

SURF 알고리즘 기반 특징점 추출기의 FPGA 설계

류재경[†], 이수현^{**}, 정용진^{***}

요 약

본 논문에서는 특징점 정합을 통한 객체인식, 파노라마 이미지 생성, 3차원 영상 복원 등에 사용될 수 있는 알고리즘 중 대표적인 SURF 알고리즘 기반 특징점 추출기의 하드웨어 구조 설계 및 FPGA 검증 결과에 대해 기술한다. SURF 알고리즘은 크기와 회전변화에 강한 특징점과 서술자를 생성함으로써 객체인식, 파노라마 이미지 생성, 3차원 영상 복원 등에 활용될 수 있다. 하지만 ARM11(667Mhz) 프로세서와 128Mbytes의 DDR 메모리를 사용하는 임베디드 환경에서 실험결과 VGA(640x480) 해상도 영상의 특징점 추출 처리 시 약 7,200msec의 시간이 걸려 실시간 동작이 불가능한 것으로 파악되었다. 본 논문에서는 SURF 알고리즘의 핵심 요소인 적분 이미지 메모리 접근 패턴을 분석하여 메모리 접근 횟수와 메모리 사용량을 줄이는 방법을 이용해 실시간 동작이 가능하도록 하드웨어로 설계하였다. 설계된 하드웨어를 Xilinx(社)의 Vertex-5 FPGA를 이용하여 검증한 결과 100Mhz 클럭에서 VGA 영상의 특징점 추출시 약 60frame/sec로 동작하여 실시간 응용으로 충분함을 알 수 있다.

FPGA Design of a SURF-based Feature Extractor

Jae-Kyung Ryu[†], Su-Hyun Lee^{**}, Yong-Jin Jeong^{***}

ABSTRACT

This paper explains the hardware structure of SURF(Speeded Up Robust Feature)-based feature point extractor and its FPGA verification result. SURF algorithm produces novel scale- and rotation-invariant feature point and descriptor which can be used for object recognition, creation of panorama image, 3D image restoration. But the feature point extraction processing takes approximately 7,200msec for VGA-resolution in embedded environment using ARM11(667Mhz) processor and 128Mbytes DDR memory, hence its real-time operation is not guaranteed. We analyzed integral image memory access pattern which is a key component of SURF algorithm to reduce memory access and memory usage to operate in c real-time. We assure feature extraction that using a Vertex-5 FPGA gives 60frame/sec of VGA image at 100Mhz.

Key words: SURF, Feature Extraction(특징점 추출), Field Programmable Gate Array(FPGA)(하드웨어 구현), Integral Image(적분이미지)

1. 서 론

객체인식 및 영상분석을 위한 방법으로는 사물의

움직임을 이용하는 방법, 엣지 성분을 추출하는 방법, 사물의 고유 특징점을 추출하는 방법 등이 있다. 사물의 특징점을 추출하는 대표적인 알고리즘으로

* 교신저자(Corresponding Author): 류재경, 주소: 서울특별시 노원구 월계1동 광운대학교 참빛관 810호 실시간구조 연구실(139-701), 전화: 02)940-5551, FAX: 02)942-5517, E-mail: jjackq@kw.ac.kr

접수일: 2010년 9월 10일, 수정일: 2010년 11월 30일
완료일: 2011년 1월 26일

[†] 정회원, 광운대학교 전자통신공학과 석사

^{**} 정회원, 광운대학교 전자통신공학과 박사과정
(E-mail: madvam@daum.net)

^{***} 정회원, 광운대학교 전자통신공학과 정교수
(E-mail: yjjeong@kw.ac.kr)

* 이 논문은 2010년 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업 연구임. (NRF-2010-0014557)

SIFT(Scale Invariant Feature Transform)[1], MSER(Maximally Stable Extremal Regions)[2], SURF(Speeded Up Robust Features)[3] 등이 있다. 그 중 SURF 알고리즘은 SIFT처럼 크기와 회전 변화에 강건한 특징점을 추출하는 알고리즘으로 객체 인식[4]이나 파노라마 이미지 생성[5] 3차원 영상 복원[6] 등에 활용할 수 있다. 하지만 복잡한 반복연산이 많아 연산시간이 오래 걸린다는 단점이 있다. 예를 들어 임베디드 환경의 ARM11(667Mhz) 프로세서와 128Mbytes DDR 메모리를 이용하여 VGA(640× 480) 해상도의 영상을 처리한 결과 약 7,200msec의 시간이 소요됨으로서 30frame/sec의 실시간 동작이 불가능한 것으로 파악되었다. 그러므로 모바일 기기와 같은 임베디드 환경에서 실시간으로 동작하기 위한 고성능의 하드웨어 IP 개발이 필요하다.

