얼굴 후보 영역들을 하나의 최종 얼굴 영역으로 그룹화 시켜주고 얼굴 영역이 원본 이미지에서 어느 위치에 존재하는지 좌표를 결과로 출력시켜주는 역할을 한다. 알고리즘의 연산 분석 및 파이프라인이 적용된 모듈에 대한 것들을 아래에 설명한다.

3.1 라인 메모리를 이용한 파이프라인 구조

3.1.1 얼굴 검출 연산 분석

파이프라인 구조의 얼굴 검출 하드웨어를 구현하기 위해서는 알고리즘의 흐름 및 연산 분석이 핵심 요소이다. 표 2에서 볼 수 있듯이 알고리즘의 특성상 Filter Set과 Resize Set 연산을 각 필터의 크기에 대하여 한번만 수행하고, Grouping 연산은 검출된 얼굴 후보 영역의 개수에 따라 가변적으로 수행한다.

하지만 Resize, ICT, Find Candidate 연산은 영상의 축소 크기에 따라 동일한 횟수로 여러 번 수행한다. 각 필터별 영상 축소는 8×8 Filter를 적용할 때는 원본 영상을 일정 비율에 맞춰 축소시키고, 16×16, 20×20 Filter를 적용할 때는 8x8 Filter 연산을 통해 나온 얼굴 후보 영역들의 주변 영역을 같은 크기로 여러 번 축소시켜 좀 더 정확한 얼굴의 후보 영역을 찾게 된다.

표 2. 연산별 수행 횟수

연산	8×8 Filter	16×16 Filter	20×20 Filter
Filter Set	1 회		
Resize Set	1 회		
Resize	24 회 (160×120～12×9)	11 회 (22×22)	14 회 (26×26)
ICT	24 회	11 회	14 회
Find Candidate	24 회	11 회	14 회
Grouping	Number of Face Candidates		

또한 각 연산이 모두 메모리에 접근하기 때문에 파이프라인 구조를 사용하지 않을 시에는 수행 횟수에 비례하여 연산 시간이 늘어가게 된다. 따라서 병렬 연산이 가능한 파이프라인 구조를 갖는 하드웨어로 설계하여 연산 시간을 크게 감소시킬 수 있다.

3.1.2 라인 버퍼 메모리

본 논문에서 제안하는 파이프라인 처리가 가능한 얼굴 검출 하드웨어 구현을 위해서는 각 모듈의 버퍼와 필터에 필요한 메모리의 병렬 구성이 필요하다. 이때 사용되는 메모리 사용량을 줄이기 위하여 필요한 최소한의 메모리 사용량을 계산하고 연산을 분석하여 메모리 사용량을 최소화 시켰다.

각 연산은 영상에 대하여 정사각형 형태의 픽셀 영역이 슬라이딩 되면서 해당 모듈의 결과 값을 얻는 형태로 되어있다. 따라서 연산에 필요로 하는 픽셀 영역만큼 각 모듈의 결과 값을 저장하는 버퍼 메모리를 병렬 구조의 라인 메모리 형태로 구성하였다. ICT와 Find Candidate 연산을 하기 위해서는 축소된 영상의 가장 큰 크기인 160×120 크기의 메모리가 필요하다. 하지만 ICT 연산에는 단지 3×3 픽셀 크기의 영역만을 필요로 하다. 따라서 각 모듈과 시스템이 파이프라인 처리가 가능하다면 라인 메모리를 재활용하는 방법을 사용하여 메모리 사용량을 크게 줄일 수 있다.

그림 4에서 볼 수 있듯이 연산에 필요한 라인 메모리에 하나의 라인 메모리를 추가하여 연산에 필요한 데이터를 읽어 가는 동안 다음 연산의 결과를 추가된 라인 메모리에 쓰는 방식으로 모듈의 연산이 끝남과 동시에 다음 연산이 바로 시작 가능하도록 구성하였다.

