파노라마 이미지 생성시간을 단축하기 위한 멀티코어 환경에서 특징점 추출 병렬화

(Parallelizing Feature Point Extraction in the Multi-Core Environment for Reducing Panorama Image Generation Time)

김 건 호 † 최 태 호 ††
(GeonHo Kim) (Taiho Choi)

정 희 진 †† 권 범 준 ††
(Heejin Chung) (Bomjun Kwon)

요 약 본 논문에서는 멀티코어 환경에서 파노라마 이미지 생성 시간을 단축시키기 위해 특징점 추출 알고리즘을 병렬화한다. 여러 장의 사진들을 합성하여 파노라마 이미지를 만드는 과정에는 사진들 간의 겹치는 영역을 찾아내기 위해 각 사진의 특징점을 추출하는 단계가 필요하다. 계산량이 많은 특징점 추출 단계를 빠르게 수행하기 위해 비대칭 멀티 프로세서 아키텍처인 CBE(Cell Broadband Engine)를 사용하여 특징점 추출 병렬 알고리즘을 개발하고, 성능이 얼마나 향상되는지 실험하였다. 실험 결과, 본 논문에서 개발한 병렬 알고리즘은 프로세서 수에 비례하여 성능이 높아지는 선형 확장성의 특징을 보였다. 이처럼 멀티코어 환경에서 이미지 프로세싱 작업 수행 시에 어떻게 하면 높은 성능의 좋은 결과를 낼 수 있는지 알아본다.

키워드 : 멀티코어, 특징점 추출, 파노라마 이미지

Abstract In this paper, we parallelized a feature point extraction algorithm to reduce panorama image generation time in multi-core environment. While we compose a panorama image with several images, the step to extract feature points of each picture is needed to find overlapped region of pictures. To perform rapidly feature extraction stage which requires much calculation, we developed a parallel algorithm to extract feature points and examined the performance using CBE(Cell Broadband Engine) which is asymmetric multi-core architecture. As a result of the exam, the algorithm we proposed has a property of linear scalability - the performance is increased in proportion the number of processors utilized. In this paper, we will suggest how Image processing operation can make high performance result in multi-core environment.

Key words : Multi-core, Feature extraction, Panorama Image

1. 서 론

파노라마 이미지는 광각의 시각을 제공하는 사진, 그림으로서 특수한 장치를 이용해야 만들 수 있는 이미지이다. 하지만 최근 디지털 사진이 대중화 되면서 항공사진, 건물내부사진, 관광지 정보등을 나타내기 위해 여러 장의 사진을 이어 붙여서 파노라마 이미지를 만드는 방법이 많이 사용되고 있다.

파노라마 이미지 생성 시간은 처리할 이미지의 수와 이미지 해상도에 의존적이다. 그래서 생성 시간을 빠르게 하기 위해서는 하드웨어의 성능이 뒷받침 되어야 한다. 하지만 단일 프로세서 구조의 하드웨어로 만족할만한 성능을 달성하는 데에는 한계가 있다. 그래서 최근 이러한 한계를 극복하기 위해 멀티코어 구조의 하드웨어가 보편화 되고 있다. 계산량이 많은 이미지 프로세싱 작업을 멀티코어 환경에 맞게 병렬화 하게 되면 단일 프로세서 환경에서 수행 하는 것 보다 성능을 월등히 높일 수 있다.

본 논문에서는 멀티코어 환경에서 병렬화를 어떻게 해야 최적의 성능을 낼 수 있는지 알아본다. 파노라마 이미지 생성 알고리즘을 분석하여 계산량이 많은 부분을 파악하고, 그 부분에 대해 병렬화 함으로써 얼마만큼의 성능 향상이 되는지 본다. 또한 알고리즘을 병렬화할 때 어떤 것을 고려해야 하는지도 논의한다.

본 논문의 구성은 다음과 같다. 2절에서는 파노라마 생성 알고리즘에 대해 살펴본다. 3절에서는 병렬화를 수행할 멀티코어 환경에 대해 설명한다. 4절에서는 특징점 추출 알고리즘의 병렬화 방법을 기술한다. 5절에서는 병렬 알고리즘의 구현과 실험 결과를 기술하였다. 그리고 마지막 결론에서는 병렬화 시 고려사항과 향후 활용 분야에 대해 기술하였다.

