Advanced SIMD를 이용한 움직임 추정 최적화 방법

김완수*, 이재흥*
*한밭대학교 컴퓨터공학과
e-mail:ws8024@naver.com

An Optimization Method of Motion Estimation using Advanced SIMD

Wan-Su Kim*, Jae-Heung Lee*
*Dept of Computer Engineering, Hanbat National University

요 약

최근 CPU의 코어 클럭을 높이는 대신 동일한 클럭의 코어 수를 늘림으로써 성능을 향상시키고 전력 소모도 줄이는 멀티코어가 등장하고 있다. 이러한 멀티코어 플랫폼의 등장으로 인해 해당 코어들의 자원을 효율적으로 사용하여 동시에 처리하는 병렬처리 기법에 관한 연구가 활발히 진행되고 있다.

본 논문에서는 병렬처리 기법의 종류 중 하나인 Advanced SIMD기반의 NEON을 적용한 고속화 ME 방법론을 연구 및 제안하였다. 최소화 SAD를 구하고 정확한 모션벡터를 선정하기 위해 다양한 ME 방법 중 전역탐색기법을 NEON에 적용하여 동시에 128비트씩 연산을 수행하였다. 그 결과 영상의 크기에 따라 계산 성능이 최대 60% 이상 향상되는 효과를 검증하였다.

1. 서론

현존하는 비디오 코덱 중 가장 좋은 성능을 발휘하는 H.264/AVC는 많은 형태의 변환을 거쳐 최적화되고 우리 실생활 여러 곳에 사용되고 있다. 아직까지도 용도에 맞게 최적화 작업이 이루어지고 있는 H.264/AVC에서 특히, ME(Motion Estimation)는 알고리즘 측면에서 가장 많은 접근이 이루어지는 부분 중 하나이다. 화면 간 예측된 영상 데이터의 비트수와 연산 시간을 줄이기 위해 수많은 방법론들이 제안되었고 실제로 많은 성능향상 효과를 가져왔다. 하지만 이는 실행되는 플랫폼을 고려하지 않고 알고리즘적인 측면만을 고려한 것으로써 시스템 아키텍처적인 측면에서의 최적화는 아직 이루어지지 않은 부분들이 많아 반드시 최적화 할 필요가 있다.

최근 모바일 기기들이 발전함에 따라 SoC(System On-Chip) 성능도 눈부시게 발전하고 있다. ARM사에서 제공하는 Cortex-A8부터 NEON 미디어처리엔진[1]이 포함된다. NEON은 부동소수점 처리장치의 성능 및 기능을 제공하는 것에 덧붙여 미디어 및 신호 처리 함수 작업을 가속화하기 위해 도입된 Advanced SIMD 아키텍처이다. 본 논문에서는 이러한 시스템 아키텍처 접근을 통하여 ME를 고속화하여 처리할 수 있는 방법론을 제안한다.

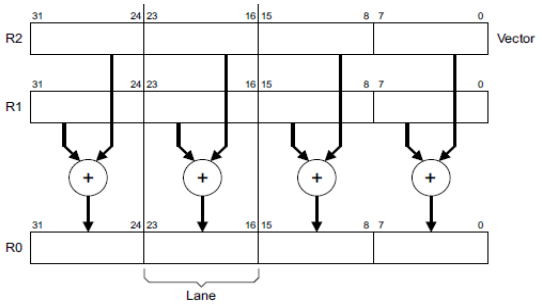
※ 본 연구는 교육과학기술부와 한국연구재단의 지역혁신인력양성사업으로 수행된 연구결과임

본 논문의 구성은 다음과 같다. 2장에서는 본 논문에서 핵심이 되는 NEON 기술과 H.264 ME에 대해 소개한다. 3장에서는 NEON 기술을 적용하여 구현한 고속화 ME를 소개하고, 4장에서 성능 측정 결과를 논의한다. 마지막으로 5장에서는 본 논문에 대한 결론 및 향 후 연구의 진행 방향을 제시한다.

