2013년 5월 전자공학회 논문지 제 50 권 제 5 호
Journal of The Institute of Electronics Engineers of Korea Vol. 50, NO. 5, May 2013 http://dx.doi.org/10.5573/ieek.2013.50.5.083

논문 2012-50-5-10

객체 추적을 위한 SURF 기반 특이점 추출 및 서술자 생성의 하드웨어 설계

(Hardware Design of SURF-based Feature extraction and description for Object Tracking)

도 용 식*, 정 용 진**

(Yong-Sig Do© and Yong-Jin Jeong)

요 약

최근 영상처리 응용의 일환으로 객체 추적 시스템에 많이 활용되는 SURF 알고리즘의 경우 영상의 회전 및 크기 변화에 강인한 특이점을 추출한다는 특징이 있지만 연산이 복잡하고 연산량이 많아 임베디드 환경에서 IP로 사용되기 위해서는 하드웨어 가속기 개발이 필수적이다. 하지만 이 때 요구되는 내부 메모리 사이즈가 매우 크기 때문에 ASIC이나 SoC 시스템으로 개발 할 때 칩 회로 사이즈가 커서 IP의 가치를 떨어뜨리게 된다. 본 논문에서는 하드웨어 가속기 개발 시 회로면적에 효율적인 설계를 위해 내부 블록메모리 사용량을 줄이고 외부 메모리와 DMA를 사용하여 세분화된 Sub-IP 구조로 설계하는 것에 대해 연구하고 간단한 객체 추적 알고리즘을 개발하여 그 결과를 적용하였다. ARM Cortex-M0, AHB-lite, APB, DMA, SDRAM Controller로 구성된 시스템 환경에서 실험 결과 VGA(640x480)영상에서 SURF 알고리즘의 처리속도는 약 31frame/sec, 블록 메모리의 크기는 81Kbytes, 30nm 공정에서 회로의 크기는 약 74만 게이트 크기로 SoC 칩의 하드웨어 IP로 활용이 가능하였다. SURF와 비슷한 영상처리 알고리즘에서도 본 논문에서 제안하는 설계방법을 적용하면 타겟 어플리케이션에 효율적인 하드웨어 설계를 할 수 있을 것으로 기대된다.

Abstract

Recently, the SURF algorithm, which is conjugated for object tracking system as part of many computer vision applications, is a well-known scale- and rotation-invariant feature detection algorithm. The SURF, due to its high computational complexity, there is essential to develop a hardware accelerator in order to be used on an IP in embedded environment. However, the SURF requires a huge local memory, causing many problems that increase the chip size and decrease the value of IP in ASIC and SoC system design. In this paper, we proposed a way to design a SURF algorithm in hardware with greatly reduced local memory by partitioning the algorithms into several Sub-IPs using external memory and a DMA. To justify validity of the proposed method, we developed an example of simplified object tracking algorithm. The execution speed of the hardware IP was about 31 frame/sec, the logic size was about 74Kgate in the 30nm technology with 81Kbytes local memory in the embedded system platform consisting of ARM Cortex-M0 processor, AMBA bus(AHB-lite and APB), DMA and a SDRAM controller. Hence, it can be used to the hardware IP of SoC Chip. If the image processing algorithm akin to SURF is applied to the method proposed in this paper, it is expected that it can implement an efficient hardware design for target application.

Keywords : SURF algorithm, Sub-IPs, DMA, Logic size, Object tracking

* 학생회원, ** 정회원, 광운대학교 전자통신공학과
(Department of electronics and communication engineering, Kwangwoon University)
※ 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.
© Corresponding Author(E-mail:doydodo@hanmail.net)
접수일자: 2013년1월25일, 수정완료일: 2013년4월19일

Ⅰ. Introduction

최근 디지털 카메라, 웹캠, 모바일 카메라 등의 기능이 다양해짐에 따라 어플리케이션으로 적용시킬 수 있는 객체 추적(object tracking) 알고리즘이 지속적으로 연구 및 개발되어지고 있다. 객체 추적 알고리즘의 종류에는 객체를 여러 점으로 나누어 추적하는 Point tracking, 물체의 생김새와 모양을 가지고 하는 Kernel tracking, 그리고 객체의 밀도, 윤곽등의 정보를 가지고 하는 Silhouette tracking 등이 있다[1]. 이 중 한 점 혹은 여러 점을 추적하는 원리인 Point tracking에는 영상의 회전 빛 크기변화에 강인한 특이점을 추출하는 SURF(Speeded Up Robust Features) 알고리즘이 많이 사용된다. 하지만 SURF 알고리즘은 연산의 복잡도가 높고 연산량이 많아 임베디드 환경에서 실시간 동작을 위해서는 하드웨어 가속기 개발이 필요하다. 특히 어플리케이션을 고려하지 않고 빠른 연산속도에 초점을 맞추어 하드웨어를 설계하면 내부 메모리가 커지고 이로 인한 전체 회로의 크기가 매우 커지는 문제점이 발생하게 된다. 더욱이 하드웨어 IP(Intellectural Propoerty)로 사용하기 어려워지며 임베디드 환경에 적용시키기 위한 ASIC이나 SoC(System on Chip) 시스템으로 개발이 어려워 진다.

따라서 본 논문에서는 회로의 크기를 줄이기 위해 내부 블록 메모리 사용량을 줄이고 외부 메모리와 DMA를 사용하는 방법에 대해 연구하였다. 임베디드 환경에서 적용하기 위한 SURF 알고리즘의 연산속도로 30frame/sec 이상, 효율적인 회로면적을 위해 30nm공정에서 메모리를 포함한 회로의 크기가 60~70만 gate 이하[13]인 하드웨어 가속기를 개발하고 간소화된 객체 추적 알고리즘을 개발하여 적용한 결과를 기술한다.

Ⅱ. Related Work

1. DMA

여기에 영상처리 알고리즘을 하드웨어 가속기로 구현함에 있어 큰 사이즈의 내부 메모리로 인한 회로 크기 증가문제를 줄이기 위해 본 논문에서는 내부 메모리의 사용을 최대한 줄이고 외부 메모리를 사용하는 방법을 제안한다. 이 때 발생되는 하드웨어 가속기와 외부 메모리간의 데이터 전송을 효율적으로 하기 위해 DMA(Direct Memory Access)를 이용하며, 그 구조는 그림 1과 같다[3].