· 이 논문은 제34회 추계학술대회에서 '파노라마 이미지 생성시간 단축을 위해 멀티코어 환경에서 특징점 추출 병렬화 하기'의 제목으로 발표된 논문을 확장한 것임

† 정 회 원 : 삼성전자 소프트웨어연구소
gh007.kim@samsung.com
†† 비 회 원 : 삼성전자 소프트웨어연구소
taiho.choi@samsung.com
heejin.r.chung@samsung.com
bomjun@samsung.com

논문접수 : 2007년 12월 6일
심사완료 : 2008년 3월 5일

정보과학회논문지: 컴퓨팅의 실제 및 레터 제14권 제3호(2008.5)

2. 파노라마 이미지 생성 방법

파노라마 이미지를 생성하기 위한 개략적인 작업 순서는 다음과 같다[1].

1. 각 이미지들의 특징점을 추출한다.
2. 각 이미지들의 특징점을 비교하여, 이미지들 간에 대응되는 점들이 있는지 탐색한다.
3. 이 특징점 대응 정보를 토대로 이미지 스티칭(stitching) 단계를 통해 이미지의 중첩되는 부분을 서로 연결시켜 준다.
4. 이미지가 연결되는 부분에 대해 블렌딩(blending) 작업을 수행한다.

특징점 생성과 매칭단계에서는 옥타브 기법을 사용한다[2]. 옥타브 기법은 한 장의 이미지를 크기를 조정하여 여러 장을 만든 다음 처리하는 기법이다. 한 장의 이미지에서 여러 장의 확대 또는 축소된 이미지가 나오기 때문에 처리시에 많은 시간이 소요된다. 스티칭단계는 소스 좌표에 대해 보간(interpolation)하는 과정에서 많은 시간이 소요된다. 블렌딩 단계에서는 중첩되는 이미지의 마스크를 생성하여 픽셀들을 혼합하는 과정을 수행하기 때문에 서로 비교하기 위한 이미지와 마스크를 유지하기 위해 많은 양의 메모리 공간이 필요하다[3].

3. 개발 환경

병렬화 작업을 수행하기 위한 멀티코어 기반의 개발 환경으로 STI(Sony, Toshiba, IBM) 컨소시엄에서 개발, 생산하고 있는 셀 브로드밴드 엔진(Cell Broadband Engine, CBE)을 사용했다.

CBE는 그림 1과 같이 한 개의 PPE(Power Processor Element)와 여덟 개의 SPE(Synergistic Processing Element), 총 아홉 개의 프로세서를 가지고 있다. SPE들은 자체적으로 데이타 저장을 위한 지역 메모리를 가지고 독립적으로 작업을 수행한다. PPE는 작업의 전체적인 스케줄링과 분배 데이타 관리 등을 수행하는 역할을 주로 맡는다. SPE는 SIMD 연산이 가능한 프로세서로 빠른 데이타 처리가 가능하다[4].

PPE와 SPE를 이용한 병렬화 방법은 다음과 같다.

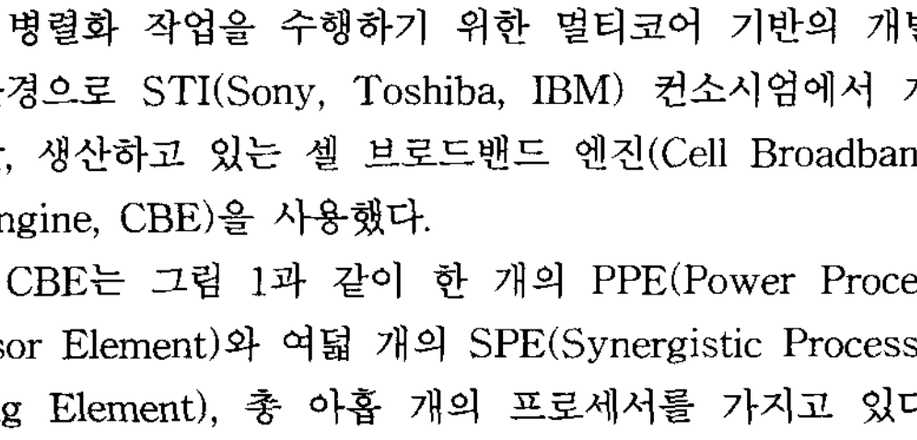

그림 1 셀 브로드밴드 엔진 구조

1. SPE에 전달하기 위해 처리할 데이타를 분배한다.
2. 최적의 성능을 낼 수 있는 스케줄링을 이용하여 데이타를 SPE에 보낸다.
3. SPE에서는 받은 데이타를 처리한다. SPE에서는 SIMD 방식을 이용하여 데이타를 빠르게 처리할 수 있다.
4. SPE를 효율적으로 사용하기 위해서 더블 버퍼링을 이용하여 PPE와 SPE간에 데이타를 전송한다.

