움직임 추정기의 병렬처리 구조 하드웨어 구현시 비유효 데이터의 효율적인처리 방법

Efficient Processing Technique for Unavailable Data in Hardware Implementation of Motion Estimator with Parallel Processing Architecture

박종화, 강현수
충북대학교 정보통신공학과

Jong-Hwa Park(dandy@chungbuk.ac.kr), Hyun-Soo Kang(hskang@chungbuk.ac.kr)

요약

본 논문은 H.264/AVC 부호화기의 실시간 동영상 부호화를 위한 하드웨어 구현과정 중 파이프라인 구조의 병렬 처리로 인한 데이터 부재문제의 해결방안을 제시하였다. 참조 소프트웨어(JM)의 움직임 추정 연산은 순차적인 처리가 가능하기 때문에 모든 데이터가 유효하지만, 파이프라인 구조로 하드웨어를 구현시 데이터가 병렬적으로 처리되므로 이전데이터가 유효하지 않은 경우가 발생한다. 본 논문에서는 MVp 연산시의 부재되는 데이터 문제를 해결하였다. 제안된 방법은 유효하지 않은 주변블록의 데이터(MV)로 인한 화질저하를 최소화하기 위하여 유효하지 않은 MV를 대신하여, 정수화소 움직임벡터, MVp(Motion Vector Predictor), MVcol(Motion vector of the Co-located block)을 사용하는 방법이다. BDPSNR로 실험 결과 같은 주제로 이전에 연구된 Huang[7]의 실험결과에 비하여 최대 QCIF영상에서 0.555dB, CIF 영상에서 0.834dB의 성능향상을 나타내고 있다.

■ 중심어 : | H.264 | MPEG4 AVC | 화면간 예측 | 움직임 추정 | 파이프라인 |

Abstract

In this paper, we propose the efficient processing technique for unavailable data in hardware implementation of motion estimator in H.264/AVC with parallel processing architecture. Motion estimation processing in the hardware is generally based on pipe-lining, some MV data of neighbor blocks are not available, whereas all MV data are valid in software processing where the data are sequentially processed. In this paper, we solve the problem of data being unavailable in MVp computation. To minimize the quality degradation caused by unavailable MVs, in the proposed method, the unavailable MV of a neighboring block is replaced with an integer pel unit MV, an MVp of neighboring blocks, or an MVcol (MV of co-located block). Comparing to the conventional method [7], our method outperformed maximally 0.832dB and 0.179dB for QCIF and CIF, respectively, in terms of BDPSNR.

■ keyword : | H.264 | MPEG4 AVC | Inter Prediction | Motion Estimation | Motion Compensation |

* "본 논문 2006년 정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구입니다." (KRF-2006- 521-D00324)
접수번호 : #081013-002
접수일자 : 2008년 10월 13일
심사완료일 : 2009년 02월 16일
교신저자 : 강현수, e-mail : hskang@cbnu.ac.kr

I. 서 론

최근 DMB(Digital multimedia Broadcasting)의 고화질의 동영상 서비스를 제공하기 위한 표준으로 H.264/AVC가 채택되었다[1][2]. DMB표준 채택 등으로 인하여 H.264/AVC에 대한 연구가 더욱 부각되고 있다. H.264/AVC는 ITU-T의 H.264와 ISO/IEC의 국제 표준인 14496-10(MPEG-4 part 10) Advanced Video Coding(AVC)에서 공동 작업으로 제정되었다.

H.264/AVC는 높은 복잡도 때문에 채택되지 못했던 기술들을 적극적으로 도입하여, 이전까지의 표준들 MPEG-2[3], MPEG-4 보다 성능이 향상되었다.

휴대용 단말기와 같이 전력 및 컴퓨팅 파워의 제약이 있는 경우에는 하드웨어 복잡도를 최적화할 수 있는 기술 개발[4][5]이 필수적이다. H.264/AVC의 동영상 압축 방식은 시간적 중복성과 공간적 중복성을 제거 하는 방식이다. 이는 프레임 간의 상관관계를 이용한 움직임 예측을 통해 시간적 중복성을 제거하는 움직임 예측[6]을 사용하는데 H.264에서 인코더에서 상당한 연산량을 차지하는 부분으로 이 부분의 연산량을 감소시키는 연구가 활발하다.

하드웨어 구현에 있어서, 디코더부분에 대한 논문은 많이 찾아볼 수 있지만, 인코더 구조에 대한 공개 자료[9][10]는 디코더 구조에 비해 극히 적은 편이다. 또한, 주로 특정 주요 모듈에 대한 공개 자료가 대부분으로써 본 논문에서 지적하고 있는 전체 인코더의 파이프라이닝으로 인해 발생하는 문제에 대한 언급은 찾기 어렵다.

본 논문에서는 움직임 추정기를 하드웨어로 구현 시, 파이프라이닝의 병렬적인 처리 과정[7][9]으로 인해 부재되는 데이터에 대한 문제점을 인식하고, 이 문제를 해결할 방법들을 제안하였다.