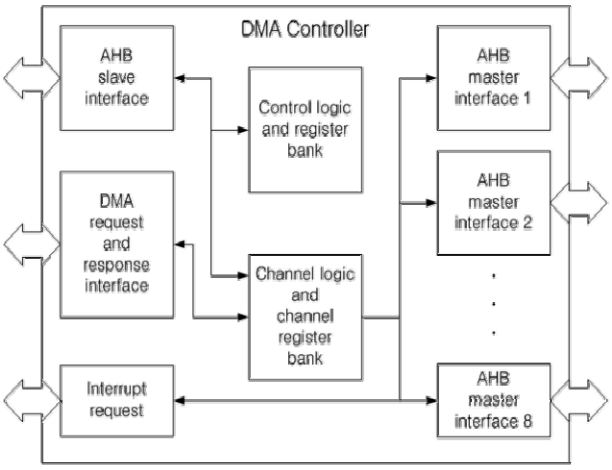


그림 1. DMA 컨트롤러 블록 다이어 그램
Fig. 1. DMA Controller Block diagram.

시스템 버스(System bus)로 AHB-lite에 적용시킬 수 있는 DMA를 사용하였으며 동작 및 최대 데이터 전송크기와 로직의 크기는 아래와 같다.

- Operation mode :
  Independent mode - Outstanding channel mode
- Channel number : Single Channel
- Bus width : 64bit
- FIFO size : 128bytes
- Maximum transfer size : 1024bytes
- Gate-count(45nm at tsmc) : about 14,000
- Maximum operation frequency : 400MHz

사용된 DMA의 크기는 약 14,000 Gate이며, 본 논문에서 제안하는 SURF 기반의 특이점 추출 하드웨어 가속기 보다 회로의 규모가 매우 작기 때문에 DMA로 인한 회로의 크기는 크게 증가하지 않는다.

2. SURF Algorithm

영상의 특이점을 추출하고 서술자를 생성하는 알고리즘들은 여러 논문들에서 소개 되었다. 대표적인 것으로 SIFT(Scale Invariant Feature Transform), MSER(Maximally Stable Extremal Refions), SURF, Harris-Laplace, Hessian-Laplace 등이 있다. 이 중 SURF가 회전 및 크기 변화에 강인한 특이점을 추출하며 연산속도가 가장 빠르기 때문에 임베디드 환경에 적용시키기에 적합하다고 판단된다. SURF 알고리즘의 핵심적인 연산 흐름도는 그림 2와 같다.

SURF 알고리즘은 특이점을 추출하는 Feature

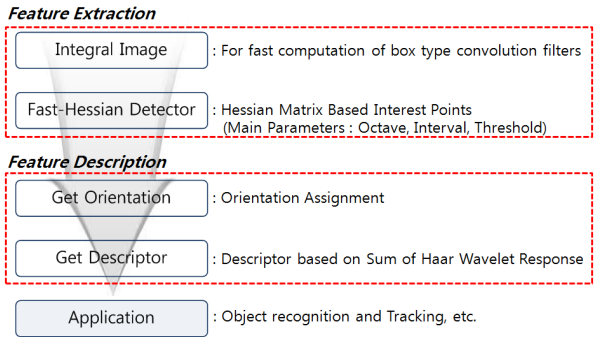


그림 2. SURF 알고리즘 흐름도
Fig. 2. SURF Algorithm Flow.

Extraction 부분과 특이점 주변 영상에 대한 정보를 서술하는 Feature Description 부분으로 나눌 수 있다.

Feature Extraction 부분에서 적분영상을 생성하는 이유는 SURF 알고리즘에서 박스타입의 필터와 컨벌루션 연산을 빨리하기 위함이며, 고속 헤이시안 행렬(Fast-Hessian Matrix)에서는 x, y, xy 축으로 근사화된 2차미분 LoG(Laplacian of Gaussian) 박스필터와 적분영상간의 컨벌루션 연산을 통해 헤이시안 행렬을 구성한 후 행렬의 디터미넌트 값을 계산한다. 그리고 Non-maxima Suppression을 통해 헤이시안 행렬식(Determinant)이 임계값보다 크고 인접 스케일과 비교하여 디터미넌트가 가장 클 경우 특이점으로 결정된다. Feature Description 부분에서는 추출된 특이점의 주방향(dominant direction)을 결정하고 그 주변의 영상의 서술자를 생성한다.

Ⅲ. Algorithm

1. Simplified Object Tracking Algorithm

기존의 개발된 객체 추적 알고리즘들은 대부분 연산의 복잡도와 연산량이 매우 많아 임베디드 환경에서 실시간으로 동작하기에 어려움이 있다. 특히 영상처리 알고리즘에 연산이 집중되지만, 객체 추적 성능을 향상시킬 수 있는 학습 알고리즘도 요구되어지는 메모리가 크고 연산량이 많아 실시간 동작이 어렵다.

본 논문에서는 디지털 카메라, 웹캠 등 임베디드 환경에서 영상의 특이점을 추출하는 SURF 알고리즘 기반의 간소화된 객체 추적 알고리즘을 개발하였으며 알고리즘의 흐름 및 구성은 아래 그림 3과 같다.

그림 3에서 Learning은 Nearest-neighbor ratio 매칭[1]을 통해 추적 물체의 정보를 지속적으로 갱신하여 성능을 향상시키는 학습기 이다. 총 8개의 공간에 학습된

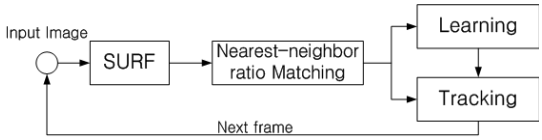


그림 3. 물체 추적 알고리즘 구성도
Fig. 3. Flow of Object Tracking Algorithm.

정보가 저장되며 지정한 객체를 처음부터 연속 5 frame까지 5개의 공간에 객체 정보를 저장하고 이 후 부터는 입력되는 영상을 순차적으로 3개의 공간에 학습하고 지속적으로 최신의 정보로 갱신한다. Tracking은 매칭과 학습된 정보를 이용해 객체를 찾고 다음 영상에서 객체를 찾기 위한 관심영역을 지정한다. 관심영역은 사용자가 지정한 객체 크기의 1.5배 크기이며, 관심영역을 설정하는 이유는 매칭에서 발생되는 연산량을 줄여 속도를 향상시키기 위함이다.

2. SURF for Object tracking

본 논문에서는 SURF 알고리즘을 디지털 카메라 및 웹캠 등에서 객체 추적에 이용할 수 있도록 SURF 알고리즘의 주요 설정값을 조정하였다. 그리고 리피터빌리티(Repeatability), 리콜(Recall), 프리시젼(Precision)을 통해 설정값 변경에 따른 알고리즘의 성능을 검증하였다[4-5].

우선 여러 스케일별로 특이점을 추출하는 Feature Extraction의 고속 헤이시안 행렬식 연산 중 크기변화에 강인한 특이점을 얻기 위해 그림 4와 같이 박스 필터의 크기를 일정 간격으로 변화시키며 스케일별로 헤이시안 행렬식을 구하는 연산에서 박스 필터의 크기, 즉 스케일을 나누는 값을 설정한다.