처리할 데이타를 병렬화 하면 데이타는 PPE에 의해 분할되어 SPE에 할당되어, SPE에서 데이타를 SIMD를 이용하여 처리할 수 있다.

4. 성능 향상을 위한 알고리즘 병렬화 작업

4.1 병렬화 전략

알고리즘을 병렬화하기 위해 우선 살펴보아야 할 것이 어떻게 병렬화를 해야 하는지 결정하는 것이다. 순차적인 알고리즘을 병렬화하기 위한 기본 전략으로 크게 두 가지가 있다. 하나는 태스크 병렬화이고, 다른 하나는 데이타 병렬화이다. 하나의 작업을 동시에 수행 가능한 여러 개의 부 작업으로 나누는 것이 가능하다면 태스크 병렬화가 용이하고, 처리할 데이타를 여러 개로 나누어 독립적으로 처리할 수 있다면 데이타 병렬화가 바람직하다. 이제 특징점 추출 알고리즘을 분석하여 병렬화를 수행할 것이다. 특징점 추출 과정은 이미지들 간의 비교에 반드시 필요한 단계이고 수행 시간도 오래 걸린다. 그래서 특징점 추출단계를 병렬화하여 성능향상이 얼마만큼 되는지 알아볼 것이다.

4.2 특징점 추출

본 논문에서는 특징점을 추출하는 알고리즘으로서 SIFT(Scale Invariant Feature Transform) 알고리즘[2]을 사용하였다. SIFT 알고리즘은 다음과 같다.

1. 입력 이미지를 블러링을 하여 옥타브를 생성하고 DoG 이미지를 생성한다.
2. 이미지 사이즈를 1/2, 1/4, 1/8로 줄여 가면서 과정 1을 반복하여 DoG 이미지 피라미드를 생성한다.
3. 과정 2에서 생성한 DoG 이미지들의 피라미드로부터 특징점을 추출한다.
4. 입력 이미지를 이용하여 그래디언트 맵을 생성하고 각 특징점 주변의 인접 16×16픽셀들을 이용하여 해당 특징점의 디스크립터를 생성한다.

그림 2는 전체 단계의 흐름을 나타내고 각 단계에서 괄호 안의 숫자는 해당 단계에서 걸리는 시간으로, 전체 시간을 100%로 봤을 때 해당 과정을 처리하는데 소요되는 시간을 의미한다.