2. 관련연구

2.1 NEON

NEON은 ARM Cortex-A8부터 지원하는 Advanced SIMD(Single Instruction Multiple Data)기반의 아키텍처로 오디오와 비디오, 그래픽 처리를 위한 명령어를 추가한 미디어처리 전용엔진이다. 정수와 부동소수점을 모두 지원하며, 메모리에 직접 접근이 가능하여 효율적으로 데이터를 조작하고 성능을 향상시킬 수 있다.



(그림 1) NEON 오퍼레이션

NEON은 (그림 1)과 같이 32비트 레지스터 R1, R2를 동시에 계산하여 R0에 결과 값을 저장하는 오퍼레이션을 제공한다. NEON은 128비트 레지스터 (Q0~Q15) 16개와 32비트 레지스터 (D0~D31) 32개를 지원하여 최대 128비트의 데이터를 동시에 연산할 수 있는 성능을 지닌다[2].

2.2 NEON Intrinsic 함수

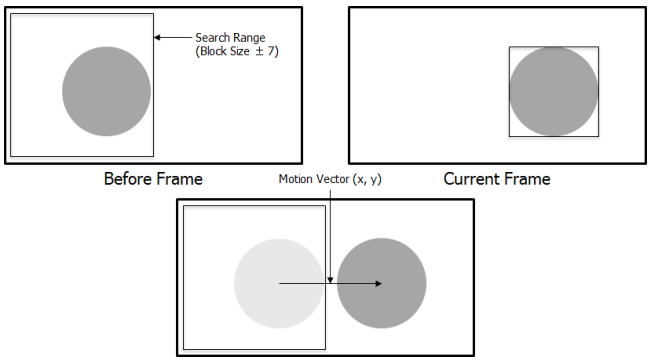
SIMD 명령 기반의 NEON은 어셈블리 언어구조이기 때문에 C/C++에 비해 많은 구현의 어려움이 따르게 된다. ARM사에서는 이런 단점을 보완하기위해 어셈블리코드로 이루어진 Intrinsic 함수를 제공한다. Intrinsic 함수는 C/C++ 환경에 익숙한 함수형태로 제공되며 파라미터 값만 넘겨주면 해당되는 SIMD 명령을 동일하게 수행한다.

2.3 H.264/AVC ME

ME는 프레임 간의 시간적 상관성을 이용하여 움직임을 압축하는 기법이다. ME는 현재 프레임을 MxN 블록단위로 나누어서 이전 프레임과의 최소 블록매칭 오차 지점을 찾는데 단순한 연산으로 이루어진 SAD(Sum of Absolute Difference) 방정식이 일반적으로 많이 사용된다.

$$SAD(i,j)=\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}|F_t(x,y)-F_{t-1}(x,y)| \quad (1)$$

여기서 F_t는 현재 프레임이고, F_{t-1}은 이전 프레임을 나타낸다. 식 (1)을 통해 알 수 있듯이 MxN 블록의 SAD를 구하기 위해서는 MxN번의 연산이 필요하다. 이와 같이 전역탐색기법은 해당되는 모든 위치의 SAD를 계산하여 최소의 SAD를 찾아 움직임벡터를 결정짓기 때문에 탐색영역 내에서 최소 SAD를 찾을 수 있고 구현이 쉽다. 하지만 모든 블록을 계산해야 하기 때문에 다른 탐색기법들에 비해 연산량이 많아 실시간 처리를 요하는 시스템에서의 적용은 어렵다는 단점을 가지고 있다.

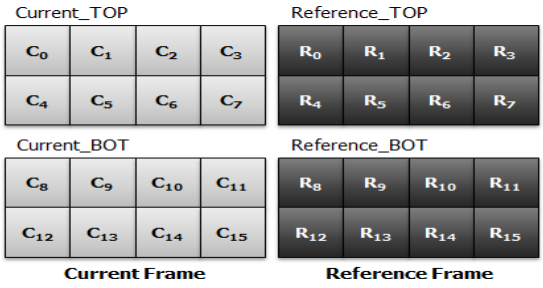


(그림 2) 전역탐색기법(Full Search)

ME는 전역탐색기법을 제외하고도 3-Step 탐색기법, 육각탐색기법, 다이아몬드탐색기법[3], 빠른 전역탐색기법[4] 등 다양한 탐색기법이 존재한다. 이러한 탐색기법들은 계산량을 줄이기 위해 모든 블록을 탐색하기 보다는 정해진 패턴에 해당하는 블록들만을 사용하여 ME를 수행한다. 그러면 계산량은 현저히 줄어들어 속도는 개선되지만 정확한 모션벡터를 예측하지 못해 화질의 저하를 가져오는 단점을 가지고 있다.

