JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지(J. Korea Inst. Inf. Commun. Eng.) Vol. 20, No. 9 : 1778~1784 Sep. 2016

멀티코어를 이용한 차선 검출 병렬화 시스템 설계

이효찬[1] · 문대철[1*] · 박인학[2] · 허강[1]

Design of Parallel Processing of Lane Detection System Based on Multi-core Processor

Hyo-Chan Lee[1] · Dai-Tchul Moon[1*] · In-hag Park[2] · Kang Heo[1]

[1*]Department of Information and Communication Engineering, Hoseo University, Asan 31499, Korea
[2]System Centroid Inc.

요 약

본 논문에서는 차선 검출 알고리즘에 병렬처리를 적용하여 성능을 개선하였다. 차선 검출은 지능형 보조 시스템으로써 자동차가 차선을 이탈하면 경보음 또는 핸들을 보정해줌으로써 운전자를 돕는 보조 시스템이다. 병렬 처리 알고리즘 중 데이터 레벨 병렬처리는 설계가 간단하지만 병목현상이 발생하는 문제가 있다. 제안하는 고속 데이터 레벨 병렬처리 알고리즘은 병목현상을 줄여 성능이 향상되었다. 실제 블랙박스 도로 영상을 도입하여 알고리즘을 측정한 결과 싱글 코어 경우 약 30 Frames/sec의 성능을 얻었다. 병렬처리를 적용한 결과로써 옥타코어 기준으로 데이터 레벨인 경우 약 100 Frames/sec의 성능을, 고속 데이터 레벨인 경우는 약 150 Frames/sec의 성능을 얻을 수 있다.

ABSTRACT

we improved the performance by parallelizing lane detection algorithms. Lane detection, as a intellectual assisting system, helps drivers make an alarm sound or revise the handle in response of lane departure. Four kinds of algorithms are implemented in order as following, Gaussian filtering algorithm so as to remove the interferences, gray conversion algorithm to simplify images, sobel edge detection algorithm to find out the regions of lanes, and hough transform algorithm to detect straight lines. Among parallelized methods, the data level parallelism algorithm is easy to design, yet still problem with the bottleneck. The high-speed data level parallelism is suggested to reduce this bottleneck, which resulted in noticeable performance improvement. In the result of applying actual road video of black-box on our parallel algorithm, the measurement, in the case of single-core, is approximately 30 Frames/sec. Furthermore, in the case of octa-core parallelism, the data level performance is approximately 100 Frames/sec and the highest performance comes close to 150 Frames/sec.

키워드 : 멀티코어, 병렬 처리, 차선 검출, 데이터 레벨 병렬화

Key word : Multi-core, Parallel Processing, Lane Detection, Data Level Parallelism

Received 11 July 2016, **Revised** 14 July 2016, **Accepted** 17 July 2016

* **Corresponding Author** Dai-Tchul Moon (E-mail:dtmoon@hoseo.edu, Tel:+82-41-540-5683)
Department of Information and Communication Engineering, Hoseo University, Asan 31499, Korea

Open Access http://dx.doi.org/10.6109/jkiice.2016.20.9.1778 print ISSN: 2234-4772 online ISSN: 2288-4165

Ⅰ. 서 론

ADAS(Advanced Driver Assistance System)는 운전자를 돕는 지능형 운전자 보조 시스템이다. ADAS의 기술 중에는 차선 이탈 경보, 전방 추돌 경보, 앞차 출발 알림, 하이 빔 제어, 교통 표지판 인식, 교통 신호등 인식 , 보행자 인식, 차선 유지 제어 등 다양한 기술을 지원함으로써 스마트카의 실용화를 가능하게 한다. 특히 차선 검출은 운전자가 차선을 무의식적으로 넘고 있다고 판단되면 핸들을 조정하여 차로를 벗어나지 않도록 제어함으로써 운전자의 생명을 보호한다. 교통 표지판 인식, 교통 신호등 인식, 보행자 인식, 차선 검출 기술은 레이더 센서나 초음파 센서로 차선을 인지하지 못해 대부분 카메라 센서로 검출한다. 스마트카에 다양한 기능들이 지원될수록 프로세서의 성능이 좋아야 하고 이에 대응하여 멀티코어 프로세서를 대안으로 사용하고 있다. 그러나 단순히 멀티 프로세서를 사용한다고 좋은 성능을 출력하지는 않는다. 프로그램 내부에서 역할 분담(병렬화)가 제대로 되지 않는다면 성능 향상에 한계가 생기기 때문이다. 따라서 영상처리에서의 효율적인 병렬화 알고리즘은 연구의 필요성이 존재한다.