그림 4와 같이 일정 간격으로 박스 필터의 크기를 변화시키며 스케일별로 헤이시안 행렬식을 구한다. 박스 필터의 크기를 결정하는 수식은 아래와 같으며 *octave*와 *interval* 두 파라미터에 의해 결정된다.

$$Filter\,size = 3(2^{octave} \times interval + 1) \quad (1)$$

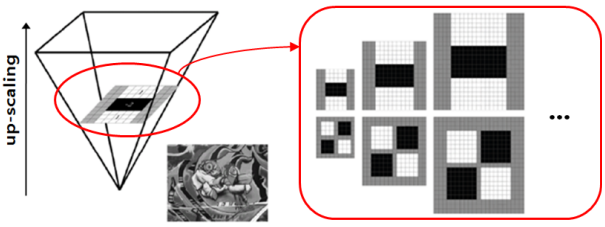
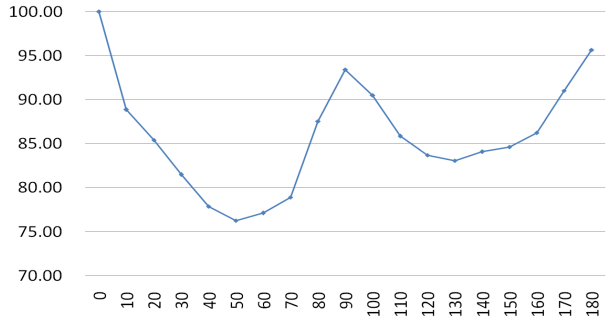
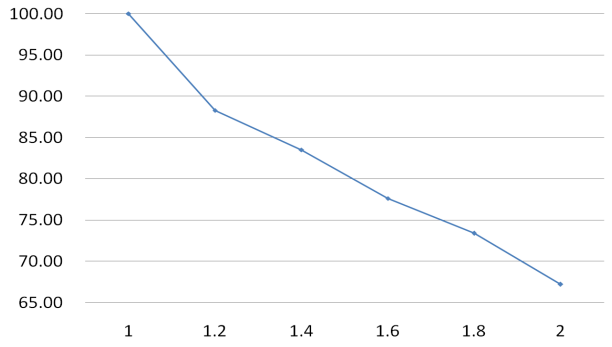


그림 4. 박스필터의 크기 변화
Fig. 4. Up-scaling of approximated LoG filter.



(a) Image rotation



(b) Image up-scaling

그림 5. 영상의 회전과 크기변화에 따른 리피터빌리티
Fig. 5. Repeatability of image rotation and scaling.

본 논문에서 *octave*는 1, *interval*은 4를 기본 설정값으로 사용하였으며, [4]에서 특이점 추출 알고리즘의 성능을 평가하기 위해 사용한 영상들에 대한 리피터빌리티는 그림 5와 같다.

그림 5 (a)는 영상을 10°간격으로 회전했을 때, (b)는 영상을 20%씩 확대했을 때의 리피터빌리티이며 오차범위를 나타내는 입실론(Epsilon)은 6으로 하였다. SURF 알고리즘을 객체 추적에 적용했을 때 추적하고자 하는 객체가 갑자기 변하는 경우가 드물다고 가정하고 30°미만 회전 했을 때의 수치와 영상의 크기가 20%미만 증가했을 때 리피터빌리티를 기준으로 성능을 판단하였다. 따라서 octave와 interval이 1과 4이면 객체추적에 사용이 가능하다고 판단된다.

특이점 추출이 이루어진 후에는 특이점의 주방향을 설정하고 서술자를 생성하는 부분으로 구성된다. 먼저 주방향 설정의 연산과정을 단계별로 설명하면 아래와 같다.

< 특이점의 주방향 설정 >

Step 1 : 특이점으로부터 반경 6s 이내의 적분영상을 읽음(s는 특이점이 추출되는 스케일)

Step 2 : 하알 웨이블릿(Haar-wavelet) 박스필터와 적분영상간의 x축 y축 방향의 컨벌루션 응답인 d_x, d_y를 계산

Step 3 : 각 응답값을 d_x, d_y 벡터공간에 표현

Step 4 : $\frac{\pi}{3}$ 크기인 윈도우를 d_x, d_y 벡터공간에서 일정 슬라이딩(Sliding) 간격으로 회전시키며 윈도우 내에서 응답값의 합이 가장 큰 방향을 주방향으로 결정

특이점의 주방향을 검출한 후 하알 웨이블릿 필터와 원본 영상간의 컨벌루션 연산을 통해 64차원의 서술자를 생성한다.

본 논문에서는 서술자 생성의 연산량을 줄이고 객체추적에 적합한 성능을 찾기 위해 슬라이딩 간격을 달리하여 실험을 했다. 성능을 판단하기 위해 리콜과 프리시젼 수치를 이용하며 슬라이딩 간격을 달리 하였을 때 결과는 아래 그림 6과 같다.

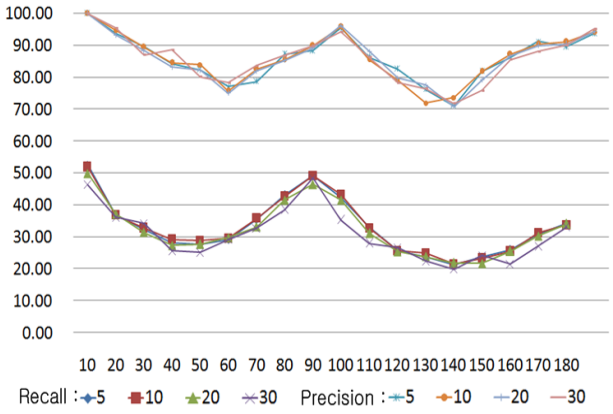


그림 6. 영상의 회전변화에 대한 리콜과 프리시젼
Fig. 6. Recall and Precision of Image-rotation.

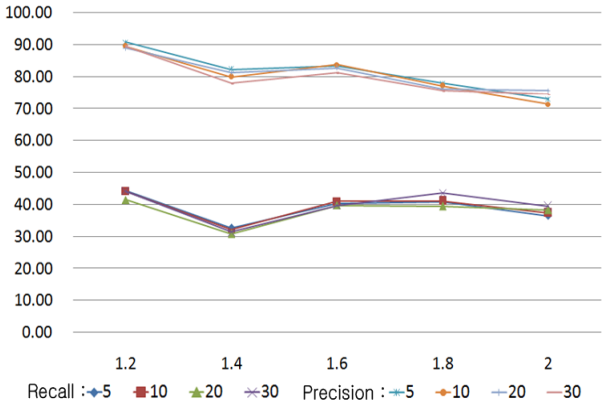


그림 7. 영상의 크기변화에 대한 리콜과 프리시젼
Fig. 7. Recall and Precision of Image-scaling.