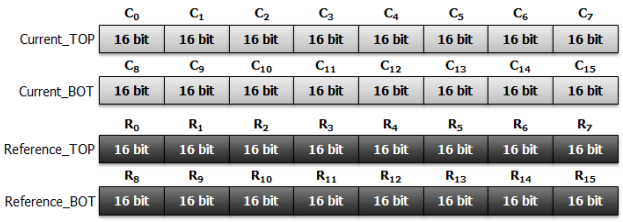
3. 본 논문의 제안 방법

본 논문에서는 다양한 ME 탐색기법 중 전역탐색기법을 NEON에 적용하여 최소의 SAD를 계산하고 정확한 모션벡터를 구하는 방법을 제안한다. 먼저 NEON 오퍼레이션은 스칼라가 아닌 벡터방식[1]으로 연산이 이루어진다.



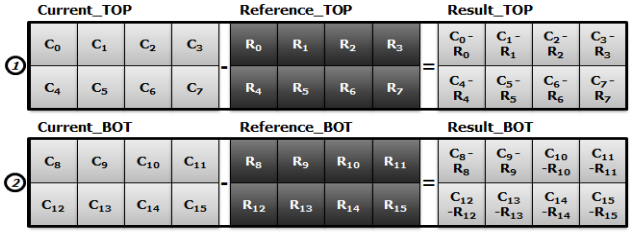
(그림 3) 맵핑된 4x4 블록 데이터

NEON은 벡터 방식의 연산을 지원하기 때문에 스칼라 데이터인 블록 값들을 벡터로 변환시키면 (그림 3)과 같이 현재 프레임의 블록 값을 나타내는 Current_Top과 Current_Bottom, 이전 프레임의 블록 값을 나타내는 Reference_Top과 Reference_Bottom으로 맵핑된다.



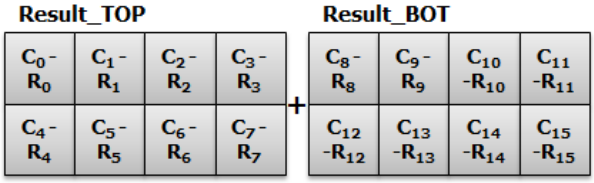
(그림 4) NEON 레지스터 구성

벡터 방식으로 맵핑된 스칼라 데이터들은 (그림 4)와 같이 NEON 16비트 레지스터 8개의 공간에 각각 할당된다. 이렇게 할당된 데이터는 NEON 오퍼레이션을 통해 동시에 128비트 연산을 수행하게 된다.



(그림 5) NEON 뺄셈 오퍼레이션을 이용한 SAD 연산

탐색영역 내에서 현재 프레임과 이전 프레임의 최소 SAD를 계산하기 위해 (그림 5)와 같이 128비트 뺄셈 연산을 동시에 수행시킨다. 기존 전역탐색기법을 이용한 4x4 블록매칭에서는 16번의 연산이 순차대로 이루어지지만, NEON을 적용한 전역탐색기법에서는 NEON 오퍼레이션을 2번만 사용함과 동시에 128비트 뺄셈 연산이 이루어지게 된다. 이와 같이 NEON은 기존의 블록매칭방법보다 8배 더 빨리 계산된다.