이와 같이 라인 메모리를 재사용 하는 구성을 사용한 다면 Resize 메모리의 경우 160×120 크기의 메모리가 아닌 160×4 크기로 구성되고, ICT 결과 값을 저장하는 메모리 또한 가장 큰 필터인 20×20의 크기에 하나를 더하여 160×21로 메모리 사용을 최소화

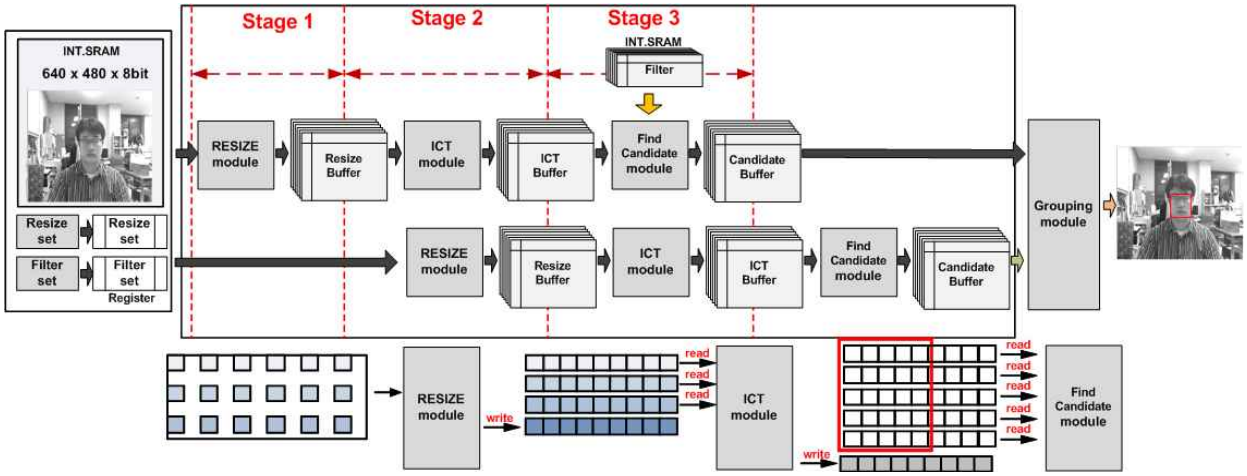


그림 3. 얼굴 검출 하드웨어 블록 다이어그램

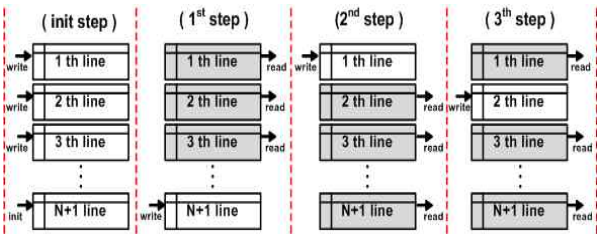


그림 4. 라인 버퍼 메모리 구성

표 3. 모듈별 처리 속도

모 듈	소요 시간
Resize	(3 clock + width size) × height size
ICT	(4 clock + width size) × height size
Find Candidate	(filter width size + width size) × height size
Grouping	1024 clock × face candidates

할 수 있다. 결과적으로 전체적으로 약 35 Kbyte의 내부 메모리 절약이 가능하고 라인 메모리의 사용으로 인하여 병렬처리가 가능하다.

3.1.3 라인 메모리를 이용한 파이프라인 구조

본 논문에서는 그림 3과 같이 3단계의 파이프라인 구조를 갖는 얼굴 검출 하드웨어를 제시하였다. 연산을 분석한 결과 Resize 모듈, ICT 모듈, Find Candidate 모듈은 영상이 축소되는 비율에 따라 동일한 수행 횟수를 가지므로 파이프라인 처리가 가능하다. 또한 각 모듈의 결과 값을 저장하는 버퍼 메모리는 라인 형태로 구성하여 병렬처리가 가능하게 하였고 하나의 라인 메모리를 더 두어 다음 연산에서 데이터를 읽어갈 때 이전 연산의 결과 값을 미리 쓸 수 있도록 구성하였다.