4.2.1 이미지 블러

특징점을 추출하는 알고리즘의 첫 번째 단계는 입력 이미지로부터 여러 옥타브를 가지는 이미지 피라미드를

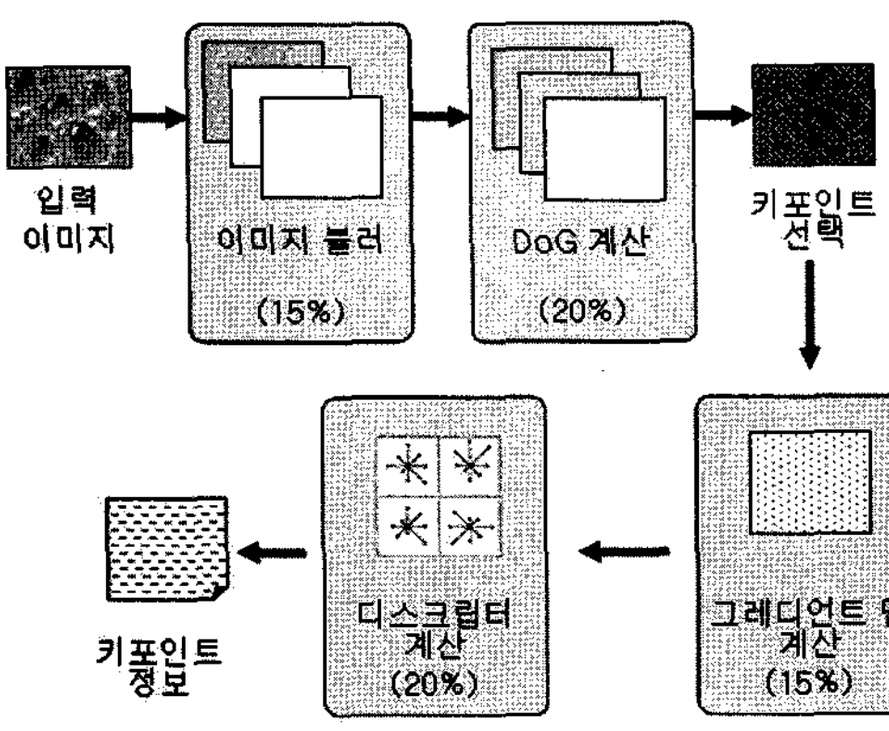

그림 2 특징점 추출 과정

생성하는 것이다. 여러 개의 옥타브는 입력이미지에 여러 개의 서로 다른 블러 커널을 적용하여 생성한다. 입력 이미지에서 데이타를 읽을 때는 순차적으로 읽어서 이미지 피라미드를 생성한다. 순차적으로 라인 단위로 데이타를 읽기 때문에 라인 단위 처리가 가능하다. 그러므로 SPE에 데이타를 보낼 때 라인 별로 데이타를 적절히 나누어서 SPE에 데이타를 보내고 처리한다면 처리 속도에 향상을 가져올 것이다. PPE에서 SPE로 데이타를 전송하기 위해서는 데이타 분배 작업이 필요하다. SPE의 메모리의 크기가 256KB이므로 한번에 데이타를 보낼 때 256KB만큼 데이타를 보내면 전송 시간을 최소화할 수 있지만 그러기 위해서 PPE에서는 분배 데이타를 만들기 위한 복잡한 연산을 해야 하고 SPE는 유휴 상태로 있게 된다. 이런 것을 방지하기 위해 PPE에서는 분배 데이타를 간단하게 만들어 바로 전송하기 위해 라인 별로 데이타를 전송하는 방식을 사용한다. 전체적인 알고리즘은 다음과 같다.

1. SPE의 수에 맞춰 이미지를 균등하게 분할한다.
2. 각각의 SPE에 블러 마스크와 해당 이미지 영역의 첫 번째 줄의 주소를 전달한다.
3. SPE는 첫 번째 줄을 LS로 로드하여 블러링을 수행한다.
4. 2~3의 과정을 해당 영역의 모든 줄에 대해 수행한다.
5. 이미지를 90도 회전 후 1~4의 과정을 다시 한번 수행하고 -90도 회전하여 원상 복구한다. 연속적인 데이타를 SPE에 보내면 SPE에서도 데이타를 접근하기 용이하기 때문에 이미지를 회전시켜 데이타를 구성한다.

블러 알고리즘 병렬화 작업을 위해 다음과 같이 기존 알고리즘을 수정하였다. 기존의 블러 알고리즘은 블러 커널이 각 픽셀마다 이동을 하면서 결과 값을 계산 하는 것이다.