그림 6은 영상을 0°부터 180°까지 10°간격으로 회전시켰을 때의 리콜과 프리시젼이다. 슬라이딩 간격은 5°, 10°, 20°, 30° 으로 했으며 5°일때와 30°일 때 수치의 차이는 크지 않다. 그림 7은 영상을 20%간격으로 두 배까지 확대시켰을 때의 리콜과 프리시젼이다. 슬라이딩 간격은 위에서 언급한 바와 동일하며 슬라이딩 간격을 달리 했을 때 수치의 차이는 크지 않다.

따라서 적용하고자 하는 객체 추적에서 SURF의 연산량을 줄이기 위해 필터의 크기를 결정하는 octave와 interval은 각각 1과 4로 설정하며 특이점의 주방향을 설정하는 부분의 슬라이딩 간격은 30°으로 한다.

Ⅳ. Hardware design methodology

하드웨어 가속기를 개발하기 앞서 알고리즘의 어느 부분에 연산이 집중 되는지 그 이유를 정확히 판단해야 한다. 객체 추적을 위한 SURF 알고리즘의 연산량을 파악하기 위해 PC 환경에서(Intel i5 3.3Ghz) 연산시간을 측정한 결과는 아래 표와 같다.

표 1에서 특이점의 개수에 상관없이 특이점을 추출하는 부분의 속도는 일정하다. 하지만 서술자 생성은 특이점의 개수에 따라 연산 속도가 달라지며 표 1의 서술자 생성 연산시간은 특이점이 1,312개 일 때 시간이다. 서술자를 생성하는 부분은 특이점의 개수에 따라 속도가 달라지지만 대체적으로 가장 많은 연산량을 차지한다. 따라서 고속헤이시안 행렬식과 서술자 생성 부분의 연산 고속화가 이루어져야 30 frame/sec 이상의 연산속도로 처리할 수 있다.

헤이시안 행렬식과 서술자 생성의 주된 연산은 여러 스케일의 박스필터와 영상간의 컨벌루션 연산이다. 이러한 경우 하드웨어로 구현할 때 연산속도를 향상시키기 위해 병렬화 설계와 라인 버퍼를 사용하여 반복 연산이 많은 부분의 연산 속도를 빠르게 한다[6]. 컨벌루션 연산을 하는 모듈을 중심으로 라인버퍼가 있도록 하드

표 1. SURF 알고리즘의 주요 연산별 연산시간
Table 1. Execution time of SURF Algorithm.

구 분		Time(ms)
Feature Extraction	Integral Image	1.23
	Fast-Hessian Matrix	26.5
	Non-maxima Suppression	2.95
Descriptor Generation		195.04
Total		225.72

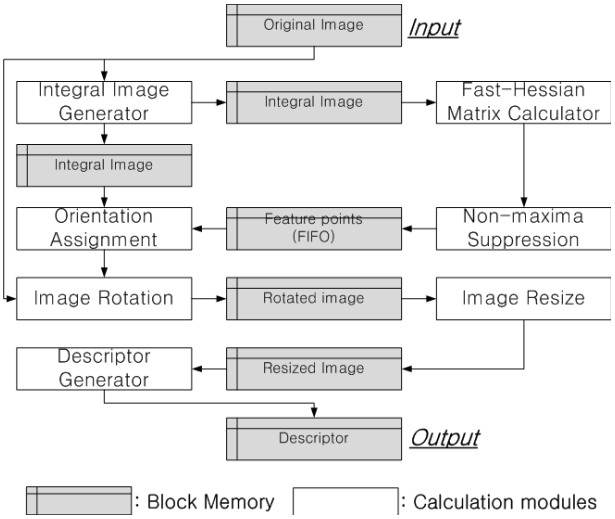


그림 8. SURF 알고리즘의 하드웨어 구조
Fig. 8. Hardware Architecture of SURF Algorithm.

표 2. 임베디드 시스템 환경
Table 2. Embedded System Environments.

구 분	내 용
Embedded Processor	ARM Cortex-M0
System bus	AHB-lite(64bit, 50MHz), APB(64bit, 50MHz)
SDRAM Controller	PL243(200MHz)
DMA	PR201(Provartec DMA)

웨어를 설계했을 때 구조는 그림 8과 같다.

그림 8에 나타낸 하드웨어 구조의 핵심적인 부분은 헤이시안 행렬식과 주방향 설정 연산의 고속화를 위해 적분 영상을 블록 메모리에 저장하여 연산 모듈간의 데이터 전송 지연을 최소화 시키고 컨벌루션 연산이 매 클럭 단위로 이루어질 수 있도록 하는 것이다. 하지만 SURF 알고리즘의 경우 헤이시안 행렬식과 주방향을 설정하는 연산을 할 때 여러 스케일의 박스필터와 컨벌루션 연산을 하기 때문에 연산의 고속화를 위해 요구되어지는 라인 버퍼가 매우 커져 블록 메모리 사용량이 증가하는 문제점이 발생한다.

게다가 적분 영상의 데이터 크기는 일반 영상에 비해 크기 때문에 요구되어지는 블록 메모리가 더욱 커지게 된다. 따라서 내부 블록 메모리의 사용을 최대한 줄이기 위해 아래 표와 같이 정의한 시스템 환경에서 외부 메모리와 DMA를 이용해 설계하는 방법을 이용하였다.

외부 메모리와 DMA를 이용하면 하드웨어 가속기가 하나의 IP로 시스템 버스에 연결되는 것이 아닌 아래 그림과 같이 몇 개의 Sub-IP로 나누어져 시스템 버스에 연결되어진 SoC 구조가 된다.

표 3. Sub-IP들로 구성된 서로 다른 구조
Table 3. The different Architecture consisting of Sub-IPs.

Architecture	'A'	'B'	'C'
The number of Sub-IPs	1	4	6
Set of Sub-IPs	① {Integral Image Generator, Fast-Hessian Matrix Calculator, Nom-maxima Suppression, Orientation Assignment, Image Rotation, Image Resize, Descriptor Generator }	① {Integral Generator} ② {Fast-Hessian Matrix Calculator, Nom-maxima Suppression} ③ {Orientation Assignment} ④ {Image Rotation, Image Resize, Descriptor Generator}	① {Integral Generator} ② {Fast-Hessian Matrix Calculator, Nom-maxima Suppression} ③ {Orientation Assignment} ④ {Image Rotation} ⑤ {Image Resize} ⑥ {Descriptor Generator}

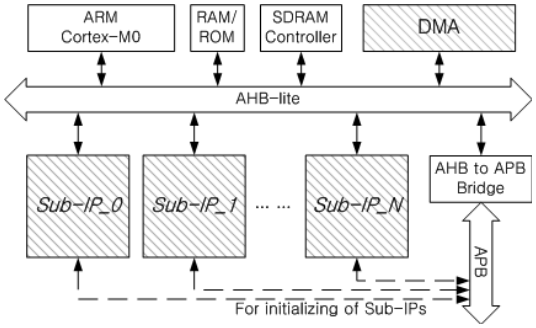


그림 9. Sub-IP들로 구성된 SoC 구조
Fig. 9. SoC Architecture of consisting of Sub-IPs.