컴퓨터 비전 기술로 처리되는 차선 검출 알고리즘은 잡음 제거, 외곽선 검출, 직선 검출 등 많은 연산량이 필요하다. 카메라에서 획득한 많은 영상들을 빠르게 처리하는 방법은 알고리즘을 개선해야 한다. 더욱 효율적인 방법은 멀티 코어 프로세서를 통해 알고리즘을 병렬화하여 효율적으로 처리하는 방법이다. 성능을 효율적으로 향상시키기 위해 다수의 프로세서를 사용하는 병렬화 기법은 잡음 제거, 외곽선 검출 같은 필터들은 데이터 레벨 병렬화 알고리즘을 직선 검출은 쓰레드 레벨 병렬화 알고리즘을 주로 사용하게 된다. 그러나 데이터 레벨 병렬화는 설계가 간단하지만 처리하는 일이 많을수록 병목현상이 발생하는 단점이 있다. 본 논문에서는 데이터 레벨 병렬화의 단점을 최소화하여 향상된 고속 데이터 레벨 병렬화 기법을 제안하였다.

Ⅱ. 차선검출 시스템 개발

차선 검출에 대한 전체적인 알고리즘 흐름은 그림 1과 같다.

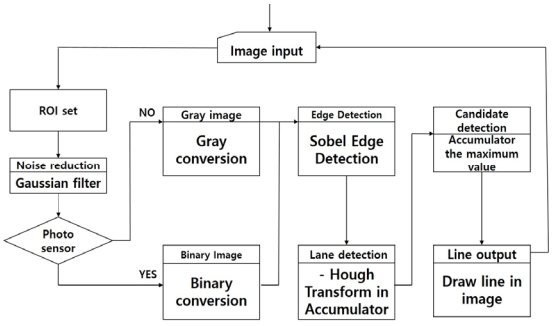

Fig. 1 Lane detection flow chart

2.1. Gaussian Smoothing Filter

잡음을 제거하려는 방법으로 가우시안 스무딩 필터 또는 메디안 필터 등이 있다. 가우시안 필터는 영상의 잡음을 제거할 때 사용한다. 동작이 빠른 편은 아니지만 스무딩시 가장 유용한 필터이다. 스무딩 필터는 주변 색상을 흐릿하게 하여 노이즈를 제거한다. 가우시안 마스크를 구한 후에 전체영상 픽셀에 대해 회선 연산을 수행함으로써 픽셀의 값을 정할 수 있다.

본 논문에서는 표본화된 3x3 가우시안 마스크를 사용하였다. 마스크의 특징은 행렬 총합을 역수를 취해 마스크에 곱하면 그 합이 1이 된다. 5x5, 7x7 마스크도 있는데 마스크 크기가 커질수록 스무딩 효과가 크게 나타나게 된다. 그림 2 와 그림 3은 가우시안 필터에 사용된 마스크와 결과 사진이다.

1/16	1/8	1/16
1/8	1/4	1/8
1/16	1/8	1/16

1	2	1
2	4	2
1	2	1

Fig. 2 3x3 Gaussian mask

2.2. Sobel Edge Detection

본 논문에서 사용된 소벨 외곽 검출은 영상의 외곽선을 검출하는 기법으로써 영상의 명도 변화량을 검출한다. 외곽검출은 영상의 명암도를 기준으로 명암의 변화가 큰 지점이다. 따라서 이러한 명암, 밝기 변화율은 기울기를 검출할 수 있다. 소벨 외곽 검출 마스크를 이루는 숫자 특징은 합이 0이며 전체 영상 픽셀에 대해 수직 마스크와 수평 마스크를 각각 회선 연산을 수행하고 합연산을 하면 차선의 외곽을 나타내는 데이터를 얻어 영

(a) (b)

Fig. 3 Gaussian filter image (a) Original image (b) Image applied Gaussian filter

상에 출력할 수 있다. 소벨 마스크의 특징은 모든 방향의 에지를 추출하고 잡음에 대체로 강하며 수직, 수평 방향성 보다 대각선 방향 외곽선에 더 민감하다. 그림 4와 그림 5는 3x3 소벨 수직, 수평 마스크와 결과 사진이다[1,2].