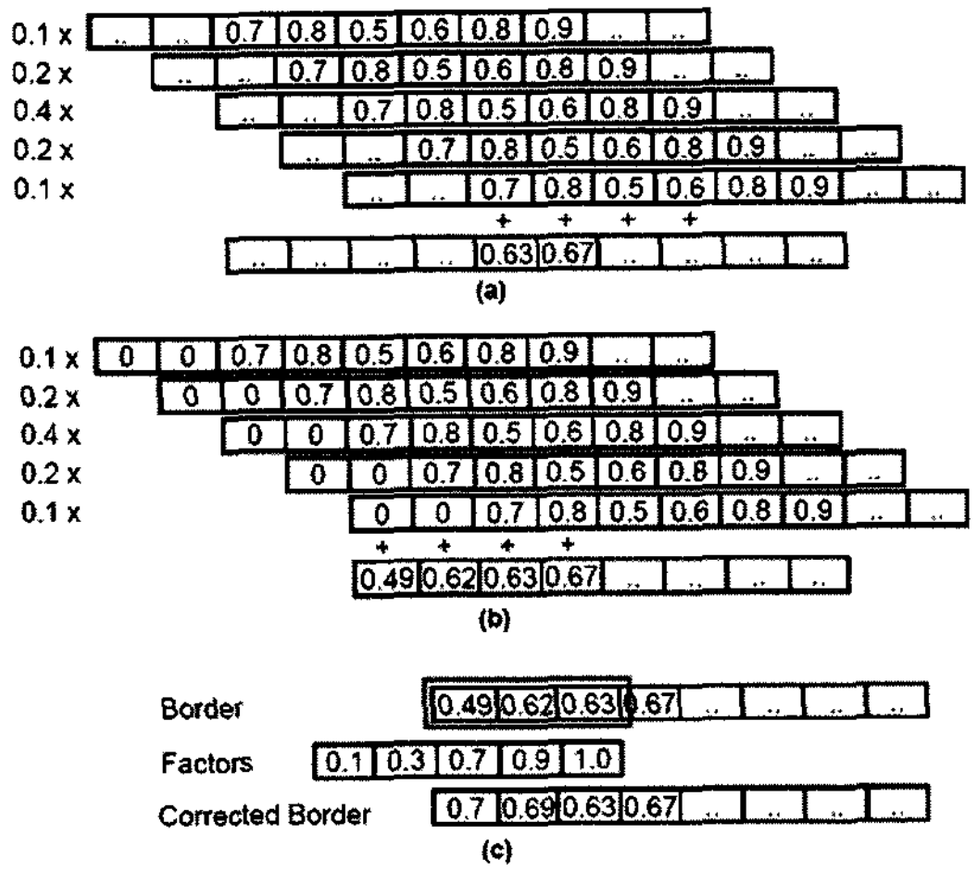

그림 3 수정된 이미지 블러 과정

그림 3(a)의 변형된 블러 알고리즘은 블러 커널의 각 항으로 이미지 전체를 곱하여 블러 커널의 오프셋만큼 이동 시켜 결과배열에 더함으로써 블러를 수행한다. 기존의 블러 알고리즘에 비해 루프의 분기를 개선하여 SIMD 활용이 쉽도록 수정한다. SIMD는 연속적인 데이타를 처리하는데 적절하기 때문에 루프의 분기 수정으로 연속적인 데이타를 한꺼번에 처리할 수 있도록 만드는 것이다. 그림 3(b)에서 보듯이 이미지의 경계부분에 대해 따로 처리를 해주지 않으면 경계부분의 색이 어두워 지는 현상이 발생한다. 이때 경계부분을 체크하기 위해 조건문이 사용되는데 이러한 조건문을 없애는 것도 블러 알고리즘을 향상 시킨다. 본 논문에서는 다음과 같은 방법으로 조건문을 제거하였다.

우선 경계부분을 위한 가중치를 계산한다. 이미지의 앞,뒤에 계산을 위한 영역을 생성하고 블러를 수행한다.

이후 그림 3(c)의 붉은색 박스부분이 경계부분으로 실제 이미지보다 어두워진 부분이다. 그림 3(c)에서처럼 계산한 가중치를 이용하여 경계 영역을 나누어 준다. 이렇게 함으로써 어두워진 경계영역을 보정할 수 있다.

가중치를 계산하는 방법은 목적에 따라 다를 수 있으나 본 논문에서는 블러 시에 이미지 영역의 바깥을 계산하는 블러 커널의 합을 뺀 값을 이용하였다.

4.2.2 DoG 계산

DoG(Difference Of Gaussians)는 특징점의 후보값을 찾기 위한 것으로, 각 옥타브마다 다른 가우시안 필터가 적용된 이미지를 읽어온다[5]. 그리고 읽어온 이미지들의 픽셀들의 차를 구한다. DoG 단계에서는 대부분의 시간을 이미지들 간의 차를 구하는데 사용한다. 그렇기 때문에 이미지 데이타를 분할하여 차를 구하는 작업을 병렬화 하면 된다.

SPE에 작업을 할당하기 위해서는 SPE를 몇 개를 사용하는지에 따라 다르게 할 수 있을 것이다. 만약 SPE의 개수가 다섯 개로 정해져 있고, 이들을 모두 사용할 수 있다면 각 옥타브에서 다섯 번의 DoG를 수행하는 작업을 각각 SPE에 하나씩 할당하면 되지만, 활용 가능한 SPE의 수가 항상 유동적인 것을 감안해서 DoG 병렬화를 설계해야 한다. 그래서 여기서는 PPE의 이미지를 불러올 때 SPE의 수에 맞춰 이미지를 골고루 분할하여 할당하고, SPE에서 차를 구한 뒤 결과값을 저장하고 다음 이미지를 처리하는 방식을 택했다.