이후 본 논문의 구성은 다음과 같다. Ⅱ장에서 H.264의 움직임 추정 연산과 매크로블록 모드 결정방법에 대해 설명하였다. Ⅲ장에서는 하드웨어 구현에 따른 문제점을 제기하고, 제시한 문제점의 해결 방법들에 대하여 설명한다. Ⅳ장에서는 실험결과 분석을 통해 제안한 방법의 우수한 성능을 분석하며 마지막으로 Ⅴ장에서 결론을 맺는다.

II. 관련 연구

H.264/AVC의 인터 예측의 기본 기능은 시간적 중복성을 제거하여 동영상 압축 효율을 높이는데 있다. 시간적 중복성을 줄이기 위해서는, 영상프레임을 매크로블록 단위로 이전에 부호화된 영상프레임으로부터 움직임을 추정하는데, H.264/AVC에서는 이전의 매크로블록 단위의 움직임 추정뿐만 아니라, 블록과 서브블록 단위로 블록화하여 움직임 추정을 할 수 있도록 표준화 하였다.

인터 예측에서는 블록화 하여 부호화된 이전 영상과 현재 영상과의 시간적 중복성을 제거하기 위하여, 영상블록의 움직임을 추정하게 되는데 그 결과 얻어지는 것을 움직임 벡터라고 한다. 움직임 벡터는 블록의 크기가 작을수록 정확한 예측을 할 수 있으나, 더 많은 움직임 정보를 부호화하여야 하기 때문에 부호화 비트가 증가하게 된다. 이러한 발생 비트량과 화질사이의 관계를 고려하여 최적 모드를 선택하여 부호화하게 된다.

1. 움직임 예측 벡터와 움직임 추정 연산

부호화되고 있는 블록의 움직임 벡터는 주변 블록의 움직임벡터와 밀접한 상관관계를 갖기 때문에, 주변블록의 움직임 벡터 데이터들로부터 현재 블록의 움직임 벡터를 예측할 수 있다.

[그림 1]과 같이 H.264/AVC에서 사용하는 중간 값 예측 방식은 현재 구하려는 매크로블록의 주변 블록 좌측(A), 상단(B), 우측상단(C)의 블록들의 움직임벡터 x, y 성분 각각의 중간 값을 예측 움직임 벡터로 취한다.

H.264/AVC에서는 가변블록 움직임 보상이 사용되기 때문에 블록의 모양과 위치에 따라 예측 방법이 다음과 같이 다르다.

1.1 일반적인 블록의 예측 방법

- [그림 1]에 나타난 A, B, C 블록의 움직임 벡터의 중간 값을 사용.

1.2 예외적 처리

- 블록 B와 C 모두 시퀀스 또는 슬라이스 경계 밖에 위치할 경우, 블록 A의 움직임 벡터를 사용.

- 블록 A, B, C 중에 부호화 하려는 블록과 참조 시퀀스 번호가 같은 블록이 하나만 있는 경우, 그 블록의 움직임 벡터를 예측에 사용.
- 매크로블록이 16x8 모드인 경우, 상단블록은 블록 B를, 하단 블록은 블록 A의 움직임 벡터를 예측에 사용.
- 매크로블록이 8x16 모드인 경우, 좌측 블록은 블록 A를, 우측 블록은 블록 C의 움직임 벡터를 예측에 사용.

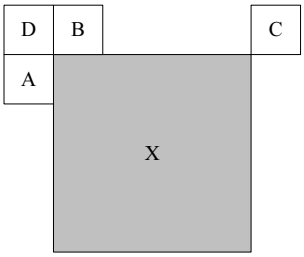


그림 1. 예측 벡터를 위한 참조 블록의 위치

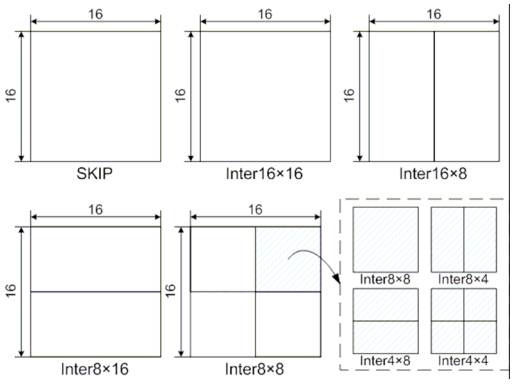


그림 2. 인터 모드에서의 매크로블록 모드

2. 매크로블록 모드

H.264/AVC은 더욱 정확한 예측을 위해서 더 작은 블록으로 분할하여 움직임 추정할 수 있도록 하였다.

[그림 2]는 H.264/AVC의 매크로블록 모드를 나타낸다. 움직임보상 블록크기를 MB유형으로 표시한 후, 16x16, 8x16, 16x8, 8x8단위의 4가지 방법으로 블록 분할을 수행한다. 8x8로 분할될 시에는 다시 8x8(화소)마다 sub-MB유형으로 8x8, 8x4, 4x8, 4x4의 블록 모드가 있다. 인트라 블록은 16x16, 8x8, 4x4 중 한 모드로 선택 된다.