-1	0	1
-2	0	2
-1	0	1

(a)

-1	-2	1
0	0	0
1	2	1

(b)

Fig. 4 (a) Vertical mask & (b) Horizontal mask

Fig. 5 Sobel edge detection

2.3. Hough Transform

그림 6과 같이 영상에서 (x,y) 좌표 공간의 픽셀들은 (r,Θ) 매개변수 공간에서 곡선의 형태로 나타난다. r은 원점에서 직선까지의 수직 거리를 의미하고 Θ는 원점에서 직선에 수직선을 그렸을 때 y축과 이루는 각도를 의미한다. 또한, 좌표 공간에서 같은 직선 상에 존재하는 픽셀들의 경우 매개변수 공간에서 교점을 가지게 된다. 허프 변환 기법은 이러한 특징을 이용하여 영상의 픽셀들을 좌표 공간에서 매개변수 공간으로 사상시킨 후, 누적 과정을 통해 교점을 찾아 직선 성분을 추출하는 방법이다[3,4]. 일반적인 허프 변환 기법에서는 좌표 공간의 픽셀들을 매개변수 공간으로 사상시키기 위해 식 (1)을 사용한다. 그리고 이를 통해 구해진 값을 누산기 배열에 저장하며, 가장 많은 교점을 가지고 있는 지점이 직선 성분일 가능성이 뚜렷한 지점이다. 누산기배열의 크기는 r-Θ를 갖는다. (r,Θ) 매개변수 공간에서 영상의 좌표공간으로 식 (2)와 같이 사상시키면 그림 7과 같이 차선의 라인을 검출할 수 있다.

$$r = x\cos\theta + y\sin\theta \tag{1}$$

$$x = \frac{r - y\sin\theta}{\cos\theta} \quad y = \frac{r - x\cos\theta}{\sin\theta} \tag{2}$$

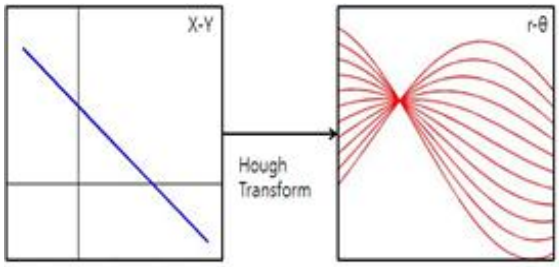

Fig. 6 Hough transform (r,θ) parameter space

Fig. 7 Lane detection result with Hough transform

2.4. 차선검출 시스템 개발 환경

차선검출 알고리즘을 구현하기 위한 개발환경은 Windows에서 Visual Studio C++로 개발되었다. 컴퓨터 비전 소프트웨어는 오픈소스 라이브러리를 이용하면 더욱 쉽게 구현할 수 있다. 필요한 함수를 찾아 시스템에 알맞게 추가하여 구현하면 되기 때문이다. 하지만 오픈 소스를 이용하면 이미 캡슐화가 이루어진 객체를 분해하기 쉽지 않아 자유자재로 코드를 재구성 하기가 쉽지 않다.

빠른 성능을 갖는 병렬 알고리즘의 실험을 위해 차선검출에 사용되는 필터들을 분해하고 코딩하여 실험해야 하므로 단점이 될 수 있다. 또 플랫폼 간에 이식성은 떨어지는 단점도 있다. 예를 들면 임베디드 플랫폼에 오픈소스를 지원하지 않는다면 다시 새로 구현해야 하기 때문이다. 따라서 본 논문에서는 차선검출 소프트웨어 이식성을 높이기 위해 오픈 소스를 사용하지 않고 C 코드로 직접 코딩하였다.