SURF 알고리즘은 크기 변화에 강건한 특징점을 추출하는 스케일 공간(Scale space)[7]을 표현하기 위해 이미지 피라미드(Image pyramids)를 생성하고 생성된 이미지 피라미드에서 특징점을 추출한다. 이때 사용되는 적분 이미지(Integral image)는 SURF 알고리즘의 핵심 요소이다. 하지만 실시간 동작을 위해 병렬구조를 이용한 하드웨어 설계시 적분이미지 메모리의 사용량이 크다는 단점이 있다. 본 논문에서는 적분 이미지 메모리의 접근 패턴 분석을 통해 메모리 접근 횟수와 메모리 사용량을 줄여 하드웨어를 설계하고 SURF와 유사한 성능의 SIFT 알고리즘 기반 특징점 추출기를 하드웨어로 구현한 논문[8,9]과 성능을 비교한다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문에서 구현한 SURF 알고리즘의 특징점 추출 과정에 대해 기술하고, 3장에서는 하드웨어 설계를 위한 가우시안 가중치(Gaussian weight)의 고정 소수점 모델링의 성능 분석, 4장에서는 본 논문에서 제안하는 하드웨어 설계 기법에 대해 기술한다. 5장에서는 제안된 하드웨어 설계의 검증 결과와 분석에 대해 기술하고, 마지막으로 6장에서 결론 및 향후 연구방향에 대해 논한다.

2. 알고리즘

SURF 알고리즘은 크기와 회전 변화에 강건한 특

징점(Feature point)과 서술자(Descriptor)를 생성하는 알고리즘이다. 헤이시안 검출기(Fast-Hessian detector)를 이용하여 크기 변화에 강한 특징점을 추출하고 추출된 특징점에 회전 변화에 강한 서술자를 부여한다. 본 논문에서 설계하고자 하는 헤이시안 검출기는 SURF 알고리즘에서 가장 반복연산이 많은 부분으로 적분 이미지를 이용해 박스 필터(Box filter) 연산을 수행하고, 박스 필터 연산을 통해 계산된 헤이시안 행렬식을 이용하여 이미지 피라미드를 생성하여 특징점을 추출한다[7].

2.1 적분 이미지

적분 이미지는 그림 1과 같이 영상 내에서 특정 사각형 영역 내에 존재하는 화소값의 합을 식 (1)을 이용하여 계산한다. 식(1)의 i 는 적분 이미지 화소값을 나타내고, i 는 8비트 명암도 입력 영상을 나타낸다.

$$\ddot{i}(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y') \quad (1)$$

그림 1의 우측 그림을 보면 특정 영역의 화소값의 합을 식 (2)와 같이 간단한 연산만으로 결과를 얻을 수 있다. 식 (2)의 D(Sum of Pixel)는 D공간 내에 있는 화소값들의 합을 나타낸다.

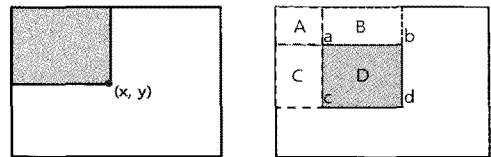


그림 1. 적분 이미지(좌), 적분 이미지를 이용한 연산의 예(우)

$$\sum (i, j) \text{ in } D = \ddot{i}(d) + \ddot{i}(a) - \ddot{i}(b) - \ddot{i}(c) \quad (2)$$

적분 이미지를 이용하면 헤이시안 검출기에서 사용되는 박스 필터의 크기와 상관없이 일정시간에 필터내 화소값의 합을 얻을 수 있다.

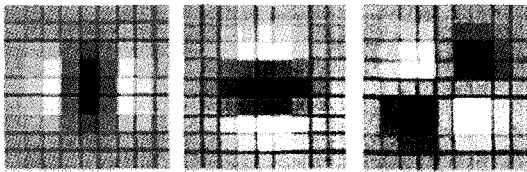
2.2 헤이시안 검출기

헤이시안 검출기는 식 (3)과 같은 헤이시안 행렬을 기반으로 특징점을 추출한다. 이때 L_{xx} 는 x 위치의 입력영상 화소값과 σ 의 분산을 갖는 가우시안의 x 방향 2차 미분값과의 컨볼루션 값을 의미하고, 나머지

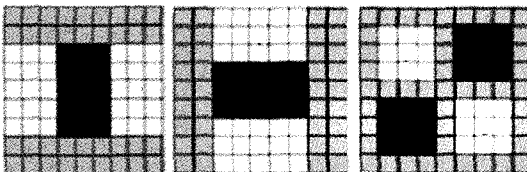
L_{yy}, L_{xy} 는 y 방향으로의 2차미분, xy 방향으로 미분된 가우시안 필터와의 컨볼루션 값을 의미한다[4].

$$\mathcal{H}(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix} \quad (3)$$

SURF 알고리즘에서 사용하는 헤이시안 검출기는 그림 2 (A)의 가우시안 2차 미분필터를 사용하지 않고, 그림 2 (B)와 같이 표현된 근사화 된 박스 필터를 적용한다. 앞서 얻어진 적분 이미지와 근사화 된 박스필터를 이용함으로써 박스 영역의 크기에 상관 없이 빠르게 헤이시안 행렬식의 근사값을 계산할 수 있다. 적분 이미지에 근사화 된 박스필터를 적용한 결과값을 식 (4)에 대입하여 헤이시안 행렬식을 구할 수 있다. 식 (4)의 $\det(H_{approx})$ 는 헤이시안 행렬식 근사값이고, D_{xx}, D_{yy}, D_{xy} 는 가로, 세로, 대각선 방향의 근사화된 박스필터 컨볼루션 결과값을 나타낸다.