3. H.264/AVC의 부호기와 하드웨어 구조

H.264 부호기는 [그림 3]에 나타난 바와 같이 인트라 예측모듈과 디블록킹 필터(deblocking filter)가 예측 루프(prediction-loop)에 들어있는 것이 특징이다.

H.264 부호기는 가변 블록 움직임 벡터 추정 모듈, DCT 정수변환 및 양자화 모듈, 엔트로피 부호화 모듈, 디블로킹 모듈, 등으로 이루어져 있다.

일반적으로 baseline profile의 경우, [그림 4]에 나타난 바와 같은 파이프라인 구조로 H.264가 구현된다[9].

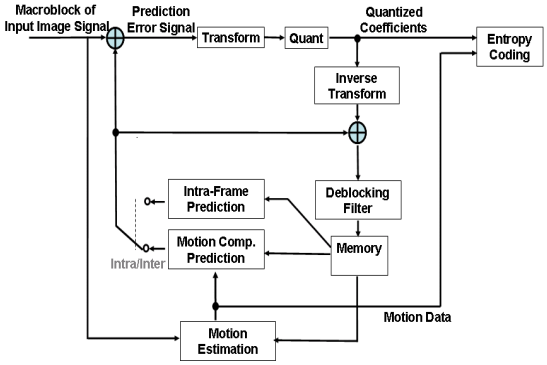


그림 3. H.264 부호기 블록다이어그램

III. 하드웨어 구현 시 비유효 데이터 발생의 문제점과 해결 방법

1. 문제점

파이프라인에서 핵심 모듈인 움직임 추정 모듈은 바로 이전 블록과 위쪽의 블록들을 이용하는 모듈로서 파이프라이닝에 의한 지연 때문에 현재 블록의 처리에 필요한 바로 이전 블록에 대한 정보의 부재라는 문제에 부딪히게 된다[7]. [그림 4]에서 점선(색선)으로 표기된 화살표가 문제가 발생하는 부분이다. (i+1)번째 MB의 정수화소 움직임 추정 모듈에서 (i)번째 MB의 모드 정보를 필요로 하나 이 정보는 유효하지 않고, 같은 사이클에 포함되어 있으므로 사용할 수 없다.

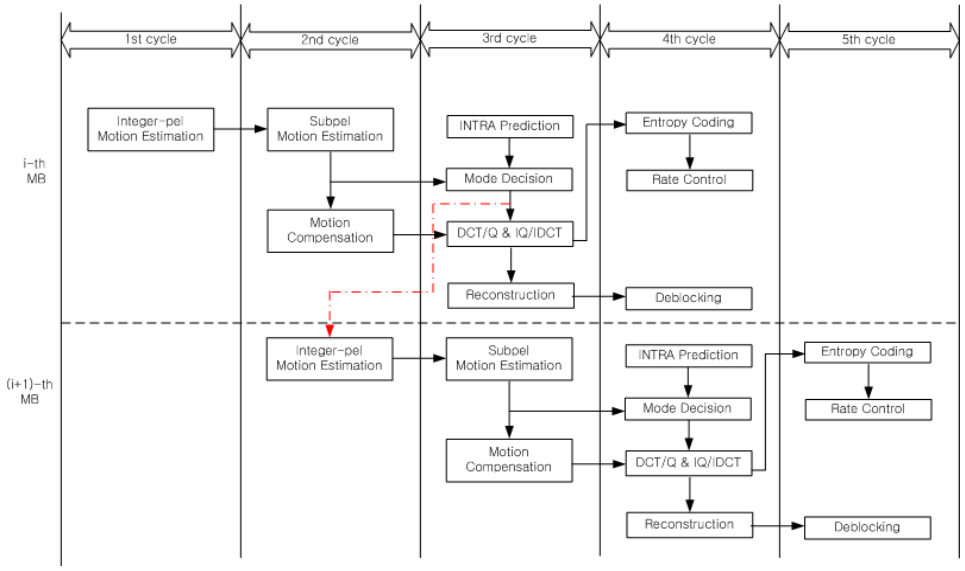
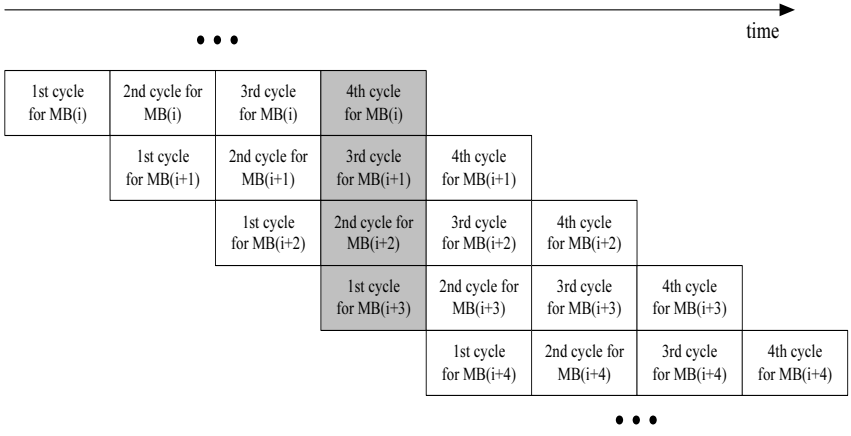