제안한 파이프라인 구조를 갖는 얼굴 검출 하드웨어 모듈별 처리 속도는 표 3과 같다. Grouping 모듈의 경우는 검출된 얼굴 후보 영역의 개수에 따라 처리속도가 틀려지지만, Resize, ICT, Find Candidate 모듈은 two-level pipeline구조로 처음 필요한 값들이 모두 채워진 후로는 매 클록마다 각각의 결과 값을 도출해 낼 수 있다.

또한 전체 동작을 3단계의 파이프라인 구조로 동작시키기 위해서는 처리속도가 가장 오래 걸리는 Find Candidate 모듈의 연산에 처리 속도를 맞추어야 한다. 따라서 Resize 연산과 ICT 연산의 결과 값을 저장해 놓기 위해 레지스터를 추가로 넣어 전체 시스템이 파이프라인 구조로 동작하게 설계하였다. 그림 5는 파이프라인이 적용된 모듈별 타이밍 분석도이다. 처음 160 × 120 사이즈로 영상을 축소할 때 총 20,160 ((160 + filter width) × 120) 클록이 소모되는 것을 확인할 수 있다. 축소된 영상의 가로 크기가 w, 세로 크기가 h일 때 파이프라인 타이밍 T는 식 (2)와 같다.

$$T=(filter\ width\ size+w)\times h \tag{2}$$

3.2 실시간 동작을 위한 설계

3.2.1 영상 축소(Resize) 모듈

Resize 모듈은 주어진 영상에 존재하는 다양한 크기의 얼굴을 검출하기 위해 그림 6과 같이 전체 영상의 크기를 줄여 나가는 피라미드 구조의 영상 축소

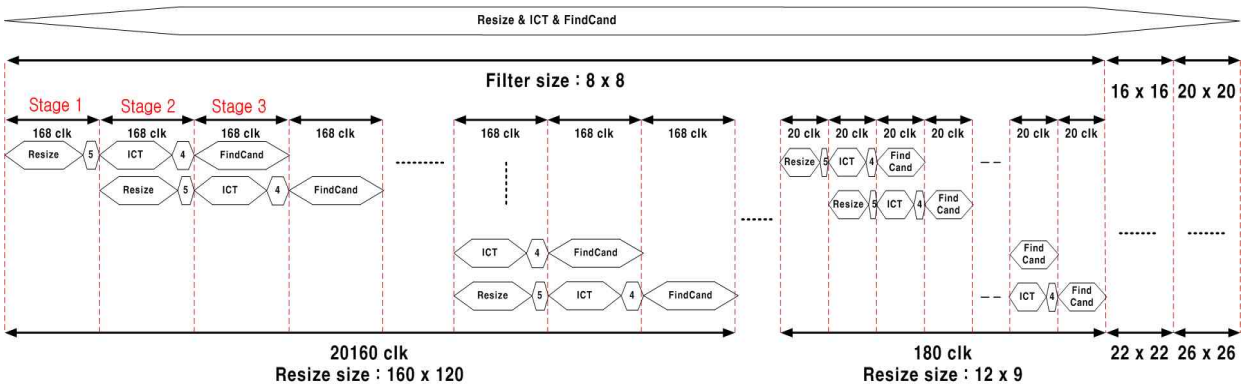


그림 5. 파이프라인을 적용된 모듈별 타이밍 분석도

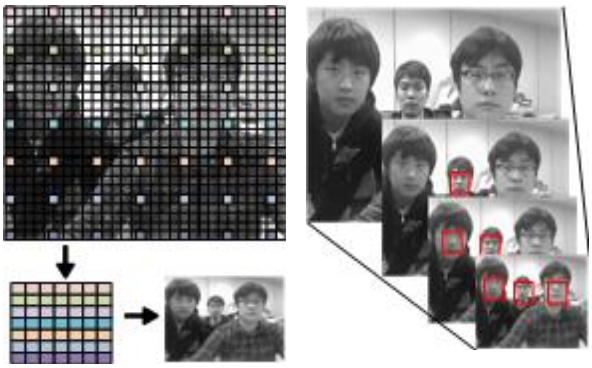


그림 6. 이미지 피라미드를 통한 얼굴 검출

방법을 사용한다. 영상의 크기를 줄이는 이유는 고정된 크기의 얼굴 검출 필터를 사용하여 다양한 크기의 얼굴을 찾기 위해 입력 영상을 일정 비율에 맞추어 축소시킨다.