(A) xx, yy, xy 방향의 가우시안 2차 미분필터



(B) xx, yy, xy 방향의 근사화된 박스필터

그림 2. 가우시안 2차 미분과 근사화된 박스 필터

$$\det(H_{approx}) = D_{xx} D_{yy} - (D_{xy})^2 \quad (4)$$

각 스케일 별 박스필터의 크기를 변경시켜 헤이시안 행렬식의 값을 구하게 되면 그림 3의 좌측과 같은 이미지 피라미드를 생성할 수 있다. 그리고 생성된 이미지 피라미드에서 헤이시안 행렬식의 값이 임계값(Threshold)보다 크고 인접 3×3 영역의 이웃 픽셀보다 클 경우 특징점으로 추출된다.

3. 고정 소수점 모델링

SURF 특징점 추출 알고리즘의 부동 소수점 모델

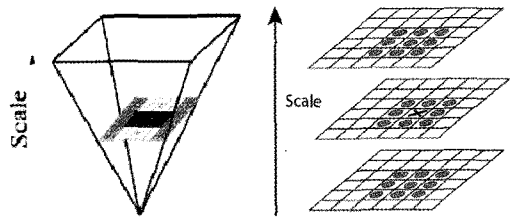


그림 3. 이미지 피라미드(좌), 특징점 추출 방법(우)

은 ARM11(667Mhz)의 프로세서를 사용하는 임베디드 환경에서 약 7,200msec 정도의 수행시간이 걸렸다. 따라서 실시간 처리를 위해 하드웨어로 설계하기 위해서는 고정 소수점 모델링을 통한 데이터패스의 적당한 비트 길이(Bit-width)를 정해야 한다. SURF의 헤이시안 검출기에서 헤이시안 행렬식을 계산하기 위해 사용되는 각 박스필터에 곱해지는 가우시안 가중치를 고정 소수점 모델로 변경해야 한다.

고정 소수점 모델의 성능을 측정하기 위해 K.Mikolajczyk 의 평가 소스 [10,11]를 이용해 Repeatability를 측정하였다. Repeatability는 서로 다른 영상에서 밝기, 시점 등과 같은 환경변수의 변화에 관계없이 특징점이 같은 부분에서 반복적으로 추출되는 정도를 나타낸다. 식 (5)와 같이 두 특징점이 추출된 영역 사이의 차영역(intersection)과 공통영역(union)을 이용하여 에러율(E)을 구하고 에러율을 이용해 얼마나 동일한 지점에서 특징점이 추출되었는가를 평가한다.

$$E = 1 - \frac{\text{intersection}}{\text{union}} \quad (5)$$

그림 4는 가우시안 가중치 값의 비트 길이를 늘려가며 Repeatability 성능을 측정한 결과이다. 그림에서 볼 수 있듯이 98% 이상의 Repeatability 성능을 만족하기 위해서는 15bits 이상의 비트수를 사용해

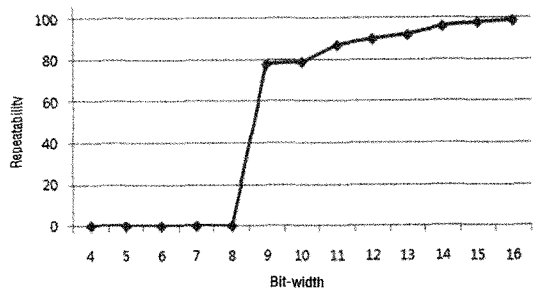


그림 4. Repeatability 측정 결과

야 함을 확인할 수 있다. 또한 데이터베이스에 저장된 물체와 화면상의 물체 사이의 호모그래피(Homography)를 찾아 화면에 표시하는 소프트웨어를 이용해 크기와 회전, 밝기 변화 등 서로 다른 환경의 약 100장의 사진을 실험한 결과 15bits의 가우시안 가중치 사용 시 부동 소수점 모델의 결과와 비교하여 97% 이상의 인식 성능을 보였다.

이 결과를 바탕으로 SURF 기반 특징점 추출기를 설계함에 있어 15bits의 가우시안 가중치 값을 사용하였다.

4. 하드웨어 설계

4.1 적분 이미지 접근 횟수 감소

적분 이미지는 SURF 알고리즘에서 특징점을 추출하기 위한 박스필터 연산의 핵심 요소이다. 하지만 적분 이미지 메모리 접근 횟수가 너무 많아 실시간 동작을 위한 하드웨어 구현이 어렵다.

그림 5는 근사화 된 박스필터이다. xx방향 필터의 값을 계산하기 위해서는 그림에서 흰색과 검정색 사각형으로 표시된 모서리 부분의 적분 이미지 화소값을 모두 읽어야 하기 때문에 8번의 적분 이미지 메모리 접근이 필요하다. 따라서 헤이시안 행렬식 화소값 하나를 계산하기 위해 xx방향 필터를 계산하기 위해 8번, yy방향 필터를 계산하기 위해 8번, xy방향 필터를 계산하기 위해 16번, 총 32번의 적분 이미지 메모리 접근이 필요하다.