그림 4. H.264의 파이프라인 구조



* 1st cycle for MB(i) : Integer-pel motion estimation, Intra prediction for i-th MB
* 2nd cycle for MB(i) : sub-pel motion estimation, mode decision, and motion compensation for i-th MB
* 3rd cycle for MB(i) : Q/DCT, IQ/IDCT, Deblocking, and reconstruction for i-th MB
* 4th cycle for MB(i) : Entropy coding and rate control

그림 5. 파이프라인 사이클에 따른 모듈의 동작

[그림 5]는 파이프라인 사이클에 따른 해당 모듈의 동작을 표기한 것이다. 그림에서 음영진 파이프라인 사이클 동안에 수행되는 동작을 열거하면, (i)번째 매크로블록에 대한 4번째 사이클 동작(entropy coding, rate control), (i+1)번째 매크로블록에 대한 3번째 사이클 동작(Intra frame prediction, DCT/Q, IQ/IDCT, deblocking, reconstruction), (i+2)번째 매크로블록에 대한 2번째 사이클 동작(sub-pel motion estimation, mode

decision, motion compensation), (i+3)번째 매크로블록에 대한 1번째 사이클 동작(integer-pel motion estimation)이다.

(i+3)번째 매크로블록을 현재 매크로블록으로 생각해 보자. 이때, (i+2)번째 매크로블록은 (i+3)번째 매크로블록의 바로 이전에 부호화된 매크로블록에 해당된다. 현재 매크로블록의 정수화소 단위 움직임 추정을 위해서는 예측 움직임 벡터 MVp를 필요로 하게 되며 이는 (i+2)번째 매크로블록의 움직임 벡터뿐만 아니라 매크로블록 타입을 필요로 하게 된다. 그러나 (i+2)번째 매크로블록의 정보는 다음 파이프라인 사이클에서나 유효하다. 즉, [그림 5]에서 시간적으로 '1st cycle for MB(i+3)' 바로 오른쪽에 위치한 '2nd cycle for MB(i+3)'의 수행이 완료된 후 유효하게 된다. 현재 매크로블록의 움직임 추정에 필요한 바로 이전 매크로블록의 정보가 유효하지 않기 때문에 MVp를 구할 수 없게 되고, 결과적으로 움직임 추정 시 사용되는 데이터의 부재 문제점이 발생하게 된다.

2. 문제점 해결 방안

[그림 6]은 기존의 문제 해결 방법[7]으로서 각 연산 블록의 위치와 주위의 이용 가능 블록들의 참조되는 내용을 나타내고 있다. C22의 위치 블록 움직임 벡터 예측 연산에는 유효한 주변 데이터 MV0, MV1, MV2를 사용하는 것을 보여준다. 이 방법은 모든 블록이 같은 MVp를 사용하게 됨으로써 성능 저하를 가져오므로 이의 개선이 필요하다.

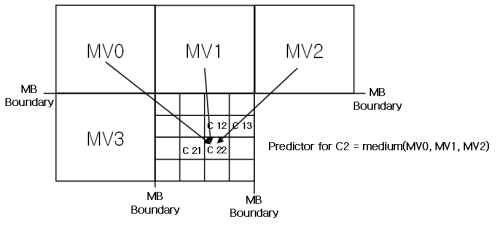


그림 6. Huang의 비유효 데이터 처리 방법

[그림 7]은 데이터 부재에 따른 매크로블록(16x16) 내의 연산 가능한 데이터의 종류를 나타내고 있다. A행이 움직임 벡터 추정 연산에 사용될 경우에는 정수화소 단위의 움직임 벡터값이 사용가능하다. B열은 MVp추정 연산에서 MV값을 사용할 수 있다. Cu블록들은 데이터 부재 현상으로 인하여 다음 소절에서 설명하는 추정된 MVp값을 사용한다.

	B 1 (MV)	B 2 (MV)	B 3 (MV)	B 4 (MV)
A 1 (Int-pel)	Cu11 (MVp)	Cu12 (MVp)	Cu13 (MVp)	Cu14 (MVp)
A 2 (Int-pel)	Cu21 (MVp)	Cu22 (MVp)	Cu23 (MVp)	Cu24 (MVp)
A 3 (Int-pel)	Cu31 (MVp)	Cu32 (MVp)	Cu33 (MVp)	Cu34 (MVp)
A 4 (Int-pel)	Cu41 (MVp)	Cu42 (MVp)	Cu43 (MVp)	Cu44 (MVp)