축소시키는 방법은 입력된 영상의 좌표를 기준으로 일정한 비율에 따라 축소할 영상의 좌표를 계산하여 해당 좌표의 픽셀 값만을 읽어오고, 축소된 영상이 저장될 메모리의 어드레스를 0부터 축소할 영상의 크기만큼 순차적으로 증가시키며 SRAM에 저장하는 방법을 사용한다. 또한 그림 7에서 볼 수 있듯이 좌표 계산의 중간마다 레지스터를 삽입하여 연산이 동작할 동안 다음 좌표 계산의 중간 결과 값을 저장하여 처음 3클록 후 매 클록마다 결과를 뽑을 수 있도록 모듈 내부도 파이프라인 형태로 설계 하였다.

3.2.2 ICT 모듈

ICT 모듈은 축소된 영상을 구조 정보만 가진 영상으로 변환하는 연산을 한다. 3×3 픽셀 크기의 영역에서 중심 픽셀과 상, 하, 좌, 우 네 개의 십자 형태의 픽셀을 입력으로 받아 변환 결과는 7비트의 비트 스

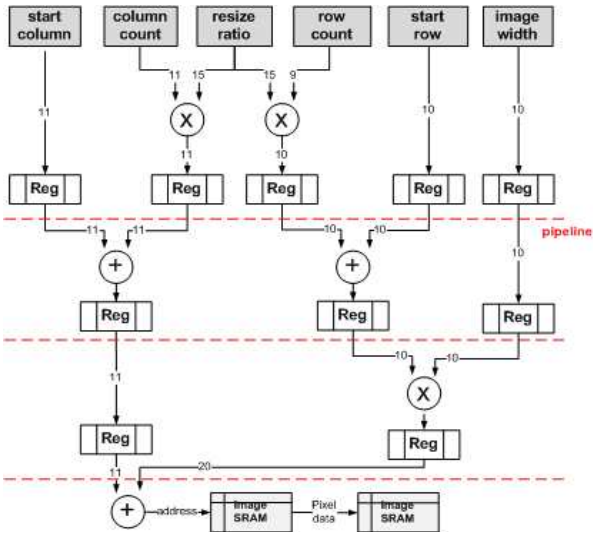


그림 7. 리사이즈 모듈 데이터 패스

트림으로 나온다.

먼저 가로 방향으로 한 픽셀 크기만큼 슬라이딩을 하면서 변환을 해 나가므로, 각 라인 별로 중간에 레지스터를 넣어 쉬프트 시켜주면서 진행하면 4개의 픽셀을 다음 연산에 재사용 할 수 있게 되어 메모리 접근 시간을 줄일 수 있다. 또한 ICT 연산의 중간 값을 미리 MUX를 통해 설정하여 곱셈기의 사용을 줄이는 방법을 사용하였다. 이와 같이 레지스터를 사용하여 재사용 할 수 있는 픽셀을 최대한 활용하고 연산기의 개수를 줄여 처음 4클록 후 매 클록마다 결과 값을 뽑을 수 있게 모듈 내부도 파이프라인 구조로 설계 하였고 각 단계를 점선으로 표시하였다.