본 논문에서는 그림 6과 같은 메모리 구조를 이용하여 메모리 접근 시간을 감소시켰다. 그림 6의 메모리 구조를 보면 적분 이미지 메모리를 행(Row)단위로 구성하여 각각의 행에 주소를 주어 행의 개수만큼의 데이터를 동시에 읽을 수 있도록 구성한다. 이런 메모리 구조를 이용하여 그림 5의 같은 도형으로 표시된 화소값을 동시에 읽어 총 4번의 적분 이미지 메모리 접근으로 헤이시안 행렬식의 화소값을 계산하였다.

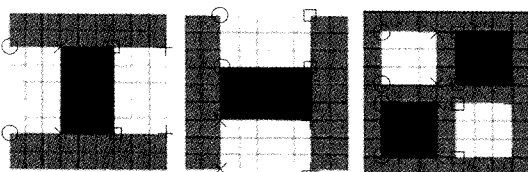


그림 5. 적분 이미지 접근 방법

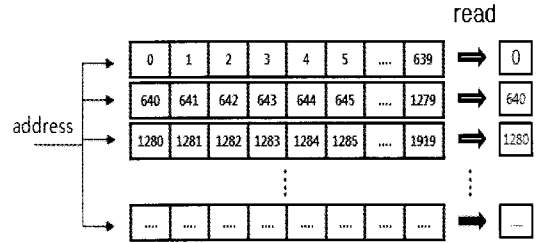


그림 6. 메모리 구조

4.2 적분 이미지 메모리 사용량 감소

본 논문에서 제안하는 실시간 동작이 가능한 하드웨어 구현을 위해서는 각 스케일에 필요한 적분 이미지 메모리의 병렬구성이 필요하다. 이 때 사용되는 메모리 사용량을 줄이기 위해 각 스케일 공간에서 필요한 최소한의 메모리 사용량을 계산하고 적분 이미지 메모리 접근 패턴을 분석하여 메모리 사용량을 최소화 하였다.

표 1은 헤이시안 행렬식을 계산하기 위해 각 스케일에서 사용되는 박스필터 크기이다. 각 스케일 공간에서 사용되는 박스필터 크기를 고려하여 VGA영상 전체에 대한 적분 이미지 화소값을 저장하지 않고 각 스케일 공간에서 필요한 최소한의 메모리만 구성한다. 그리고 적분 이미지 메모리 구성시 한 행의 메모리를 추가하여 박스필터 연산을 위한 읽는 동작 외에 적분 이미지 화소값을 쓸 수 있도록 그림 7과 같이 구성하였다.

또한, 적분 이미지 메모리 접근 패턴을 분석하여 추가적으로 메모리 사용량을 줄일 수 있다. 헤이시안 검출기에서는 박스 필터의 크기가 커짐에 따라 화소값을 샘플링하는 간격이 넓어지게 된다. 따라서 적분 이미지 화소값을 모두 사용하지 않고 헤이시안 행렬식을 계산하게 된다. 그림 8은 연속된 헤이시안 행렬식 화소값을 계산하기 위한 컨볼루션 연산 과정을 나타낸 것으로, 사각형 안의 숫자는 해당 화소에 접

표 1. 각 스케일에서 사용되는 박스필터 크기

	박스필터 크기
스케일 0	7×7
스케일 1	9×9
스케일 2	15×15
스케일 3	21×21
스케일 4	27×27
스케일 5	33×33

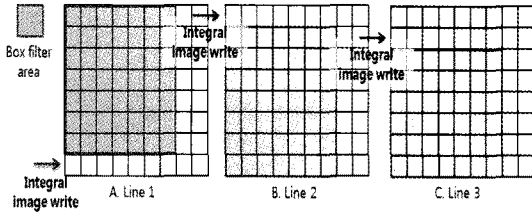


그림 7. 적분 이미지 메모리 접근 방법

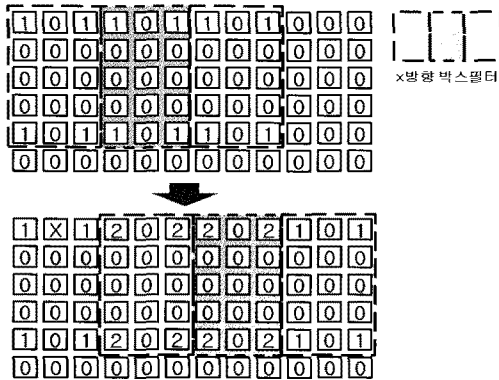


그림 8. 적분 이미지 접근 패턴

근한 횟수를 나타낸다. 그림 9의 하단 그림은 두 번째 헤이시안 행렬식 화소값을 계산하기 위해 박스필터를 이동한 것으로, 그림에서 'X'로 표시된 화소는 컨볼루션을 위한 박스필터의 이동에서 더 이상 접근이 필요 없는 화소를 표시한 것이다.