그림 7. MB(16x16)블록내의 데이터 종류

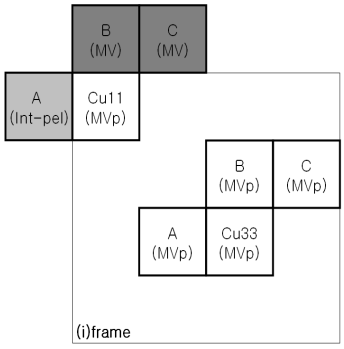


그림 8. 제안된 MVp 움직임 벡터 예측

2.1 제안된 방법 1 : 주변블록의 MVp를 이용한 현재 블록의 MVp추정방법

제안한 방법은 각 주변 참조블록들의 부재되는 MV값을 대신하여 정수화소 단위의 움직임 벡터 결과와 MVp를 사용한다. 즉, 제안된 MVp 추정 연산은 [그림 7]에 나타난 것처럼 연산 블록의 위치에 따라 데이터를 이용한다. 예를 들어, [그림 8]과 같이 Cu11블록의 위치에서의 MVp 추정 연산에서 사용되는 블록 A는 정수화소 단위의 움직임 벡터 값([그림 7]의 A1)으로 사용한다. 또한, Cu33블록의 MVp 추정 연산에는 A, B, C블록의 부재되는 MV값을 대신하여 MVp값([그림 7]의 Cu32의 MVp값, Cu23의 MVp값, Cu24의 MVp값)을 사용하여 MVp

추정연산을 한다.

2.2 제안된 방법 2 : MVcol을 현재 블록의 MVp 추정치로 사용하는 방법

JM의 MVp 추정연산과 보다 비슷한 결과를 얻기 위해 시간적 상관관계를 고려하여 현재 블록의 MVp 추정치를 이전 프레임의 같은 위치의 움직임 벡터(MVcol)값으로 한다. [표 1]는 IPPP로 코딩하였을 때, 주변 블록과 MV값의 차이를 나타내고 있다[11]. [표 1]의 결과에서 알 수 있듯이 MVt-1(이전블록의 MV값)의 값이 MV와 가장 근사한 값을 나타냄을 알 수 있다. 제안 방법은 [그림 9]에서처럼 Cu23의 움직임 추정 연산결과를 이전 프레임의 같은 위치의 MV값으로 이용한다[11].

표 1. QCIF 시뮬레이션 결과 – BDPSNR

Sequence	MVleft	MVtop	MV(t-1)	MVcol
Bus	1.4481	1.6413	1.8814	0.8164
Coastguard	0.7329	0.6344	0.7378	0.5792
Foreman	1.3963	1.2969	1.6348	1.5397
Stefan	2.2522	2.5026	2.7951	2.0852
Table Tennis	0.7275	0.7172	0.8814	0.5720

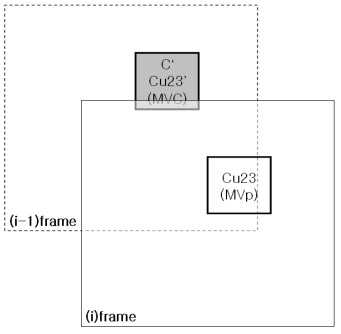


그림 9. MVcol 움직임 벡터 예측

2.3 제안된 방법 3 : 정수 화소 단위의 움직임벡터, MVp, MVcol을 이용한 MVp추정방법

앞에서 제안한 방법들은 주변의 추정된 값들을 사용하기 때문에 JM의 MVp 예측에 비하여 정확도가 떨어진다. 보다 참조 소프트웨어(JM)의 MVp 예측 결과에 근접할 수 있도록, 이전 절에서 제시한 정수화소 단위의 움직임 벡터, MVp, MVcol값 모두를 이용하는 방법을 생각할 수 있다.

[그림 7]의 Cu11, Cu21, Cu31, Cu41위치의 블록의 MVp 추정 연산에서는 JM에서 사용되는 MV와 가장 인접한 정수화소 단위의 움직임 벡터 데이터를 참조블록 A의 데이터 값으로 사용하였다. MV값이 유효하지 않은 Cu2x이하의 위치의 블록 연산 시에는 참조블록 B의 데이터로 MVp값을 사용하였고, 참조블록 C의 데이터 값으로 MVcol값을 사용하였다.

예를 들어, [그림 10]의 Cu12블록의 MVp 추정 연산의 참조 블록A의 데이터는 Cu12의 MVcol값을 사용하고, 참조블록 B, C의 데이터는 MV값을 이용한다. 다른 예로, [그림 11]와 같이 참조 가능한 주변 블록의 데이터가 모두 MVp인 경우, MVp 추정 연산에서 인접한 참조 블록 A, B의 MVp데이터 값을 사용하고, 참조 블록C의 데이터는 블록 Cu23의 MVcol을 이용하였다.