4.2.3 그레디언트 맵 계산

가우시안 블러가 적용된 이미지들 간의 차를 구한 후에는 이 DoG 이미지들에 대해 그레디언트 값을 계산하게 된다. 이미지의 (x,y) 픽셀에 대한 그레디언트 값을 구하는 식은 아래와 같다.

$$m(x,y)=\sqrt{(L(x+1,y)-L(x-1,y))^2+(L(x,y+1)-L(x,y-1))^2}$$

$$\theta(x,y)=\tan^{-1}((L(x,y+1)-L(x,y-1))/(L(x+1,y)-L(x-1,y)))$$

위 수식에서 알 수 있듯이 한 픽셀에 대한 그레디언트 값을 구하기 위해서는 상하좌우 4 픽셀의 값이 필요하다. 따라서 특정 라인의 픽셀 값들은 대부분의 경우 최대 세 개 라인의 그레디언트 값을 계산하는데 사용된다. 여러 개의 SPE에서 특정 라인에 대한 픽셀 값을 중복해서 가져오지 않도록 하기 위해 각 프로세서에 라인 단위로 작업을 할당 할 때 한 라인이 아니라 M개의 라인씩 할당한다. 또, 프로세서의 활용도를 높이기 위해 무조건 같은 수의 라인들을 할당하는 것이 아니라, 각 SPE에서 할당 받은 M개의 라인에 대한 그레디언트 값 계산이 완료되면 새로운 라인을 할당 받아 계산을 수행하도록 한다.

4.2.4 디스크립터 계산

특징점은 이미지의 종류에 따라 위치와 수가 변한다. 그러므로 각 SPE에 균등하게 할당하기 위해 라운드 로빈 방식으로 특징점을 할당하였다. 또한 디스크립터를 계산하는 알고리즘은 특징점의 주변 16×16픽셀의 그래디언트 맵을 필요로 한다.

5. 병렬화 알고리즘의 구현 및 결과

본 절에서는 2절에서 기술한 특징점 추출 병렬 알고리즘을 구현하고 실험한 결과를 설명한다. 하드웨어 환경은 CBE가 장착되어 있는 소니 플레이스테이션 3을 이용하고 운영체제는 YDL(yellow dog linux)를 설치하였다. 구현된 특징점 추출 프로그램은 오픈 소스인 Autopano-SIFT-C 소프트웨어를 이용한다[6]. Autopano-SIFT-C는 SIFT알고리즘을 구현한 소프트웨어로 위에서 언급한 알고리즘을 충실히 수행하고 있다. 기존에 있는 소스코드에서 병렬화 할 처리부분은 SPE에서 처리하도록 따로 소스코드를 분리시키고, PPE는 처리할 이미지 데이타를 분리하고 DMA를 통해 SPE에 데이타를 전달하도록 한다. 여기서 중요한 것은 PPE에서 SPE로의 DMA 전송시간과, SPE에서 전송 받은 데이타의 처리시간을 잘 조절해야 한다. SPE가 데이타를 처리하는 동안 그 다음에 처리할 데이타를 DMA를 통해서 SPE에 전송하고 있어야 하며 SPE가 어떤 데이타를 처리했을 때 그 다음 데이타를 바로 처리할 수 있도록 만들어야 한다. 병렬화 구현 시 가장 중요한 것 중 하나가 SPE들을 기다리게 하면 안 되는 것이다. 가장 높은 성능을 달성하기 위해서는 모든 프로세서가 쉬지 않고 작업을 수행하고 있어야 하기 때문이다.

각 단계에 대한 알고리즘을 구현한 후 각 단계를 수행하여 얼마만큼의 성능 향상을 보이는 지 실제 데이타를 산출하였다. 병렬화 전 후의 시스템 성능 변화에 대해 표 1에 정리하였다. 여기서 사용된 이미지는 1600×1200의 해상도, 435KB 크기의 이미지였다.