따라서 모든 스케일에서 박스 필터 크기내의 적분 이미지 화소값을 모두 저장할 필요가 없다. 본 논문에서 사용된 6개의 스케일을 예로 들었을 경우 세 번째 스케일까지는 적분 이미지 화소값을 모두 사용하지만 나머지 세 개의 스케일에서는 그림 9에서 어두운 색으로 표시된 것과 같이 일정한 패턴으로 적분 이미지 메모리에 접근하게 된다. 제안하는 하드웨어 구조에서는 그림 9의 어두운 색으로 표시된 부분의

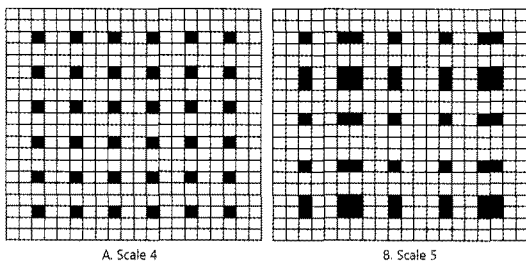


그림 9. 각 스케일의 적분 이미지 접근 패턴

표 2. 적분 이미지 메모리 사용량

	박스필터 크기만 고려한 경우 메모리 사용량	적분 이미지 패턴 분석을 이용한 메모리 사용량
스케일 0	640×8×27 bits	640×8×27 bits
스케일 1	640×10×27 bits	640×10×27 bits
스케일 2	640×16×27 bits	640×16×27 bits
스케일 3	640×22×27 bits	458×16×27 bits
스케일 4	640×28×27 bits	214×10×27 bits
스케일 5	640×34×27 bits	291×17×27 bits
전체	2,039,040 bits	976,725 bits

메모리 크기만 고려하여 메모리를 생성하였다.

표 2는 박스필터 크기만 고려한 경우의 메모리 사용량과 적분 이미지 메모리 접근 패턴을 분석하여 메모리사용량을 추가적으로 줄인 결과이다. 샘플링 간격과 박스 필터의 크기가 큰 스케일에서는 박스필터의 크기가 작은 스케일에 비해 필요한 메모리 사용량이 적다.

스케일 0, 1, 2에서는 메모리 사용량이 같지만 스케일 3, 4, 5에서는 적분 이미지 화소값을 저장하기 위해 필요한 행과 열의 크기가 작은 것을 확인할 수 있다. 표 2의 전체 적분이미지 메모리 사용량을 보면 박스필터의 크기만 고려한 메모리 구성보다 약 53% 가량 메모리 사용량이 추가적으로 감소한 것을 확인할 수 있다.

4.3 실시간 동작을 위한 설계

본 논문에서는 SURF 알고리즘 기반 특징점 추출기의 실시간 동작을 위해 적분 이미지 접근 패턴을 분석하여 메모리 접근 횟수와 메모리 사용량을 최소화하여 설계하였다. 그리고 이를 이용해 그림 10의 전체 하드웨어 구조도와 같이 적분 이미지를 계산하는 모듈과 헤이시안 행렬식을 계산하는 모듈, 그리고 특징점 추출 모듈을 파이프라인으로 처리하고 각 스케일 공간 처리를 병렬 구조로 설계하였다. 다음에서 적분 이미지 생성 모듈, 헤이시안 행렬식 계산 모듈, 특징점 검출(Non-maxima suppression) 모듈의 하드웨어 구조에 대해 자세히 기술한다.

4.3.1 적분 이미지 생성 모듈

그림 11은 적분 이미지 생성기에 대한 블록 다이

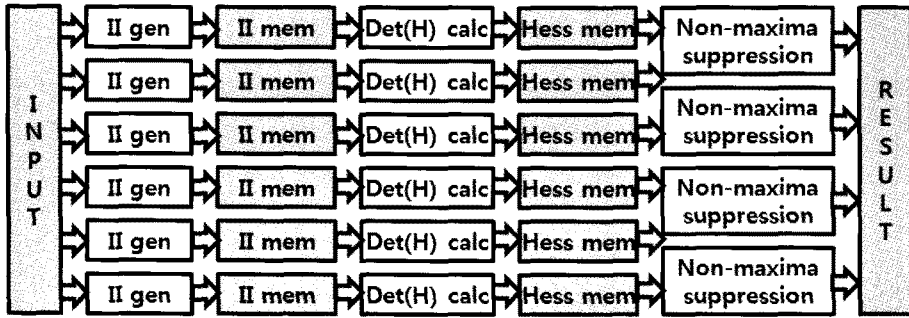


그림 10. 제안된 하드웨어 전체 구조도

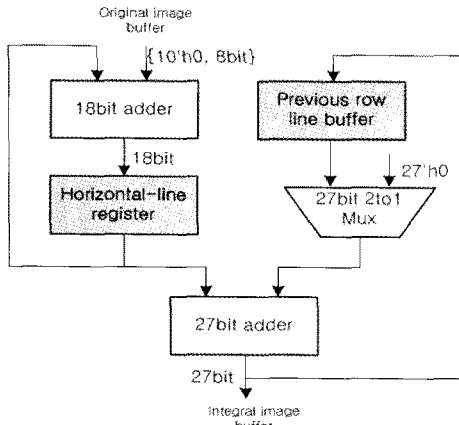


그림 11. 적분 이미지 생성 모듈의 블록 다이어그램

어그램이다. 각 행의 화소값의 누적 합을 계산하기 위한 18bits Adder기와 레지스터를 두고 이미 계산된 이전 행의 적분 이미지 화소값을 읽어 27bits 덧셈기를 이용해 적분 이미지 화소값을 계산한다. 이때 이전 행의 화소값을 저장하기 위한 버퍼를 별도로 생성하였다. 계산된 화소값은 해당 화소값이 필요한 스케일들의 적분 이미지 버퍼들에 동시에 저장한다.