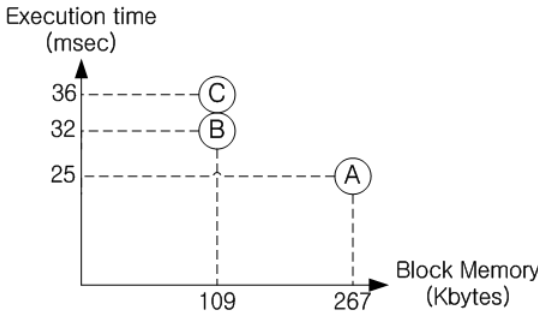


그림 10. 블록 메모리 대비 연산시간 비교 그래프
Fig. 10. Compare of block memory with execution time.

Sub-IP들간의 데이터 전송은 DMA를 사용하여 전송 지연 문제를 줄일 수 있다. 하지만 내부 블록메모리를 줄이기 위해 무조건적인 외부 메모리 사용과 Sub-IP로 나누는 것은 데이터 전송 지연을 매우 증가시키기 때문에 외부메모리를 사용했을 때 내부 메모리 감소 대비 동작 속도 지연이 얼만큼 발생되는지 파악해야 한다.

본 논문에서는 객체 추적에 적합한 연산시간과 회로 면적에 효율적인 하드웨어 설계를 위해 세 가지 다른 구조로 나누어 연산시간과 내부 블록 메모리 크기를 비교해 보았다. 나누는 기준은 그림 8의 블록 메모리들이다. 블록 메모리를 외부 메모리로 대체하게 되면 DMA와 시스템 버스를 통해 데이터 전송을 해야하고 그림 9와 같이 하나의 IP가 아닌 여러개의 Sub-IP로 나뉘게 된다. SURF 알고리즘의 경우 적분영상과 고속 헤이시안 행렬식을 계산하는 모듈사이의 적분 영상 블록 메모리를 외부 메모리로 대체하는 방법과 특이점의 주방향 설정 및 영상을 회전하고 영상의 크기를 변화시키는 모듈 사이의 블록 메모리를 외부 메모리로 대체 하는 방법이 있다. 각 방법을 적용시켰을 때와 적용시키지 않았을 때를 기준으로 서로 다른 세 가지 구조로 나뉘게 된다. 각 구조는 각각 다른 Sub-IP들로 구성되며 나누어진 Sub-IP의 개수와 Sub-IP에 어떠한 연산 모듈이 있는지에 대한 설명은 표 3과 같다.

'A' 구조는 Sub-IP의 개수가 1개이며 모든 연산기가 하나의 IP로 구현되고 시스템 버스에는 단 한 개의 Sub-IP만 연결된다. 'B' 구조는 Sub-IP의 개수가 4개이며 'C' 구조는 총 6개의 IP로 구성되고 그림 9처럼 여러개의 Sub-IP가 시스템 버스에 연결된 SoC 구조가 된다. A, B, C 세 가지 구조에 대해서 VGA(640x480) 영상에서 특이점이 1,312개일 때 각 구조별 연산속도와 블록 메모리 사용량은 그림 10과 같다.

하나의 IP로 구현되는 'A' 구조의 경우 연산속도가 25ms로 가장 빠르지만 블록 메모리의 사용량은 약 267Kbytes로 다른 두 가지 구조에 비해 두 배 이상 크다. 본 논문에서 어플리케이션에 적용하기에 적합한 속도로 지정한 30frame/sec 이상의 속도로 A와 B 구조가 만족하며 이 중 면적에 가장 효율적인 설계를 위해서는 메모리가 가장 작은 B 구조가 적합하다.

Ⅴ. SoC Architecture

본 논문에서는 객체 추적을 위한 면적에 효율적인 하

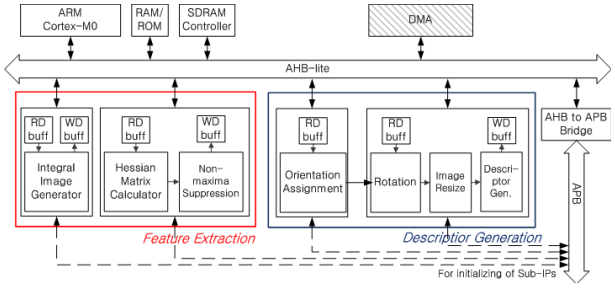
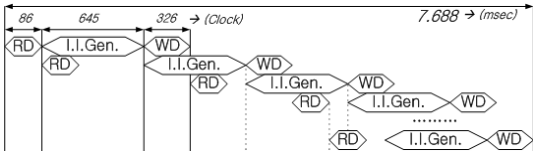
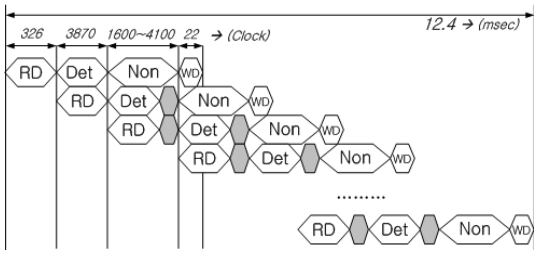


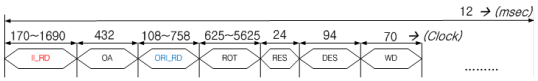
그림 11. SURF 알고리즘의 SoC 구조
Fig. 11. SoC Architecture of SURF Algorithm.



(a) Integral Image Generator



(b) Fast-Hessian Detector



(c) Feature point Description

그림 12. 각 Sub-IP별 타이밍
Fig. 12. Timing flow of each Sub-IPs.

드웨어 가속기 설계를 목표로 함에 따라 하드웨어는 'B' 구조인 총 4개의 Sub-IP들도 구성된 구조로 설계하였다. Sub-IP들을 시스템 환경에 적용시켰을 때 구조는 그림 11과 같다.

그림 11의 구조에서 특이점 좌표를 추출하는 Feature Extraction부분은 적분 영상을 생성하는 Integral Image Generator와 고속 헤이시안 검출기(Fast-Hessian Detector) 총 두 개의 Sub-IP로 나뉜다. 고속 헤이시안 검출기에는 헤이시안 행렬식을 계산하는 모듈과 Non-maxima suppression 모듈로 구성된다.