표 1(a)는 각 알고리즘 단계에서 병렬화를 수행한 부분에 대한 성능 향상 결과를 보여준다. 여섯 개의 SPE를 사용하여 PPE를 한 개 사용할 때보다 대략 3~4배의 속도 향상을 확인할 수 있다. 표 1(b)는 병렬화 작업 후 전체 특징점 추출 성능이 얼마나 향상되었는지에 대한 결과이다. 실제로 구현된 알고리즘에는 메모리 전송 등 병렬화가 어려운 부분이 존재하기 때문에 단계별로 병렬화가 진행된 부분만 갖고 측정한 결과에 비해서는 조금 낮은 성능향상을 보인다.

그림 4는 활용 가능한 SPE 수를 변경하여 성능을 시험한 결과를 보여준다. 활성화되는 SPE의 수에 비례해서 처리 능력이 선형적으로 향상되는 것을 볼 수 있다. 이런 결과는 병렬 알고리즘의 성능으로서 가장 바람직한 결과로 볼 수 있다. DoG 병렬화의 경우 다른 단계

표 1 병렬화 전후의 시스템 성능 변화

(a)

수행단계	PPE 단독	PPE + 6 SPEs	성능 향상율
이미지 블러	1.44초	0.35초	406%
DoG	0.06초	0.01초	446%
그레디언트	1.26초	0.51초	245%
디스크립터	1.30초	0.44초	294%

(b)

테스트 대상	PPE 단독	PPE + 6 SPEs	성능 향상율
테스트케이스1	6.22초	2.00초	310%
테스트케이스2	6.11초	1.97초	310%
테스트케이스3	6.52초	2.14초	304%
테스트케이스4	6.15초	2.04초	301%
테스트케이스5	7.05초	2.21초	318%

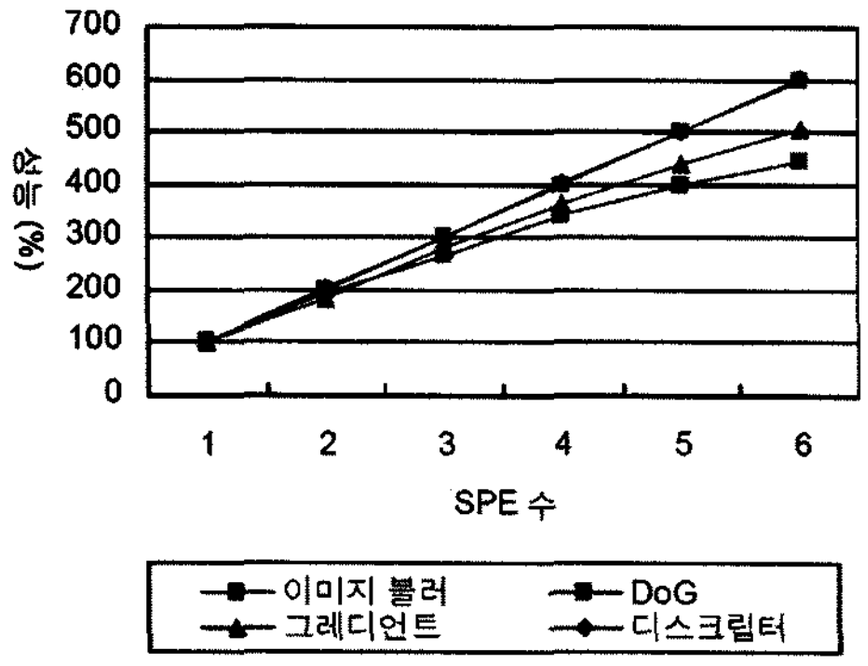

그림 4 SPE 수에 따른 단계별 성능

에 비해 성능이 향상되는 정도가 낮게 관찰되었다. 병렬화의 기본 방식은 PPE가 SPE에서 처리할 데이타를 분배하여 SPE에 데이타를 전송하면 SPE에서는 해당 데이타를 처리하는 것이다. 하지만 SPE에서 데이타를 처리하는 시간보다 PPE에서 데이타를 분해하고 전송하는 시간이 더 긴 경우가 있다. SPE에서 SIMD 등을 이용하여 데이타를 더 빨리 처리하게 된다면 이러한 증상은 더욱 두드러지게 나타날 수 있다.

그림 5는 병렬화된 알고리즘을 이용하여 이미지의 특징점을 산출한 결과를 보여주고 있다.