3.2.3 얼굴 후보 영역 검출(Find Candidate) 모듈

Find Candidate 모듈은 ICT 변환 된 영상에 각각의 얼굴 검출 필터 데이터를 SRAM에서 불러와 적용

하여 얼굴 후보 영역을 찾는 모듈이다. 얼굴을 찾는 방식은 ICT 메모리의 데이터가 Filter 메모리의 어드레스 역할을 하고 해당 어드레스의 얼굴 신뢰도 값을 필터 크기만큼 합하여 미리 정해진 임계값보다 낮을 경우에 얼굴의 후보 영역으로 인식하게 된다. 알고리즘의 특성상 ICT 메모리와 Filter 메모리 양쪽에 접근하고 내부 연산기가 가장 많은 모듈로써 가장 긴 수행시간을 소모한다. 따라서 효율적인 연산을 위하여 그림 8과 같이 각 얼굴 검출 필터(8×8, 16×16, 20×20)를 라인 단위로 SRAM에 저장하고 각 연산도 크기에 맞춰 병렬처리 한다.

또한 영상에 대하여 필터를 슬라이딩 하는 방식을 사용하기 때문에 각 연산 중간에 레지스터를 삽입하여 매 클록마다 다음 연산이 미리 이루어지도록 모듈 내부도 파이프라인 구조로 설계 하였다. Confidence generator 내부는 각각의 얼굴 신뢰도 값들이 행 단위로 합쳐지고 그 결과들을 필터의 열 크기만큼 다음 행 단위 결과 값들과 합쳐져 최종적으로 필터 크기만큼의 얼굴의 신뢰도 값을 더하고 미리 정해진 임계값