그리고 특이점의 주방향을 검출하고 서술자를 생성하는 Descriptor Generation 부분은 주방향을 검출하는 Orientation Assignment IP와 영상을 회전하고 서술자를 생성하는 IP 두 개의 Sub-IP로 구성된다. 따라서 SURF 알고리즘은 총 4개의 Sub-IP로 구성된다. 위 구조로 설계했을 때 각 Sub-IP들의 타이밍도는 아래 그림 12과 같다.

그림 12의 타이밍도에 나타낸 모든 RD와 WD는 DMA를 이용하여 데이터를 읽고 쓰는 동작이며 세부적인 설명은 다음과 같다.

(a) Integral Image Generator

- RD : SDRAM으로부터 영상 한 줄(640pixel) 읽음
- I.I.Gen. : 적분 영상 생성
- WD : 생성된 적분 영상을 SDRAM에 저장

(b) Fast-Hessian Detector

- RD : SDRAM으로부터 적분 영상 한 줄(640pixel) 읽음
- Det : 스케일별 헤이시안 행렬의 디터미넌트 계산
- Non : 디터미넌트를 가지고 스케일별 특이점 추출
- WD : 스케일별 특이점 좌표를 SDRAM에 저장

(c) Feature Pont Description

- II_RD : SDRAM으로부터 적분영상을 읽음
- OA : 특이점의 주방향을 연산
- ORI_RD : SDRAM으로부터 원본 영상을 읽음
- ROT : 주방향만큼 영상을 x축으로 회전
- RES : 서술자 생성을 위해 21x21로 영상을 Resize
- DES : Haar 필터를 이용해 64차원의 서술자 생성
- WD : 생성된 서술자를 SDRAM에 저장

적분 영상과 고속 헤이시안 검출기 모두 버스의 활용을 높이기 위해 데이터를 읽어오는 것과 연산 그리고 다시 데이터를 외부 메모리에 쓰는 것을 중첩되도록 하였다. 모든 Sub-IP의 동작속도는 200MHz이며 VGA(640x480) 영상의 적분 영상 생성 연산시간은 7.688ms 이고, 고속 헤이시안 검출기는 12.4ms이다. 특이점을 추출하는 것은 총 20.088ms 소요된다. 그리고 고속 헤이시안 검출기에서 발생되는 메모리의 크기를 줄이기 위해 모듈로 연산방법[7-8]을 통해 적분영상 한 화소의 데이터 크기를 27bit에서 17bit으로 줄였다.

서술자를 생성하는 Sub-IP들도 200MHz에서 동작하며 [5]에서 서술자의 성능을 평가하기 위해 사용한 영상중 가장 많은 특이점인 1,312개를 가지는 그래피티(Graffiti) 영상에서 서술자를 생성하는데 걸리는 시간은 약 12ms 이다. 따라서 SURF 알고리즘의 총 연산시간은 VGA(640x480) 해상도에서 약 32ms로 31frame/sec

표 4. 사용된 블록 메모리 크기
Table 4. Block memory size of each Sub-IPs.

Sub-IPs		Memory size (Kbyte)
Feature Point Detection	Integral Image Generator	3.9
	Fast-Hessian Detector	58.44
Feature Point Description	Orientation	6.98
	Descriptor	11.41
Total		80.73

의 속도로 특이점을 추출하고 서술자를 생성한다. 각 Sub-IP별 사용한 블록 메모리는 표 4와 같다.

특이점을 검출하는 부분이 전체 80Kbytes 중 약 60Kbytes 만큼이며 비중이 가장 높다. 고속 헤이시안 검출기에서 외부 메모리로부터 적분 영상을 읽어올 때 빠른 컨벌루션 연산을 위해 최소한의 라인 버퍼가 필요하기 때문이다. 서술자를 생성하는 부분도 컨벌루션 연산이 있지만 헤이시안 행렬식 보다 연산 횟수가 적어 필요로 하는 라인 버퍼의 크기가 작다. 서술자 생성의 블록 메모리는 약 18Kbyte로 특이점을 검출하는 부분에 비하면 작은 크기이다.

Ⅵ. Result and Analysis

1. Computational reduction of a SURF

본 논문에서는 SURF 알고리즘을 객체 추적에 적합한 성능을 지니도록 주요 설정값들을 조정하였다. 주방향을 설정하는 연산에서 슬라이딩 간격을 그림 6과 그림 7처럼 달리하여 서술자를 생성할 때 연산시간 감소량은 아래 표 5와 같다.

5°간격일 경우 윈도우의 360°회전을 위해 72번의 반복문이 수행되며 30°일 경우 12번이 수행된다. 반복문은 특이점 개수에 비례하여 증가하기 때문에 특이점이 많이 나오는 영상일수록 연산량의 감소효과는 더욱 크다. 슬라이딩 간격을 30° 초과로 설정하면 특이점의 패턴이 비슷한 영상, 예를 들어 사람의 얼굴이나 인형 등에서는 리콜과 프리시젼이 영상의 회전이나 크기변화가 있을 때 큰 폭으로 변한다. 비슷한 패턴이 많은 영상에서는 특이점의 주방향이 비슷하고 또한 서술자도 비슷한 경우가 많아 잘못된 매칭의 수가 증가하기 때문이다.

표 5. 슬라이딩 간격에 따른 반복문 횟수
Table 5. The loop count in accordance with sliding interval.

Sliding step	5°	10°	20°	30°
루프 횟수	72	36	18	12

2. Hardware Design

기존의 논문들에서 제안된 SURF 알고리즘 기반의 특이점 추출기 및 서술자 생성 하드웨어 가속기는 대부분 연산의 속도에 초점을 두어 사용하는 메모리의 크기가 큰 경우가 많다. 본 논문에서 제안하는 하드웨어 가속기는 나노급 공정에서 회로의 크기를 알기 위해 로직과 메모리를 30nm급 공정에서 합성한 결과 로직 사이즈가 약 20만 게이트, 메모리 80Kbytes가 약 54만 게이트로 전체 회로의 크기는 약 74만 게이트였다. 다른 논문들에서 제안한 SURF 기반의 특이점 추출 및 서술자 생성 하드웨어 가속기들의 성능을 비교한 결과는 표 6과 같다.

[9-11]의 경우 본 논문과는 달리 SURF 알고리즘 전체를 하드웨어로 구현하지 않았다. [9-10]은 특이점을 추출하는 부분만 하드웨어로 구현했으며 서술자 생성은 소프트웨어로 설계하였고, [11]은 특이점 추출 및 주방향 검출까지 하드웨어로 설계하고 서술자 생성의 나머지 연산은 소프트웨어로 구현하였다. 본 논문에서 구현한 하드웨어 가속기는 특이점 추출과 서술자 생성 모두 하드웨어 가속기로 구현되었으며 회로의 전체 크기중 약 80%가 특이점을 추출하는 부분이다.