Ⅲ. 멀티코어 기반의 병렬화 차선검출

3.1. OpenMP 라이브러리

병렬 처리를 구현하기 위해 쓰레드(thread)를 할당하여 구현 할 수 있지만 추가 작업이 많다. OpenMP 라이브러리를 사용하면 #pragma 전처리기만 이용하면 기존 코드를 거의 변경하지 않고 쉽게 병렬화 할 수 있다. Visual Studio에서도 OpenMP를 지원하며 공유 메모리 다중 처리를 위한 프로그래밍 API이다[5]. 만약 컴파일러에서 OpenMP를 지원하지 않는 다면 #pragma 전처리기를 무시하고 진행하며 에러메시지를 발생시키지 않는다. 즉 싱글 코어 기반의 프로그래밍과 동일하다. 앞서 구현된 차선검출 코드는 크게 4가지 필터를 거치게 된다. 자주 사용되는 필터 위주로 병렬화를 설계하면 된다. #pragma 전처리기에 다양한 옵션을 줄 수 있어 여러 가지 이벤트에 대해 처리할 수 있다. 특히 공유 변수를 사용할 때 뮤텍스와 같은 접근 제한 기능을 사용하지 않으면 데이터의 신뢰성이 떨어지게 된다. 이러한 이벤트에 대해서도 처리가 가능하다. 그러나 오픈 소스 기반 사용 시 사용자가 원하는 상세한 부분을 구현하기는 까다로워 쓰레드를 할당하여 사용하는 것이 편리할 수도 있다.

본 논문에서는 OpenMP를 사용하여 차선 검출 병렬화를 검증하였다. OpenMP 라이브러리를 사용하는 이유는 pThread 기반의 경우 태스크 병렬화에 주력하는 반면 OpenMP는 데이터 병렬화를 염두에 두고 있어 기존의 작성된 프로그램을 거의 수정하지 않고 구현할 수 있기 때문에 이식성이 높아진다[6].

3.2. 데이터 레벨 기반의 차선검출 설계

병렬 처리 영역은 데이터 사이의 종속성이 존재하지 않아 병렬로 동시에 처리가 가능한 영역을 말한다. 병렬 처리 기법 중 데이터 레벨 병렬 처리는 배열의 요소에서 동일한 작업이 동시에 수행되는 시나리오를 말한다[7]. 설계가 복잡하지 않아 병렬처리 알고리즘에 자주 사용된다.

본 논문에서 제안하는 알고리즘과 성능을 비교하기 위해 차선검출 소프트웨어를 데이터 레벨 병렬 처리로 구현하였다. 각 코어는 그림 8과 같이 픽셀 영역을 독립적으로 처리하게 된다. 차선 검출 알고리즘은 가우시안 필터, 회색 조 변환, 외곽선 검출, 허프 변환 순서대로 수행하게 되는데 이들의 공통점은 같은 크기의 반복문을 수행한다는 점이다. 따라서 이미지를 가로로 분할하고 각 코어는 필터들을 차례대로 병렬화 하여 수행한다. 쿼드코어라고 가정하면 각 코어는 N/4 주소로부터 시작하여 (N+1)/4까지 연산을 수행하게 된다.

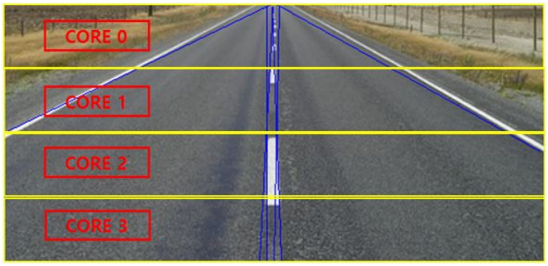

Fig. 8 Image horizontal division

3.3. 제안하는 고속 데이터 레벨 알고리즘

데이터 레벨 알고리즘은 설계가 쉽고 간단하지만 병목현상 문제가 발생하게 된다. 필터가 많을수록 암묵적으로 동기화가 발생하므로 복잡한 시스템일수록 데이터 병렬화의 이득을 얻기 힘들다. 따라서 본 논문에서는 이러한 데이터 레벨 병렬화를 향상시킨 고속 데이터 레벨 알고리즘을 제안한다. 고속 데이터 레벨 알고리즘

을 구현하는 방법은 데이터 레벨 병렬화의 각 필터들을 분해하여 하나의 기능으로 동작하도록 통합(Integrate)하여 병목현상을 감소시켰다. N단계의 필터를 수행할 때, 이미지 크기는 동일하기 때문에 각 필터를 데이터 병렬화 시, 수행하는 연산량은 동일하므로 공통으로 묶어서 처리할 수 있다.