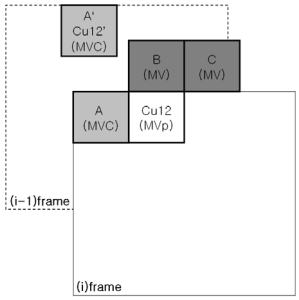


그림 10 MVp, MVcol 움직임 벡터 예측 1

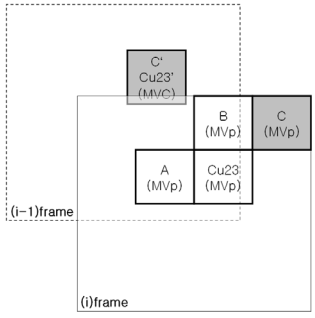


그림 11. MVp, MVcol 움직임 벡터 예측 2

IV. 실험 및 결과

제안된 방법을 사용한 MVp예측을 실험한 시뮬레이션은 JVT(Joint Video Team)에서 제공한 JM11.0을 사용하였다. 성능 비교는, 각 제안된 방법들의 실험결과에 대

한 PSNR(Peek-to-peek Signal to Noise Ratio)값과 BDPSNR[8]을 사용하였다.

실험 환경은 Baseline profile환경에서 100프레임씩 인코딩 하였다. 탐색영역은 +15 ~ -15로 하였고, 참조 프레임의 수는 1로 하였다. 움직임 탐색 블록 모드는 모든 블록을 사용 가능하게 하였다. 사용한 영상은 coastguard, foreman, mobile, table, akiyo, bus영상이며, 각각 CIF(352x288), QCIF(176x144)의 크기로 QP는 22, 27, 32, 37을 적용하여 사용하였다.

제안된 방법의 성능 평가를 위하여, 하드웨어구조를 관련논문 Huang[7]의 환경과 같이 조정하였다. 부재되는 데이터의 사용을 [그림 6]의 방법대로 수행되는 MB 블록 주위의 MV0, MV1, MV2의 데이터를 참조하여 연산하였다. 실험에서 하드웨어의 특성상 제한적인 메모리 대역폭 문제와 정확한 MVp의 계산이 불가능하므로, 움직임 벡터의 탐색영역의 시작점을 (0,0)로 설정하였다.

실제로 하드웨어 구현시 실험에서 사용되는 MVcol데이터를 이용하기 위해서는 별도의 내부 메모리가 필요할 것이다. 탐색 범위 -15~15의 모션벡터 정보를 저장하기위해 x값과 y값 각각 5비트 소요되며, CIF(352x288)의 영상인 경우 각 MB(16x16)블록 별 16개(4x4픽셀당 1개의 MV저장)의 MV의 저장소가 필요하다. 따라서 총 6336 word(1word=10bits)의 내부 저장 메모리가 필요하다.

[그림 12]와 [그림 13]은 CIF와 QCIF영상 실험에서 가장 월등한 성능을 타나내는 영상의 실험결과를 그래프로 나타내고 있다. 그래프에 비교의 척도로 Huang[7]의 실험 결과와 비교하였다.

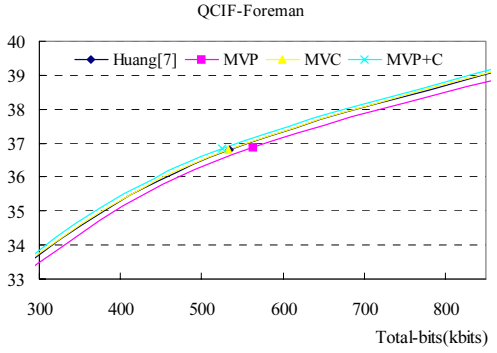


그림 12. Foreman(QCIF)의 실험결과

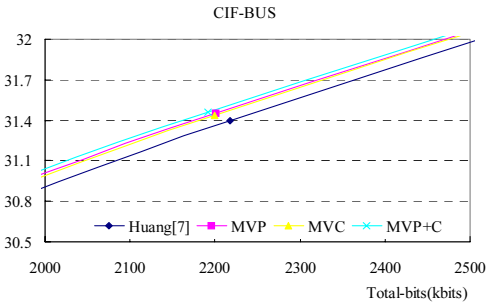


그림 13. BUS(CIF)의 실험결과

[표 2]와 [표 3]은 QCIF(176x144)와 CIF급 영상(352x288)에서 움직임 벡터 예측 결과를 움직임 추정 모듈의 연산을 제거한 Huang[7]의 방법과 제안된 방법들을 BDPSNR으로 비교한 결과이다.

표 2. QCIF 시뮬레이션 결과 – BDPSNR

QCIF(BDPSNR)	MVP	MVC	MVP+C
coastguard	-0.2032	0.0208	0.0518
foreman	-0.2155	0.0381	0.1190
mobile	-0.1997	0.0189	0.0396
table	0.0801	0.0588	0.1100
akiyo	0.1260	0.8161	0.8324
bus	0.0013	0.0006	0.0006

표 3. CIF 시뮬레이션 결과 – BDPSNR

CIF(BDPSNR)	MVP	MVC	MVP+C
coastguard	-0.2496	0.0328	0.0740
foreman	-0.1612	0.0973	0.1797
mobile	-0.1852	0.0375	0.0653
table	0.0755	0.0521	0.1070
akiyo	0.0152	0.0174	0.0408
bus	0.0684	0.0651	0.0928

[표 4]와 [표 5]는 QCIF와 CIF급 각 영상의 PSNR과 Total-bits 결과이다. 참고적으로 JM의 결과와 Huang[7]의 결과를 비교하였다. 실험 결과, 주변 블록의 MVp를 사용하는 것보다 MVcol을 사용하는 것이 더 좋은 결과를 나타내고 있다. 이것은 대부분의 영상에서 시간적 상관도가 공간적 상관도 보다 높음을 알 수 있다. [표 2]과 [표 3]에서는 좋은 성능을 나타낸 실험결과를 음영 처리하였다. 음영 처리된 칸의 대부분이 제안 방법 3에 해당함에 따라 시간 상관도와 공간 상관도를 모두 고려한 방법이 가장 우수함을 알 수 있다.