4.3.2 헤이시안 행렬식 계산 모듈

헤이시안 행렬식을 계산하기 위해서는 박스필터 각 모서리의 적분 이미지 화소값이 필요하다. 하지만 앞서 제안한 적분 이미지의 메모리를 행 단위로 구분하여 저장하게 되면 각 열의 적분 이미지 화소값을 동시에 읽을 수 있다.

그림 12는 헤이시안 행렬식 계산 모듈의 블록 다이어그램이다. 적분 이미지에 4번의 접근만으로 헤이시안 행렬식 계산에 필요한 화소값 모두 저장하고 헤이시안 행렬의 각 원소값을 계산하는 동안 다음

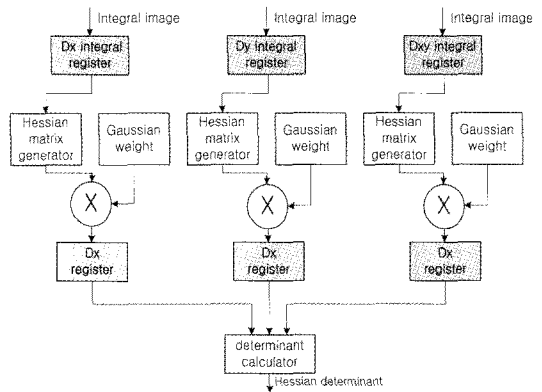


그림 12. 헤이시안 행렬식 계산 모듈의 블록 다이어그램

적분 이미지 화소값을 읽는 방식으로 파이프라인 처리한다. 가우시안 가중치를 곱하기 위해 24비트 곱셈기를 이용하여 최종 헤이시안 행렬식 화소값을 계산한다.

4.3.3 특징점 검출 모듈

헤이시안 행렬식을 이용하여 이미지 피라미드가 완성되면 임계값과 주변 화소들의 값과 비교하여 특징점을 추출할 수 있다. 본 논문에서 제안하는 방법은 행 단위로 연산을 수행하여 주변 화소값과 비교하기 위한 최소 열(Column)의 헤이시안 행렬식 화소값이 계산될 때마다 특징점을 추출하고 더 이상 필요가 없어진 열 공간은 다음 화소값을 저장하기 위해 사용한다.

그림 13은 해당 화소의 값이 특징점인지 판별하기 위한 특징점 검출 모듈을 블록 다이어그램으로 나타낸 그림이다. 특징점이 유효한지 판별하기 위해 해당 화소 주변 화소값을 9개를 총 3번의 읽기 동작으로 해당 화소가 특징점으로 유효한지 판별하게 된다.

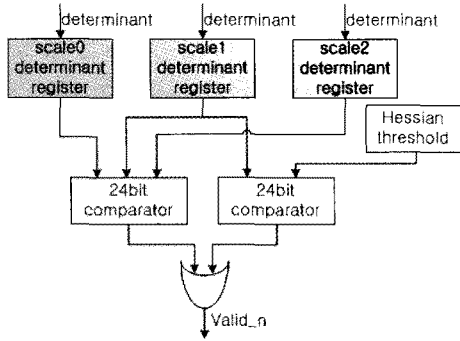


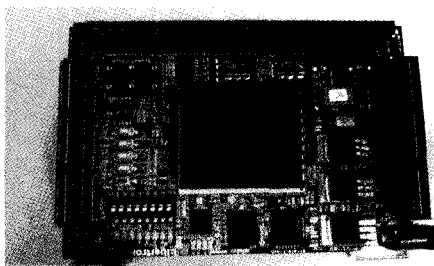
그림 13. 특징점 검출 모듈의 블록 다이어그램

5. 결과 및 분석

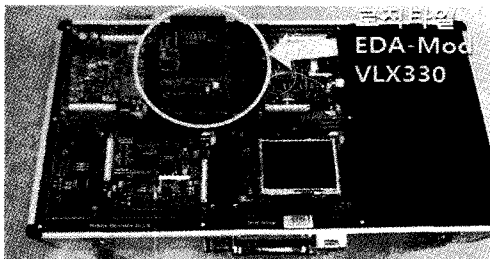
표 3. 연산시간 측정 (단위 : msec)

구동환경 / 단계	Intel Core2 Duo (2.4GHz)	ARM11 (533Mhz)	하드웨어 (100Mhz)
적분 이미지 생성	1.5	1,549.3	18.0
헤이시안 검출기	115.0	5,612.3	
전체	116.5	7,161.6	18.0

하드웨어 설계에 대한 검증은 PXA320(806Mhz)



(a) EDA-Mod 5VLX330



(b) HBE-SM2

그림 14. 검증환경

Processor와 128Mbytes의 NAND Flash Memory, 128Mbytes Mobile-DDR RAM을 탑재한 한백전자(社)의 HBE-SM2 보드와 리버트론(社)의 EDA-Mod 5VLX330 로직 타일을 사용하였다. Xilinx ISE 10.1툴에서 Verilog HDL을 이용하여 Register Transfer Level 로 설계하고 Mentor(社)의 Model-Sim 6.2b를 이용하여 시뮬레이션 검증을 하였다. 시뮬레이션 결과 최대 동작 속도는 116.2 Mhz이고, FPGA 검증을 위해 100 Mhz 클럭을 사용하였다. 그림 14는 FPGA 검증 환경을 나타낸다.