즉, 결합한 기능들을 병렬처리를 수행한다. 그림 9는 데이터 레벨 알고리즘 블록도이고 그림 10은 제안하는 고속 데이터 레벨 알고리즘의 블록도이다. Integrated functions은 그림 12를 참고하면 된다. 그림 11과 같이 데이터 레벨 기반의 알고리즘은 각 코어가 각각의 필터들에 접근하여 병렬처리를 하게 하게 된다. 단계별로 필터들을 통과할 때 동기화가 되므로 필터의 단계가 많아질수록 병목현상이 발생하게 된다. 본 논문에서 제안하는 알고리즘 그림 12는 필터의 단계가 많아도 차선 검출에 사용되는 연산들은 순차적으로 진행되고 데이터의 종속성이 존재하지 않기 때문에 병합할 수 있다. 멀티 코어가 병합된 함수를 한 번만 수행하면 되기 때문에 병목현상이 감소할 수 있다.

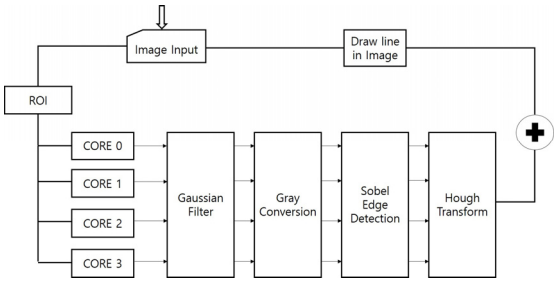

Fig. 9 Data level parallelism block diagram

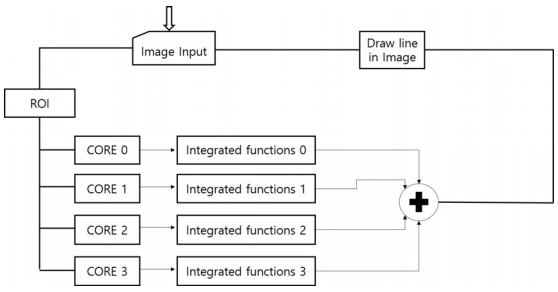

Fig. 10 High-speed Data level parallelism block diagram

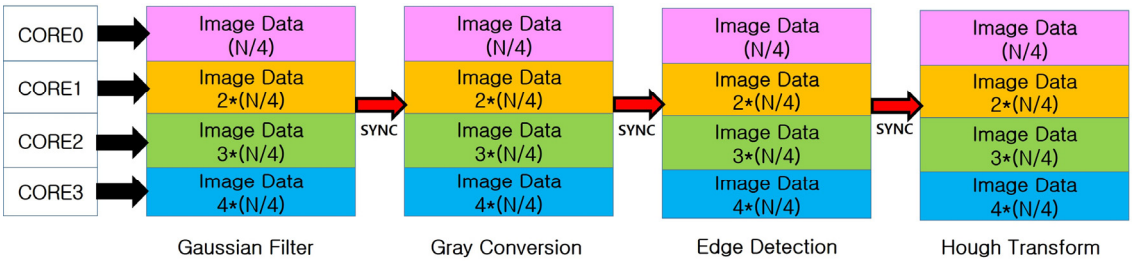

Fig. 11 Data level parallelism synchronization

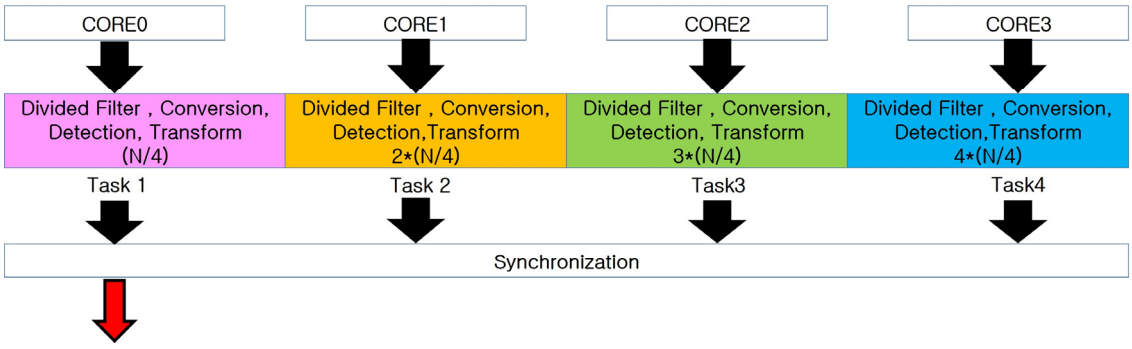

Fig. 12 High-speed data level parallelism synchronization

Ⅳ. 멀티코어 기반 시스템의 성능 검증

설계가 비교적 간단한 알고리즘인 데이터 레벨 병렬처리의 단점은 병목현상이 발생한다는 점이다. 병목현상은 필터가 많아질수록 더욱 많이 발생하며 영상처리의 경우 대부분이 다중필터를 사용하므로 병목현상의 영향이 더 커진다. 따라서 본 논문에서는 제안하는 고속 데이터 레벨 알고리즘과 기존의 데이터 레벨 알고리즘과 비교 실험하고 측정하였다. 차선 검출에서 가우시안 필터, 회색 조 변환, 외곽선 검출은 데이터 레벨 병렬 기법이 효율적이며 허프 변환은 쓰레드 병렬 기법이 효율적이다. 따라서 데이터 레벨로 병렬화가 가능한 3개의 필터를 2단계, 3단계로 나누어 측정하였다. 표 1은 차선검출에 필요한 2개 필터를 데이터 병렬화 알고리즘을 사용하여 측정한 표다. 표 2는 2개의 필터를 제안하는 고속 데이터 레벨 알고리즘으로 측정한 데이터이다. 