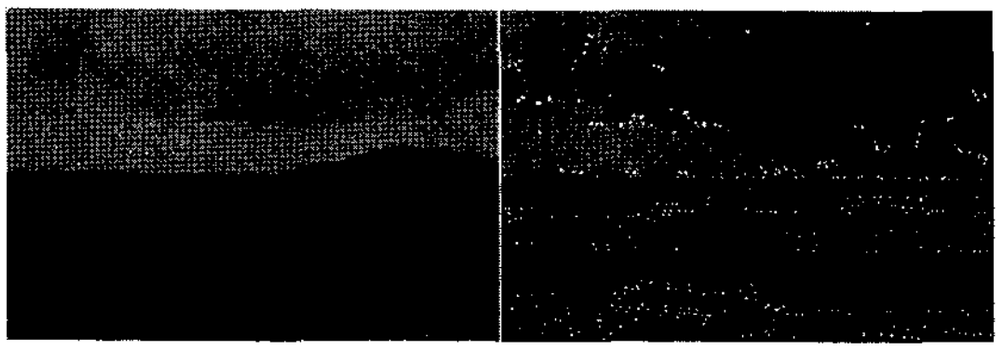

그림 5 특징점 추출 결과 이미지

6. 알고리즘 병렬화 시 고려사항

데이타 병렬화 시에 가장 좋은 성능을 내는 상황을 살펴 보면 특정 데이타에 대한 계산을 수행할 때 해당 데이타와 인접한 데이타만 필요한 경우이다. 이럴 경우 SPE에서 처리할 데이타와 인접 데이타만을 보내면 되기 때문에 데이타 분배가 용이하고 데이타 분배 스케줄링 또한 용이하다. 이미지 처리 관련 알고리즘을 병렬화 할 때는 먼저 알고리즘 분석 후 처리할 데이타가 어떤 것인지 확인하고, SPE에 데이타를 어떤 식으로 전달해야 하는 지 결정하는 것이 가장 먼저 이루어 져야 한다. 복잡한 자료구조, 예를 들어 트리 구조와 같은 자료구조를 가진 데이타들의 경우에는 자료구조의 크기가 방대한 경우 하나의 SPE에 처리하기에는 데이타의 크기가 너무 크다. 그럴 경우 분할 가능한 새로운 자료구조를 만들어 데이타를 처리하는 방식을 사용하는 것이 효율적일 수 있다. 이미 구성된 자료구조를 분할하여 각 SPE에 할당하는 작업은 복잡하고 시간도 오래 걸리는 작업이기 때문이다.

7. 결 론

이상의 병렬화 과정을 통해 멀티코어 환경에서의 병렬화를 위해 고려해야 할 여러 가지 사항에 대해 살펴보았다. 파노라마 이미지 생성 알고리즘의 경우 데이타 병렬화만 이루어져 있기 때문에 데이타 병렬화만을 이슈로 다루었지만, 더 나은 성능을 위해 태스크 병렬화를 같이 병행하는 작업도 고려할 수 있다. 이 논문에서 다루고 있는 특징점 추출 알고리즘 병렬화 작업은 스테레오 매칭, 물체인식, 3차원 재구성 등 이미지 처리에 관련된 다른 분야에도 적용이 가능하다. 특징점 추출에 대한 병렬화 작업은 파노라마 이미지 생성뿐만 아니라 다른 이미지 프로세싱 작업의 성능 향상에도 도움을 가져올 수 있을 것이다. 특히 동영상의 실시간 처리와 같은, 성능 요구사항이 중요시되는 응용에서는 이러한 방식의 병렬화가 필수적이다.

참 고 문 헌

[1] S. Richard. Image Alignment and Stitching : A Tutorial, Microsoft Research, September 27, 2004.

[2] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision, Volume 60, Number 2, pp. 91-110, Springer, 2004.

[3] P. J. Burt and E.H. Adelson. A Multiresolution Spline with Application to Image Mosaics. ACM transactions on Graphics, 2(4), 217-236, 1983.

[4] IBM. Cell Broadband Engine Programming Handbook, Version 1.0. April 19, 2006.

[5] D. G. Lowe. Object Recognition from Local Scale-Invariant Features. In Seventh International Conference on Computer Vision, Greece, pp. 1150-1157, 1999.

[6] S. Nowozin. autopano-sift: making panoramas fun. http://user.cs.tu-berlin.de/~nowozin/autopano-sift/