그림 15는 소프트웨어를 이용하여 특징점을 추출한 결과와 본 논문에서 제안된 구조를 이용한 하드웨어를 이용하여 특징점을 추출한 결과 영상이다. 고정 소수점 모델링을 통한 오차 외에 대부분의 특징점이 동일한 지점에서 추출되었다.

그림 16을 보면 적분 이미지를 생성한 후 헤이시안 행렬식을 계산하고 특징점을 비교하기 위한 최소한의 헤이시안 행렬식 화소값 계산 후 특징점 추출 모듈이 동작하는 것을 확인할 수 있다. 이때 적분 이미지 생성 모듈의 경우 읽기, 수행, 쓰기 동작의 3 클럭으로 동작이 가능하고, 헤이시안 행렬식 계산



(A) 소프트웨어



(B) 하드웨어

그림 15. 특징점 추출 결과영상

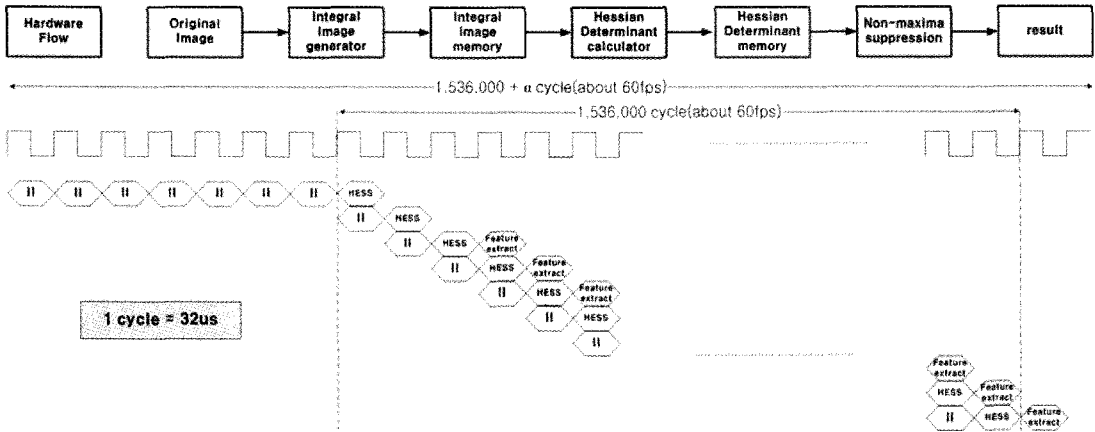


그림 16. 타이밍 분석

모듈은 읽기 동작 4번과 쓰기 동작 1번을 이용한 5 클럭에 동작이 가능하다. 그리고 특징점 검출 모듈은 읽기 3번과 쓰기 1번의 4 클럭에 동작이 가능하다.

따라서 앞에서 기술한 파이프라인 구조에 따라 가장 많은 시간이 소요되는 헤이시안 행렬식 계산 모듈은 5 클럭에 전체 동작이 가능하며 이를 frame/sec로 계산하면 100 Mhz 클럭에서 약 60 frame/sec로 동작이 가능하다. 표 3은 VGA 해상도의 영상을 데스크톱 환경과 임베디드 환경에서의 소프트웨어 연산시간과 본 논문에서 기술한 하드웨어 설계의 실제 연산시간을 측정된 결과이다. 데스크톱 환경에서 약 10 frame/sec, 임베디드 환경의 ARM11 processor에서 0.13frame/sec가 걸렸지만 본 논문에서 설계한 하드웨어의 경우 약 60frame/sec의 속도로 실시간 동작이 가능하다.

표 4는 SURF와 유사한 성능의 SIFT 알고리즘을 하드웨어로 구현한 사례[8,9] 논문과의 성능을 비교한 결과이다. 수행시간은 320×240 해상도의 입력 영

표 4. 성능비교

	[8]	[9]	본 논문
수행시간	33ms	31ms	18ms
입력영상 해상도	320×240	640×480	640×480
SRAM (Kbyte)	168.75	405.00	268.62
Device	Altera Stratix II S60	Xilinx Virtex5 ML507	Xilinx Virtex5 LX330

상을 이용한 [8] 논문보다도 빠르게 수행된 것을 알 수 있다. 또한 동일한 입력 영상 크기를 사용한 논문 [9]의 405Kbyte SRAM 사용에 비해 268.62Kbyte 메모리 사용으로 메모리 사용량에서 좋은 성능을 보였다.