두 알고리즘 표 1과 표 2를 비교하면 프레임 처리 2000개를 기준으로 쿼드코어인 경우 약 1.5초, 옥타코어인 경우는 약 0.2초정도가 빠르다. 표 3은 차선검출에 필요한 3개 필터를 데이터 병렬화 알고리즘을 사용하여 측정한 데이터이다. 표 4는 3개의 필터를 제안하는 고속 데이터 레벨 알고리즘으로 측정한 데이터이다. 표 3과 표 4 데이터를 비교하면 프레임 처리 2000개를 기준으로 쿼드코어인 경우 약 3초가 빠르고 옥타코어인 경우는 약 2초가 빠르게 나타났다. 필터의 단계가 많을수록 고속 데이터 레벨 알고리즘의 전체적인 성능 효과가 좋게 나타났다. 따라서 이미지를 처리하는데 복잡한 필터를 빠르게 병렬처리하기 위해 데이터 레벨에서 병목현상을 제거해 고속 데이터 레벨을 사용하면 성능이 우수함을 실험을 통해 증명하였다.

Ⅴ. 결 론

본 논문에서는 차선검출 시 멀티코어 플랫폼을 이용하여 병렬화 알고리즘을 제안하고 구현하였다. 또한 데이터 레벨 병렬처리 기법의 단점인 필터링 단계별 동기화로 인한 병목현상을 줄인 고속 데이터 레벨 알고리즘을 제안하였다. 데이터 레벨 병렬 처리 실험결과 싱글 코어보다 멀티코어가 처리 시간이 효율적으로 나타났다. 그러나 데이터 레벨 알고리즘은 병목현상이 있어 N개의 프로세서를 사용해도 N배의 성능을 얻기는 어렵지만 제안하는 알고리즘의 실험결과 기존의 데이터 레벨보다 평균적으로 빠른 성능을 얻게 되었다. 또한, 시스템의 복잡도가 높을수록 병목현상이 기존 알고리즘보다 많이 제거되기 때문에 고속 데이터 레벨 알고리즘이 더 높은 성능을 얻을 수 있었다. 본 차선 검출 시스템에 실제 블랙박스 도로 영상을 도입하여 측정한 결과 영상 입출력, 메모리 할당 및 해제, 관심영역 설정 등에 의한 전처리

Table. 1 Data level parallelism of 2 filters (sec)

Number of Frames	single core	quad core	octa core (Thread)
50	0.363	0.243	0.111
100	0.727	0.487	0.211
200	1.473	0.890	0.413
500	3.600	2.119	0.994
1000	7.240	4.100	1.958
2000	14.45	8.403	3.776

Table. 2 High-speed data level parallelism of 2 filters (sec)