[9]의 어플리케이션은 네비게이션 시스템이며 특이점

표 6. 구현 사례와 연산 속도 및 회로 크기 비교
Table 6. Comparison of execution time and area.

구 분		[9]	[10]	[11]	본논문
입력영상 크기		1,024 x768	640 x480	640 x480	640 x480
속 도 (ms)	특이점 추출	100	18	7.55	20
	서술자 생성	x	x	10.45[1)]	12
	합계	100	18	18	32
동작주파수(MHz)		200	100	200	200
Memory size(Kbyte)		193	268	1,310	81
Gate-count[2)] (단위:Kgate)		**1,280**	**1,780**	**8,730**	**740**

1) Orientation Assignment만 하드웨어로 구현됨.
2) [9-11]은 본 논문에서 제안한 구조의 30nm 공정에서 logic gate 수를 기준으로 예측한 결과임

추출의 연산속도는 100ms 이고 회로의 크기는 약 1,280Kgate이다. [11]의 경우는 특이점 추출 및 주방향 설정의 연산속도가 18ms로 가장 빠르지만 사용하는 블록메모리의 크기와 회로의 크기가 8,730Kgate로 제일 크다. 본 논문에서 제안한 구조는 객체 추적 시스템에 적용하기 위한 SURF 알고리즘으로 VGA(640x480) 영상에서 연산속도는 31frame/sec 이며, 회로의 크기는 약 740Kgate로 가장 작다. 표 6의 각 논문들에서 제안하는 하드웨어 가속기의 어플리케이션은 다르지만 연산의 고속화에 초점을 맞추어 설계한 것보다 본 논문에서 설계한 하드웨어의 크기가 10배 이상 작으며, 다른 논문들과도 두 배 이상 차이가 나는 것을 확인할 수 있다. 특히 본 논문에서는 외부 메모리와 DMA를 사용하여 내부 블록 메모리의 사용량을 줄임으로써 회로의 크기를 줄일 수 있었다. 실제 SoC 시스템으로 One-chip화 한다면 DMA가 추가되어 회로의 크기가 커질 수 있지만 DMA의 크기는 14,000 게이트로 SURF 알고리즘 하드웨어 가속기의 크기에 비하면 작은 크기로 회로의 전체 크기에 큰 영향을 주지 않는다.

본 논문에서 제시하는 Sub-IP들과 DMA로 이루어진 하드웨어 가속기 설계는 회로의 면적 감소효과뿐만 아니라 개발 시간의 단축 및 디버깅 등을 간편화 시킬 수 있는 장점도 있다. 복잡한 영상처리 알고리즘을 하나의 IP로 설계하게 되면 컨트롤러가 매우 복잡해져 개발 시간을 증가 시키고 디버깅의 어려움을 증가시키게 된다. 따라서 효율적인 하드웨어 설계를 위해 SURF 알고리즘을 표 2처럼 여러 구조로 설계하는 방법에 대해 연구하고 블록 메모리의 사용량에 따른 연산속도가 어떻게 되는지 분석한 것처럼, 유사한 다른 영상처리 알고리즘들에도 적용시켜 어플리케이션에 적합한 연산속도와 회로의 크기가 효율적인 구조로 설계할 수 있을 것으로 기대된다.

3. Experimental Result

본 논문에서 설계한 SURF 알고리즘 하드웨어 가속기를 간소화된 객체 추적 알고리즘에 적용시켰을 경우 성능은 그림 13과 같다. 추적 대상인 객체는 카드상자이며 입력영상 VGA(640x480)해상도에서 100x50 크기를 가지고 특이점은 약 80~100개를 가진다. 그림 13의 검정색 박스는 객체를 찾았을 때 표시해 주는 박스이며 흰색 박스는 입력 영상에서 대상을 찾기 위해 지정된 관심영역을 나타낸다. 관심영역의 크기는 추적 대상 크기의 1.5배로 설정하였다. 그림 13 (a1)부터 (a3)까지는 학습기의 장점을 보여주는 사례로서 카드가 옆으로 회전했을 때와 뒷모습을 보일 때 모두 추적이 됨을 보여주고 있다. (a4)는 다른 장애물에 의해 객체가 가려졌을 때 'X'표시와 함께 추적이 실패했음을 보여주고 (a5)는 객체가 다시 나타났을 때 학습된 데이터에 의해 성공적으로 추적함을 보여준다. (b1)부터 (b4)는 객체가 1 프레임 간격으로 90°회전하거나 180°반전 되었을 때에도 추적이 성공함을 보여준다.

추적하고자 하는 객체가 크지 않고 객체를 찾기 위한 영역도 영상의 전체 크기에 비해 작기 때문에 연산속도는 약 21ms로 32ms 보다 9ms 빨라지게 된다. 또한 워스트의 경우인 객체를 잃어 버렸을 경우 전체영상에 대해서 객체를 찾지 않고 VGA(640x480)영상을 320x480으로 두 번에 나누어 객체를 찾는다면 요구되어 지는 블록 메모리를 더욱 줄일 수 있다. 고속 헤이시안 검출

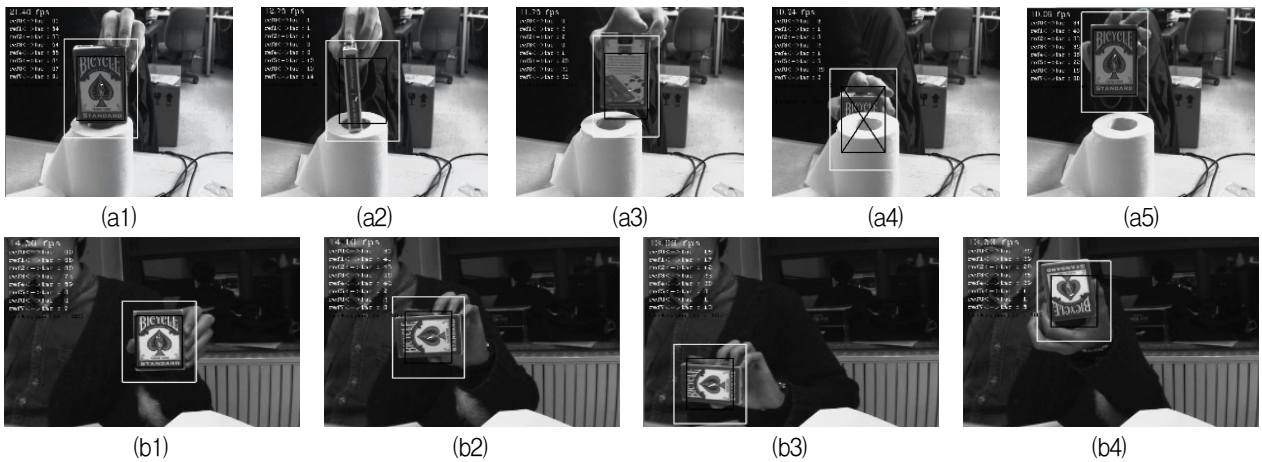


그림 13. 카드상자 추적 시나리오
Fig. 13. Scenario of tracking a card box.