표 4. QCIF 시뮬레이션 결과 – PSNR, Total-bit(T-bits)

QCIF		JM		Huang[7]		MVP		MMVcolVcol		MVP+C	
		T-bits	PSNR	T- bits	PSNR	T- bits	PSNR	T- bits	PSNR	T- bits	PSNR
coastguard	22	2157992	38.95	2158728	38.95	2260808	38.94	2159984	38.94	2156216	38.95
	27	1002928	34.74	1008656	34.71	1047944	34.72	1008688	34.73	1003376	34.73
	32	363536	30.89	366760	30.86	389008	30.86	365440	30.87	362616	30.88
	37	131864	27.96	134880	27.91	149544	27.95	132520	27.91	132376	27.95
foreman	22	1043888	40.41	1058880	40.36	1131400	40.48	1055544	40.38	1047720	40.41
	27	524128	36.85	533808	36.79	562968	36.89	531960	36.81	525544	36.84
	32	271112	33.37	275896	33.30	299136	33.48	277136	33.36	273456	33.35
	37	148184	30.28	152960	30.11	168944	30.41	153360	30.17	150120	30.16
mobile	22	3796200	38.68	3797712	38.67	3892160	38.63	3802176	38.67	3794616	38.68
	27	1945544	34.08	1949264	34.05	2003664	34.05	1947576	34.06	1950472	34.08
	32	791696	29.43	797576	29.42	850160	29.49	795184	29.43	790704	29.43
	37	308240	25.59	312616	25.58	351144	25.85	310656	25.59	308072	25.58
table	22	1502368	39.79	1515616	39.76	1508576	39.78	1507000	39.77	1502672	39.8
	27	753256	36.16	763960	36.09	760296	36.13	758712	36.13	756496	36.15
	32	355928	33.02	364248	32.96	359856	32.99	362424	32.98	359736	33
	37	182912	30.46	189976	30.36	185104	30.44	188664	30.42	185088	30.47
akiyo	22	247696	42.66	249520	42.66	185104	30.44	249240	42.66	247816	42.65
	27	118032	38.94	117696	38.94	117664	38.94	117200	38.93	117968	38.96
	32	58216	35.24	25276	35.21	58200	35.24	58664	35.24	58448	35.26
	37	33568	32.17	33224	32.17	33200	32.16	33168	32.14	32976	32.14
bus	22	7840608	40.20	7840872	40.20	7841024	40.21	7840600	40.2	7840600	40.2
	27	5227808	36.23	5228360	36.23	5228200	36.23	5228288	36.23	5228288	36.23
	32	3359552	32.72	3359480	32.72	3359464	32.72	3359312	32.72	3359312	32.72
	37	2129240	29.85	2130392	29.85	2130384	29.85	2129400	29.85	2129400	29.85

표 5. CIF 시뮬레이션 결과 – PSNR, otal-bit(T-bits)

CIF		JM		Huang[7]		MVP		MVcol		MVP+C	
		T-bits	PSNR	T-bits	PSNR	T-bits	PSNR	T-bits	PSNR	T-bits	PSNR
coastguard	22	10020496	39.19	10030248	39.17	10856064	39.14	10026128	39.18	10020488	39.19
	27	5228048	35.16	5240360	35.13	5539688	35.13	5242344	35.14	5232272	35.15
	32	2173304	31.34	2196616	31.28	2304464	31.28	2183512	31.3	2178616	31.34
	37	748920	28.19	768208	28.10	852408	28.19	759504	28.13	752360	28.18
foreman	22	3564136	40.48	3622864	40.43	3919064	40.54	3595216	40.46	3568616	40.47
	27	1557416	37.21	1602344	37.16	1706720	37.27	1582088	37.19	1566392	37.22
	32	743608	34.21	773480	34.11	843456	34.3	762760	34.16	751080	34.19
	37	406568	31.50	433632	31.41	467568	31.66	422760	31.46	410872	31.48
mobile	22	14642520	39.18	14681808	39.16	15143112	39.12	14660320	39.17	14642040	39.17
	27	7887656	34.85	7914304	34.82	8134888	34.81	7899736	34.84	7896560	34.86
	32	3322704	30.44	3343240	30.40	3535552	30.47	3333440	30.41	3325064	30.43
	37	1207120	26.52	1219888	26.42	1389920	26.76	1211936	26.51	1205112	26.53
table	22	6045992	40.00	6098688	39.96	6070792	39.98	6074280	39.98	6063888	39.99
	27	2821968	36.22	2861176	36.16	2843976	36.2	2847784	36.2	2834504	36.22
	32	1253056	32.81	1276856	32.73	1266744	32.78	1270992	32.76	1262448	32.8
	37	571328	30.19	590864	30.11	577768	30.16	587536	30.14	575848	30.18
akiyo	22	832936	43.28	838688	43.27	837624	43.27	837472	43.27	836248	43.28
	27	377552	40.29	379360	40.24	380456	40.28	379968	40.27	376576	40.27
	32	176640	36.99	177304	36.97	178088	36.98	177824	36.99	176680	36.98
	37	94232	34.11	94840	34.06	94472	34.09	94680	34.09	94832	34.09
bus	22	8912336	39.41	8929008	39.40	8911384	39.4	8912496	39.4	8907152	39.41
	27	4595128	35.37	4616144	35.35	4606312	35.37	4601848	35.37	4594912	35.37
	32	2191720	31.46	2217800	31.40	2200192	31.45	2199312	31.44	2192096	31.46
	37	1049648	28.10	1087480	28.00	1065832	28.07	1067456	28.06	1057440	28.09