Number of Frames	single core	quad core	octa core
50	0.363	0.222	0.1
100	0.727	0.337	0.193
200	1.473	0.694	0.369
500	3.600	1.82	0.908
1000	7.240	3.755	1.79
2000	14.45	6.93	3.58

Table. 3 Data level parallelism of 3 filters (sec)

Number of Frames	single core	quad core	octa core
50	0.570	0.460	0.196
100	1.130	0.740	0.385
200	2.260	1.678	0.712
500	5.656	3.929	1.740
1000	11.212	8.200	3.412
2000	22.570	15.828	6.805

Table. 4 high-speed data level parallelism of 3 filters (sec)

Number of Frames	single core	quad core	octa core
50	0.570	0.340	0.136
100	1.130	0.630	0.259
200	2.260	1.265	0.513
500	5.656	3.188	1.190
1000	11.212	6.330	2.400
2000	22.570	12.89	4.754

과정을 제외하면 싱글 코어 경우 약 30 Frames/sec의 성능을 얻었다. 표 3에서는 데이터 레벨 알고리즘을 사용하였고, 싱글코어와 옥타코어를 비교하면 약 3.3배의 빠른 성능을 보였으며 병렬 처리 프로그래밍을 적용한 결과 약 100 Frames/sec의 성능을 얻었고 실험 결과 데이터와 근사한 수치를 얻을 수 있었다. 표 4에서 제안하는 고속 데이터 레벨 알고리즘 경우에는 싱글코어와 옥타코어를 비교하면 약 5배의 빠른 성능을 보였으며 약 150 Frames/sec의 우수한 성능을 얻게 되었고 마찬가지로 실험 데이터와 근사한 수치를 얻을 수 있었다.

REFERENCES

[1] K. W. Lee, “A study on the improvement for analyzing the dna fragmentation using histogram,” M.S. theses, Sangji University, Aug. 2012.

[2] G. R. Choi, J. H. lee, “Edge detection of sonogram using sobel operator” *Journal of the Korean Society of Radiology*, vol. 2, no. 2, pp. 17-21, June 2008.

[3] J. R. Lee, K. R. Bae, B. I. Moon, “A hardware architecture for real-time line detection based on improved hough transform voting scheme” *The Korean Institute of Communications and Information Sciences Winter conference* , p. 88-89, 2013.

[4] "Hough Transform - C++ Implementation" [Internet]. Available: http://www.keymolen.com/2013/05/.

[5] "Wikipedia - OpenMP" [Internet]. Available: https://ko.wikipedia.org/wiki/OpenMP.

[6] "OpenMP (SMP)" [Internet]. Available: http://egloos.zum.com/qahuni/v/2517447.

[7] "Parallel program Environment" [Internet]. Available: http://super.kma.go.kr/super/super_program.jsp.

이효찬(Hyo-chan Lee)

2014.08 호서대학교 정보통신공학과 공학사
2016.08 호서대학교 정보통신공학과 공학석사
2016.07~현재 한국전자통신연구원 (ETRI) 위촉연구원
2016.09~현재 호서대학교 정보통신공학과 박사과정 재학 중
※관심분야 : 임베디드 시스템 설계, 병렬화프로그래밍, 영상 신호 처리

문대철(Dai-tchul Moon)

1987 고려대학교 전자공학과 공학박사
1994 North Carolina State Univ. 연구교수
2005 Minnesota State Univ. Duluth 객원교수
1984~현재 호서대 정보통신공학과 교수
※관심분야 : DSP 및 영상신호처리 칩 설계, SoC 설계, 임베디드 시스템 설계, VLSI 신호처리

박인학(In-hag Park)

1992 INPG(프)Microelectronics 공학박사
1982~1985 한국전자기술연구소 연구원
1985~2000 한국전자통신연구소 (ETRI) 책임연구원(실장)
2000~현재 (주)시스템센트로이드 이사
※관심분야 : 웹기반 CAD 시스템 개발, EDA 응용 소프트웨어 개발, 임베디드 시스템 설계, 반도체 IP 설계

허강(Kang Heo)

2008.02 호서대학교 정보통신공학과 공학사
2010.02 호서대학교 정보통신공학과 공학석사
2010.03~현재 호서대학교 정보통신공학과 박사과정 재학 중
2012.07~현재 (주)실리콘웍스 선임연구원
※관심분야 : DSP 및 영상신호처리 칩 설계, 임베디드 시스템 설계