검증에 사용된 Vertex-5 FPGA에서 사용된 LUT의 양은 39,876 LUTs 로 전체 LUT의 19%를 차지하였고 74개의 곱셈기 코어가 사용되었다. 그리고 표 5에서 확인할 수 있듯이 적분 이미지를 저장하기 위한 메모리가 122 Kbytes, 헤이시안 행렬식 화소값을 저장하기 위한 메모리가 146 Kbytes, 전체 블록 메모리가 약 268 Kbytes가 사용되었다.

표 5. 제안된 구조의 세부 메모리 사용량

	스케일	메모리 사용(bytes)	
적분 이미지 메모리	0	17,280	122,147
	1	21,600	
	2	34,560	
	3	24,736	
	4	7,222	
	5	16,753	
헤이시안 행렬식 메모리	0	19,200	146,478
	1	51,840	
	2	20,736	
	3	27,126	
	4	16,614	
	5	10,962	
전 체	-	268,625	

6. 결론 및 향후 연구방향

SURF 알고리즘을 이용하여 임베디드 환경에서 실시간 객체인식을 하기 위해서는 속도의 개선이 필요하다. 본 논문에서는 임베디드 환경에서 실시간 객체인식을 위한 SURF 알고리즘 기반 특징점 추출기의 하드웨어 설계 방법과 FPGA를 이용한 검증 결과 및 분석에 대해 기술하였다.

특징점 추출 하드웨어의 실시간 동작을 위해 각 스케일 공간 처리를 병렬로 처리하였고, 적분 이미지 생성, 헤이시안 행렬식 계산 특징점 검출 모듈의 동작을 파이프라인 처리하였다. 이때 사용되는 메모리 사용량을 줄이기 위해 적분 이미지 메모리 접근 패턴을 분석하여 메모리 사용량과 메모리 접근 횟수를 줄여 설계를 하였다.

본 논문에서 사용한 구조를 이용하여 SURF 알고리즘 기반 특징점 추출기를 구현하였을 경우 FPGA에서 전체 약 268Kbytes의 내부 메모리가 사용되었고, VGA 해상도의 영상에서 특징점 추출 동작시 약 60 frame/sec의 성능을 낼 수 있었다. 또한, SURF와 유사한 SIFT 알고리즘 기반 특징점 추출 하드웨어를 구현한 논문[8,9]과 비교한 결과 더 빠른 속도로 동작함을 확인했다.

하지만 내부 메모리로 사용되는 268Kbytes의 메모리가 임베디드 환경에서 사용하기에 크다는 단점을 보완할 필요성이 있다. 따라서 메모리 사용량을 줄이는 노력 등을 통해 임베디드 환경에서 객체인식 및 파노라마 이미지 생성, 로봇의 자기위치 판단 등에 활용이 가능할 것으로 기대된다.

참 고 문 헌

- [1] Lowe, D, "Distinctive Image Features From Scale-Invariant Keypoints, Cascade Filtering Approach," *IJCV* 60, 91-110, 2004.
- [2] J. Matas, O. Chum, M. Urban, and T.Pajdla, "Robust Wide Baseline Stereo From Maximally Stable Extremal Regions," *BMVC*, 384-393, 2002.
- [3] H. Bay, T. Tuytelaars, and L. Van Gool, "Speeded Up Robust Features," *ECCV* 1, 404, 2006.
- [4] Patricio Loncomilla and Javier Ruiz del Solar, "Improving Sift-Based Object Recognition for Robot Applications," *ICIAP*, Vol.3617, pp. 1084-1092, 2005.
- [5] Kyung Ho Jang, Jae Seok Jang, Seok Jun Lee, and Soon Ki Jung, "Semi-Automatic 3D Building Reconstruction from Uncalibrated Images," *Journal of Korea Multimedia Society* Vol.12, No.9, September 2009(pp. 1217-1232)
- [6] Luo Juan and Oubong Gwun, "SURF Applied in Panorama Image Stitching," *IEEE 10.1109/IPTA.2010.5586723*
- [7] Lindeberg T, "Feature Detection With Automatic Scale Selection," *IJCV* 30, 79-116, 1998.
- [8] V.Bonato, "A Parallel Hardware Architecture for Scale and Rotation Invariant Feature Detection," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.18, No.12, 2008.
- [9] L. Yao, "An Architecture of Optimized SIFT Feature Detection for an FPGA Implementation of an Image Matcher," *IEEE FPT*, 2009.
- [10] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool, "A comparison of affine region detectors," *IJCV* 65(1/2):43-72, 2005
- [11] The Visual Geometry Group, Katholieke Universiteit Leuven, <http://www.robots.ox.ac.uk/~vgg/research/affine/>.



류 재 경

2009년 광운대학교 전자통신공학과 학사 졸업
2011년 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 한국전자통신연구원
1995년 미국 UMASS 전자전산공학과 박사 졸업

1995년 4월~1999년 2월 삼성전자 반도체 수석 연구원
1999년 3월 광운대학교 전자통신공학과 정교수
관심분야: 무선통신, 정보보호, SoC 설계, 영상처리 및 인식, 임베디드 시스템



이 수 현

2005년 광운대학교 전자공학부 학사 졸업
2007년 광운대학교 전자통신공학과 석사 졸업
2007년 3월~현재 광운대학교 전자통신공학과 박사과정 재학

관심분야: SoC 설계, 영상처리 및 인식, 임베디드 시스템 설계