V. 결론

본 논문은 하드웨어 구현한 H.264/AVC의 움직임 추정 모듈에서 파이프라인 구조의 특성상 데이터가 병렬처리 됨으로써 나타나는 문제점 해결방안을 제시하였다. JM상에서의 순차적으로 처리되는 MVp 연산을 파이프라인 구조의 하드웨어 구현 시 독립적으로 처리가 가능하도록 참조되는 MV를 대신할 3가지 다른 데이터를 제시하였다. 참조되는 데이터는 주변블록의 MV값을 대신하여 수행되는 블록의 위치에 따라 정수화소 움직임 벡터, MVp, MVcol의 데이터를 이용하였다.

실험결과, 파이프라인 구조의 하드웨어의 MVp 추정 연산의 부재되는 데이터 문제를 해결할 수 있었다.

참 고 문 헌

[1] 한국정보통신기술협회, “초단파 디지털라디오 방송(지상파 DMB) 비디오 송수신 정합 표준”, Doc. TTAS/KO_07.0026, 2004(8).

[2] ISO/IEC 14496-10:2003, "Coding of Audio-visual Objects-Part 10: Advanced Video Coding." 2003.

[3] S. Eckart and C. Fogg, "ISO/IEC MPEG-2 software video codec," Proc. SPIE, Vol.2419, pp.100-118, 1995.

[4] J. Zhang, Y. He, S. Yang, and Y. Zhong, "Performance and complexity Joint Optimization For H.264 Video Coding," Proceedings of the 2003 International Symposium on Circuits and Systems, Vol.2, pp.25-28, 2003(5).

[5] 이성수, 이원철, “디지털 멀티미디어 방송을 위한 저전력 H.264 복호기 설계”, 전자공학회 논문지 : TC, 제44권, 제1호, pp.62-68, 2007.

[6] F. Dufaux and F. Moscheni, "Motion estimation techniques for digital TV: A review and a new contribution," Proceedings of the IEEE, Vol,83, No.6, pp.858-879, 1995(6).

[7] Y. W. Huang, "Hardware architecture design for variable block size motion estimation in MPEG-4 AVC/JVT/ITU-T H.264," Proceedings of International Symposium on Circuits and Systems, Vol.2, pp.796-799, 2003(5).

[8] G. Biontegarrd, "Calculation of average PSNR differences between RD-curves," ITU-T, Doc #VCEG-M33, 2001(4).

[9] L. V. Agostini and S. Bampi, "FPGA Based Architectures for H.264/AVC Video compression Standard," International Conference on Field Programmable Logic and Applications, pp.1-2, 2006(8).

[10] P. Kuhn, *Algorithms, Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation*, Kluwer Academic Publishers, 1999.

[11] 최민석, 김종호, 정제창, “움직임 벡터의 시공간적 상관도에 따른 효율적인 움직임 추정 기법”, 방송공학회논문지, 제12권, 제4호, pp.303-310, 2007.

저 자 소 개

박 종 화(Jong-Hwa Park) 준회원

- 2006년 2월 : 국립 한밭대학교(공학사)
- 2009년 2월 : 충북대학교(공학석사)

<관심분야> : 영상부호화, 영상정보처리

강 현 수(Hyun-Soo Kang) 종신회원

- 1999년 2월 : KAIST 전기및전자공학과 졸업(공학박사)
- 1999년 ~ 2001년 : 현대전자 과장
- 2001년 ~ 2002년 : 한국전자통신연구원 선임연구원
- 2002년 ~ 2004년 : 중앙대학교 첨단영상대학원 영상공학과 조교수
- 2005년 3월 ~ 현재 : 충북대학교 전기전자컴퓨터공학부 / 컴퓨터정보통신연구소 부교수

<관심분야> : 영상처리, 영상부호화, 컨텐츠보호기술, 사운드