(그림 6) NEON 덧셈 오퍼레이션을 이용한 SAD 연산

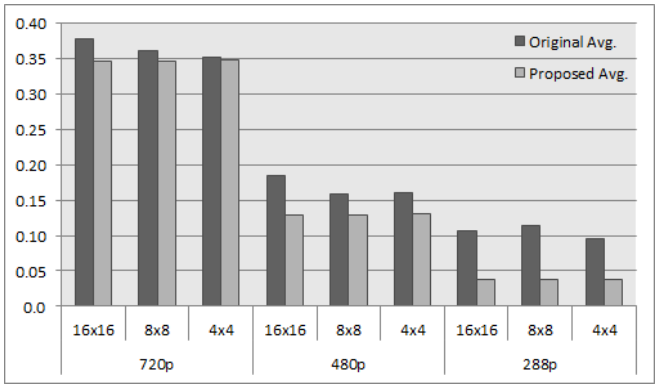
동시에 128비트 뺄셈된 데이터는 Result_Top과 Result_Bottom에 저장되고, 다시 이 데이터는 (그림 6)과 같이 NEON 덧셈연산을 통해 최종 SAD를 계산하게 된다. 계산된 SAD는 벡터 데이터에서 스칼라 데이터로 변환 맵핑시켜 사용한다. 최종적으로 계산된 SAD를 사용하여 탐색영역 안에서 최적의 모션벡터를 찾게 된다.

4. 실험결과 및 분석

본 논문에서는 SoC를 기반으로 하는 임베디드 시스템을 구축하고 NEON을 이용한 고속 ME를 구현하였다. 성능 측정을 위해 사용한 임베디드 시스템은 ARM Cortex-A9(1.3GHz)[1]을 채택한 듀얼코어 시스템에서 실험이 이루어졌다. Cortex-A9은 코어별로 NEON을 별도로 지원하고 있어 Cortex-A8보다 더 빠른 속도향상을 가져올 수 있다. 실험방법은 NEON을 적용하지 않은 전역탐색기법과 NEON을 적용한 전역탐색기법의 비교로 진행하였다. 또한 NEON에서 제공하는 자동 벡터화 방법[1]을 적용하여 프로세서 사이의 이식성을 유지시킬 수 있게 하였다.

<표 1> ME 수행시간 비교 (단위:sec)

구분		Original Avg.	Proposed Avg.	개선율
720p	16x16	0.378835	0.347117	8.37%
	8x8	0.360989	0.346568	3.39%
	4x4	0.352412	0.348214	1.18%
480p	16x16	0.185431	0.129492	30.1%
	8x8	0.159438	0.128977	19.1%
	4x4	0.161316	0.130047	19.3%
288p	16x16	0.107332	0.037868	64.7%
	8x8	0.114644	0.037723	67.1%
	4x4	0.094937	0.038131	59.8%



(그림 7) ME 수행시간 그래프 (단위:sec)

본 논문에서 제안하는 NEON을 적용한 전역탐색기법 실험결과는 <표 1>과 같다. 영상의 크기가 288p에서 720p으로 커질수록 개선율이 줄어들었지만 NEON을 적용하지 않았을 때와 비교하면 최대 67%에서 최소 1%의 속도 향상 효과를 가져왔음을 확인하였다.

5. 결론

본 논문에서는 H.264/AVC 부호화기에서 가장 많은 연산량과 시간을 차지하는 ME를 Advanced SIMD기반의 NEON을 적용한 결과를 실험을 통해 알아보았다. NEON을 적용함으로써 수행시간이 영상 크기에 따라 각각 다르게 개선됨을 확인하였다. 추후에는 DCT(Discrete Cosine Transform)와 IF(Interpolation Filter), IP(Intra Prediction) 등 H.264/AVC에서 사용되는 다양한 알고리즘들에 대해 NEON을 적용한 최적화 연구를 수행할 것이다.

참고문헌

[1] Cortex-A9 MPCore Technical Reference Manual, http://www.arm.com

[2] Chirag Pujara et al, “H.264 Video Decoder Optimization on ARM Cortex-A8 with NEON”, INDICON 2009 Anual IEEE, pp.1-3, Dec. 2009

[3] Shan Zhu, Kai-Kuang MA, “A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation”, Image Processing, IEEE Transaction on, Vol.9, pp.287-290, Feb. 2000

[4] Dong-kyun Park et al, “A Fast Motion Estimation Algorithm for SAD Optimization in Sub-pixel”, Integrated Circuits, ISIC’07 International Symposium on, pp.528-531, Sept. 2007

[5] Wing-Yee Lo, Daniel Pak-Kong Lun and Wan-Chi Siu, “Improved SIMD Architecture for High Performance Video Processors”, Circuits and System for Video Technology, IEEE Transaction on, Vol.21, pp.1769-1783, Dec. 2011