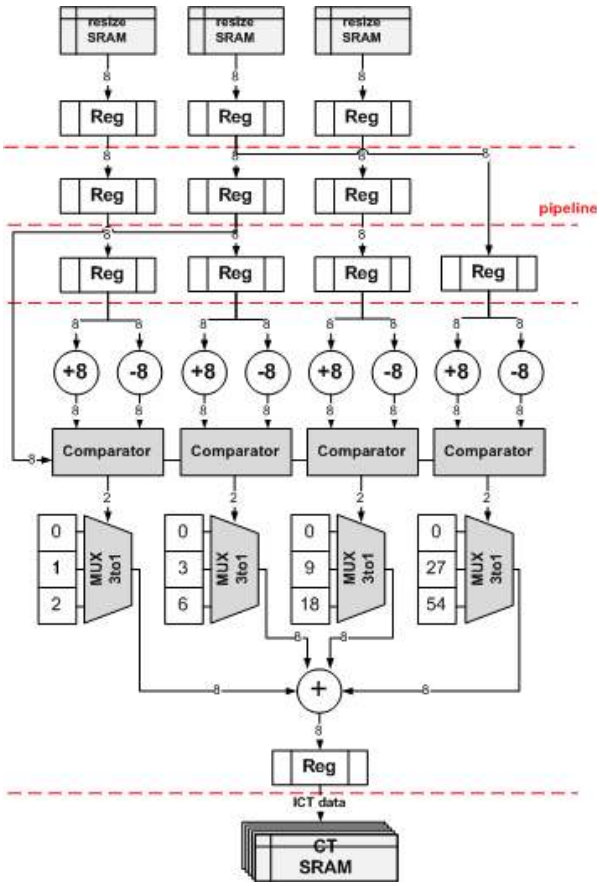


그림 8. ICT 모듈 데이터 패스

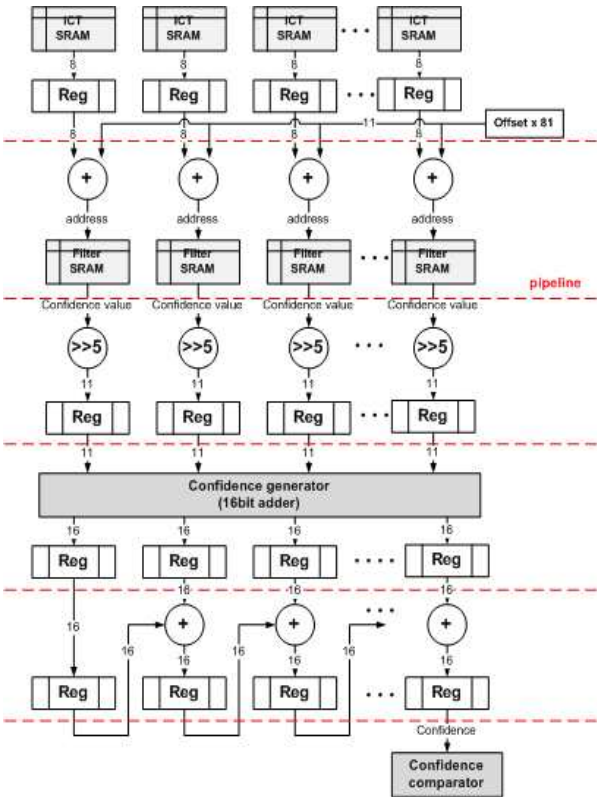


그림 9. Find Candidate 모듈 데이터 패스

과 비교하여 얼굴 인지 아닌지 판별하게 된다. 이와 같이 레지스터를 이용한 파이프라인 구조와 병렬처리를 한 결과, 처음 필터의 행 크기만큼의 클록 소모 후 매 클록마다 얼굴의 후보 영역을 찾을 수 있다.

4. 결과 및 성능 분석

본 논문에서 제안한 하드웨어 설계에 대한 검증 환경은 그림 10과 같다. ARM Cortex A8(1.0GHz) 프로세서를 내장한 Microvison(社)의 MV210 보드와 Libertron(社)의 Virtex5LX330 Logic tile을 사용하고 주변장치로 USB2.0 웹캠과 7인치 LCD로 구성하였다. 설계한 하드웨어를 FPGA에서 검증하기 위하여 Xilinx(社)의 ISE 10.1 sp3 툴을 이용하여 합성 및 검증한 결과 최대 동작 클록은 165.3Mhz이고, 9,039LUTs를 사용하였다. VGA(640×480) 해상도의 흑백 영상을 입력받아 FPGA에서 165MHz로 실시간 동작 검증 한 결과 하나의 얼굴을 검출하는데 약 108 frame/sec로 처리되어 실시간 처리가 요구되는 다양한 어플리케이션에 적용이 가능하였다. UI(User Interface)프로그램의 경우 검출된 얼굴을 소켓통신

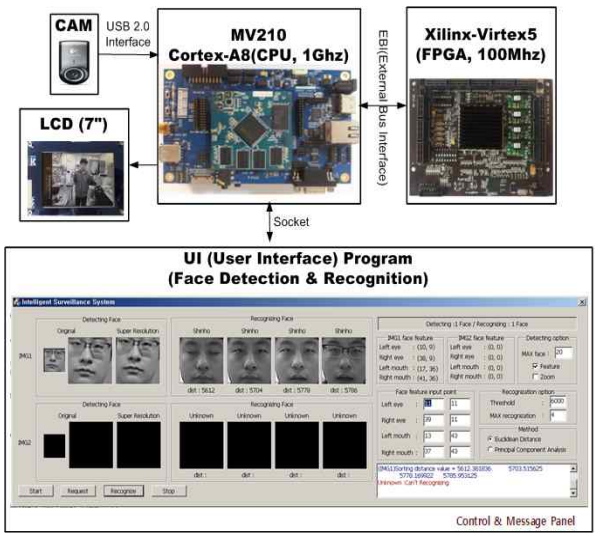


그림 10. FPGA 검증 환경

을 통해 보드로부터 전송받아 보여주고, 미리 저장된 DB와 비교하여 얼굴을 인식하는 프로그램이 내장되어 있다.

구현된 하드웨어의 성능은 표 4와 같다. 최대 20명까지 얼굴을 검출할 수 있으며 QVGA(320×240) 해상도에서는 165Mhz 동작 클록으로 최대 420 frame/sec까지 동작이 가능하지만, 저전력을 위해 12.5MHz의 동작 클록을 사용하더라도 33 frame/sec의 동작 속도를 가지므로 실시간 검출이 가능하다. 얼굴 검출 성능은 BioID[11] 얼굴 DB를 이용하여 측정해본 결과 약 98.5%의 검출률을 보였다. 다양한 환경에서 실험해 본 결과 소프트웨어와 하드웨어의 결과가 동일하게 나타났으며, 이를 실제 영상에 적용시킨 결과 만족스러운 결과를 얻을 수 있었다.