기에서 요구되어 지는 블록 메모리를 28Kbytes로 줄일 수 있으며 전체 메모리가 80Kbytes에서 약 52Kbyte가 된다. 전체 회로의 크기는 74Kgate에서 55Kgate로 줄어들게 된다. 객체를 잃어버린 워스트의 경우 연산속도가 두 배로 증가하지만 본 논문에서 적용하고자 하는 디지털 카메라, 웹캠 등의 어플리케이션에서 성능 저하에 큰 영향을 미치지 않는다.

하지만 간소화된 객체 추적 알고리즘의 임베디드 환경에서 실시간 동작을 위해서는 SURF 알고리즘 뿐만 아니라 매칭 알고리즘의 하드웨어 가속기 개발 혹은 소프트웨어 알고리즘의 개선이 필요하다. 그림 13의 객체처럼 특이점이 약 80~100개 이고 관심영역의 특이점이 약 200개 일 때 ARM Cortex-M0 1GHz에서의 매칭 연산시간은 200ms이다. Nearest-neighbor ratio 매칭 알고리즘은 64차원의 서술자간의 유클리디안 거리를 구하는 단순한 연산이지만 특이점의 개수가 증가함에 따라 연산량도 증가하여 특이점이 많아질수록 연산속도가 매우 느려지는 단점이 발생한다. 객체 추적에 적합한 매칭 알고리즘의 최적화 방안에 대한 연구의 일환으로 하드웨어 가속기 개발 및 GPU 등을 활용한 연산속도 개선이 필요하다[12].

Ⅶ. Conclusion

본 논문에서는 회로의 크기에 효율적인 하드웨어 설계를 위해 외부메모리와 DMA를 활용하는 방법에 대해 연구하였고 SURF 알고리즘에 적용했을 때 회로의 크기를 감소시킬 수 있음을 기술하였다. 하지만 지나치게 많은 외부 메모리와 DMA의 사용은 데이터 전송 지연을 발생시키기 때문에 타겟 어플리케이션에 적합한 연산속도와 회로의 크기를 알기 위해서 표 2처럼 하드웨어 구조를 달리 했을 때 회로의 크기 감소 대비 속도가 어느 정도 되는지 연구하는 것이 중요하다. 특히 SURF 알고리즘의 경우 크기가 다른 여러 개의 박스필터와 반복되는 컨벌루션 연산이 많고 필요로 하는 메모리가 중복되는 부분이 많기 때문에 크기를 줄이는데 있어 많은 효율을 볼 수 있었다.

본 논문에서 구현한 SURF 알고리즘 외에 다른 영상처리 알고리즘들도 대부분 연산이 복잡하고 연산량이 많아 실시간 동작을 위해서 하드웨어 가속기 개발을 필요로 한다. 본론에서 하드웨어 구현 방법에 대해 기술한 것처럼 다른 영상처리 알고리즘들도 연산을 분석하여 응용 프로그램에 적합하도록 효율적인 설계를 한다면 개발 시간 단축 및 디버깅의 어려움을 줄여 하드웨어 IP의 신뢰도를 향상시킬 수 있을 것으로 기대된다.

참 고 문 헌

[1] Alper Yilmaz, Omar Javed, Mubarak Shah, “Object tracking : A Survey”, ACM Computing Surveys, Vol.38, No.4, Article 13, December 2006.
[2] H.Bay, T.Tuvtelaars, and L.Van Gool, “Surf : Speeded up robust features”, Computer Vision - ECCVI, Vol. 3951, pp. 404-417,, 2006
[3] http://www.provartec.com.ipproducts/57, “PR201ConfigurableDual-coreHighPerformanceAHBDMAReferenceGuide”,Revision1.5
[4] Cordelia Schmid, Roger Mohr, Christian Bauckhage, “Evaluation of Interest Point Detectors”, International Journal of Computer Vision, Vol. 37, Issue 2, pp. 151-172, June 2000.
[5] Krystian Mikolajczyk, Cordelia Schmid, “A Performance Evaluation of Local Descriptors”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 27, Issue. 10, pp. 1615-1630, October 2005.
[6] Takashi Saegusa, Tsutomu Maruyama and Yoshiki Yamaguchi, “HoW fast in an FPGA in image processing?”, International Conference on Filed Programmable Logic and Applications, pp. 77-82, September 2008
[7] Belt, H.J.W., “Word length reduction for the integral image”, 15th IEEE international conference on Image Processing, pp. 805-808, 2008
[8] Ehsan, S., McDonald-Maier, K.D., “Exploring Integral Image Word Length Reduction Techniques for SURF Detector”, '09 Second International Conference on Computer and Electrical Engineering, Vol. 1, pp. 635-639, 2009.
[9] Jan Svab, Tomas Krajnnik, Jan Faigl and Libor Preucil. “FPGA Based Speeded Up Robust Features”. In IEEE International Conference on Technologies for Practical Robot Applications, pp. 35-41, 2009.
[10] Jae-Kyung Ryu, Su-Hyun Lee, Yong-Jin Jeong, “FPGA Design of a SURF-based Feature Extractor”, Journal of Korea Multimedia Society, Vo.14, No.3, pp. 368-377, March 2011.
[11] Bouris, D, Nikitakis, A and Papaefstathiou, I. “Fast and Efficient FPGA-based Feature Detection employing the SURF algorithm”. 18th IEEE Annual International Symposium on Field-Programmable Custom Computing

Machines, pp. 3-10, 2010.
[12] Marius Muja, David G..Lowe, “Fast Approximate Nearest neighbors with automatic algorithm configuration”, International conference on Computer Vision Theory and Applications, pp. 331-340, 2009.
[13] Personal communication with industry, 2012.

저 자 소 개

도 용 식(학생회원)
2011년 광운대학교 전자통신공학과 학사 졸업
2011년~현재 광운대학교 전자통신공학과 석사 과정
<주관심분야 : SoC 설계, 영상처리 및 인식, 임베디드 시스템>

정 용 진(정회원)
1983년 서울대학교 제어계측공학과 학사 졸업
1983년 3월~1989년 8월 한국전자통신연구원
1995년 미국 UMASS 전자전산공학과 석사 및 박사 졸업
1995년 4월~1999년 2월 삼성전자 반도체 수석연구원
1999년 3월~현재 광운대학교 전자통신공학과 정교수
<주관심분야 : 무선통신, 정보보호, SoC 설계, 영상처리 및 인식, 임베디드 시스템>