그림 11은 검증 환경에서 실제 얼굴 검출 결과를 보여주는 데모 영상의 일부이다. VGA 해상도를 입

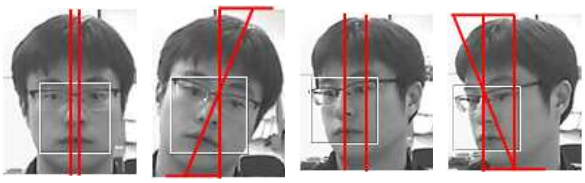
표 4. 구현한 하드웨어 성능

영상 크기	320×240	640×480
처리속도 (165 Mhz)	420 frame/sec	108 frame/sec
메모리	507 KByte	
LUT	9,039 LUTs	
얼굴 검출률 (BioID)	98.5 %	
최대 얼굴 검출 갯수	20 개	

그림 11. 얼굴 검출 데모 영상

력 시 최대 약 3미터 거리까지 얼굴 검출이 가능하였고, 최소 35×35 크기의 얼굴 까지 검출해 낼 수 있다. 광학 줌을 사용한다면 더 먼 거리까지 얼굴 검출이 가능하다. 또한 그림 12에서 볼 수 있듯이 정면 얼굴뿐만 아니라 좌, 우, 상, 하로 약 15도 정도 회전된 얼굴까지 검출해 내는 것을 확인할 수 있다.

표 5와 6은 본 논문의 결과와 유사한 알고리즘을 하드웨어로 구현한 사례[8-10] 들의 성능을 비교한 결과이다. 수행시간은 기존에 구현된 [8], [9], [10] 논문보다 더 빠르게 수행된 것을 알 수 있다. 또한 동일한 동작 클록에서도 실시간 처리가 가능한 것을 확인할 수 있다. FPGA 환경에서의 LUTs 사용면에서 보아도 9,039의 LUTs중 8,846의 Logic element를 가짐으로써 최소의 Hardware Resource로 뛰어난 검출 속도를 가지는 것을 확인할 수 있다.



(a) 정면얼굴 (b) 기울어진 얼굴 검출

그림 12. 회전된 얼굴 검출 예

표 5. 구현 사례와 성능 비교

	영상 크기	Clock speed	최대 처리속도	Logic element	얼굴 검출률
[8]	640×480	100 Mhz	5 frame/sec	15,819	98% (BioID)
[9]	320×240	41 Mhz	136 frame/sec	15,050	86.6% (Yale)
[10]	320×240	54 Mhz	149 frame/sec	26,295	80.48% (MIT/CMU/Yale)
본 논문	320×240	165 Mhz	420 frame/sec	8,846	98.5% (BioID)
	640×480	165 Mhz	108 frame/sec		

표 6. 동일 Clock speed(12.5 Mhz)에서의 성능 비교

	영상 크기	처리 속도
[8]	640×480	0.6 frame/sec
[9]	320×240	41.7 frame/sec
[10]	320×240	30 frame/sec
본 논문	320×240	31.8 frame/sec
	640×480	8 frame/sec

5. 결론 및 향후 연구 방향

필터 방식의 얼굴 검출 알고리즘은 많은 연산과 메모리 접근 시간으로 인하여 임베디드 환경에서 실시간으로 동작이 어렵다. 본 논문에서는 임베디드 환경에서 실시간 얼굴 검출을 위한 파이프라인 구조의 하드웨어를 설계하여 FPGA을 통해 검증하였다.

파이프라인 처리를 위해 알고리즘의 특성을 파악하여 각 연산 모듈을 병렬처리 하였고, 영상 축소, ICT, 얼굴 후보 영역 검출 모듈의 동작을 파이프라인 처리 하였다. 이때 사용되는 메모리의 사용량을 줄이기 위해 각 연산별 메모리 접근 패턴을 분석하고 라인 메모리를 활용하여 메모리 사용량과 접근 횟수를 최소한으로 줄여 설계하였다. 내부 메모리는 507 Kbyte를 사용하였고 FPGA에서 검증 결과 VGA (640×480) 해상도에 동작 클록이 165MHz일 경우 최대 108frame/sec 얼굴 검출이 가능하여 임베디드 환경에서 실시간 동작이 가능함을 확인하였다. 또한 최대 20명까지 얼굴 검출이 가능하며 98.5%의 검출률을 보이고 약 15도까지 기울어진 얼굴까지 검출이 가능하였다.

하지만 약 507Kbyte의 내부 메모리가 임베디드 환경에서 사용하기에는 너무 크다는 단점이 있다. 또한 15도 이상 기울어진 얼굴에 대한 검출 성능 향상도 필요하다. 따라서 메모리 사용량을 최소화하기 위한 노력과 함께 기울어진 얼굴에 대한 검출 성능을 개선 한다면 임베디드 환경에서 다양한 어플리케이션에 활용이 가능할 것으로 기대된다.

참 고 문 헌

[1] Ming-Hsuan Yang, Kriegman, D.J, and Ahuja N, “Detecting Face in Images : a Survey,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 24, pp. 24-58, 2002.

[2] G. Yang and T.S. Huang, “Human Face Detection in a Complex Background,” *Pattern Recognition,* Vol. 27, No. 1, pp. 53- 63, 1994.

[3] Haiyuan WU and Qian CHEN, “Detecting Human Face in Color Images,” *Proc. of IEEE Confon Systems, Man, and Cybernetics*, pp. 2232-2236, 1996.

[4] R. Brunelli and T. Poggio, “Face Recognition : Features versus Templates,” *IEEE Trans. PAMI*, Vol. 15, pp. 1042-1052, 1993.

[5] 이호근 정성태, “실시간 얼굴검출 시스템 설계 및 구현,” 멀티미디어학회논문지, 제8권, 제8호, pp. 1057-1066, 2005.

[6] Bernhard Fröva and Andreas Ernst, “Face Detection with the Modified Census Transform,” *Proc. of IEEE Conf. on AutoSmatic Face and Gesture Recognition*, pp. 91-96, 2004.

[7] Paul Viola and Michael J. Jones, “Robust Real-time Face Detection,” *International Journal of Computer Vision*, pp. 137-154, 2004.

[8] 김윤구, 정용진, “임베디드 시스템 적용을 위한 얼굴 검출 하드웨어 설계,” 전자공학회논문지, 제44권, SD편, 제9호, pp. 40-47, 2007.
[9] Duy Nguyen and David Halupka, “Real-time Face Detection and Lip Feature Extraction Using Fild-Programmable Gate Arrys,” *IEEE Transactions on System, Man and Cybernetics-art B: Cybernetics*, Vol. 36, No. 4, pp. 902-912, 2006.
[10] 한동일 조현종, “고성능 실시간 얼굴 검출 엔진의 설계 및 구현,” 전자공학회논문지, 제47권, SP편, 제2호, pp. 33-34, 2010.
[11] BioID Face Database, http://www.bioid.com/support/ downloads/software/bioid-face- database.html, 2001.

김 신 호

2010년 광운대학교 전자공학과 학사 졸업
2012년 광운대학교 전자통신공학과 석사 졸업
관심분야 : SoC 설계, 영상처리 및 인식, 임베디드 시스템 설계

정 용 진

1983년 서울대학교 제어계측 공학과 학사 졸업
1983년 3월~1989년 8월 한국전자 통신연구원
1995년 미국 UMASS 전자전산공학과 박사 졸업
1995년 4월~1999년 2월 삼성전자 반도체 수석 연구원
1999년 3월~현재 광운대학교 전자통신공학과 정교수
관심분야 : 무선통신, 정보보호, SoC 설계, 영상처리 및 인식, 임베디드 시스템