H.264에서 다중참조 프레임을 이용한 효율적인 움직임 예측

정회원 김 성 은*, 종신회원 한 종 기**

An Efficient Scheme for Motion Estimation Using Multi-reference Frames in H.264/AVC

Sung-Eun Kim* ***Regular Member***, Jong-Ki Han** ***Lifelong Member***

요 약

H.264에서 다중참조 프레임을 사용한 움직임 예측 방법은 단일 참조프레임을 이용한 움직임 예측보다 더 많은 시간적 중복성을 제거하여 부호화 효율을 높이거나 채널에러에 강인하게 부호화하기 위해 사용된다. 하지만 다중 참조 프레임을 이용하여 움직임 예측을 하는 것은 단일의 참조 프레임을 이용하는 것보다 많은 계산량을 요구하기 때문에 비디오 인코더의 복잡도를 증가시키게 된다. 본 논문에서는 다중참조 프레임을 사용한 움직임 예측을 화질 열화 없이 적은 복잡도로서 가능하게 하는 알고리즘을 제안한다. 움직임 예측 절차의 복잡도를 줄이기 위해, 제안한 알고리즘에서는 연속되는 프레임 사이에 구성된 움직임 벡터맵을 이용하여 움직임벡터를 추정한다. 제안한 방식은 추정된 움직임벡터를 작은 탐색영역에서 보정하는 방식을 적용하기 때문에 기존의 방식들에 비해 적은 복잡도가 요구된다. 제안된 방법으로 추정된 움직임벡터는 각 참조프레임들에 대해 최적의 움직임 벡터를 효과적으로 추적하기 때문에 부호화 된 영상의 화질은 전 탐색영역 움직임 예측 알고리즘을 이용한 결과와 매우 비슷하다.

제안된 방식은 세가지 단계로 구성된다. (a) 연속되는 두 개의 프레임 사이에 벡터맵을 구성한다. (b) 벡터맵에 있는 요소벡터를 이용하여 시간적 움직임 벡터를 구성한다. (c) 마지막으로, 임시 움직임 벡터를 좁은 탐색영역에서 보정한다.

컴퓨터 실험을 통해 제안된 방식의 효율성을 입증하였다. 제안된 방식과 기존의 방식들과의 비교를 위해 H.264 부호화기에서 움직임 예측 모듈에 의해 소비된 CPU 시간을 측정하였다. 컴퓨터 실험을 통해 알 수 있듯이 제안된 방식에 의해 부호화된 영상의 화질은 기존 방식과 을 통해 얻은 영상화질과 거의 같으면서 알고리즘 복잡도는 크게 줄어드는 것을 볼 수 있다.

Key Words : H.264, Motion Estimation

Ⅰ. 서 론

JVT(Joint Video Team)라 불리는 비디오 압축 전문가 위원회에서는 H.264로 알려진 AVC (Advanced Video Coding)표준화를 과거 몇 년 동안 진행해 왔다[1]. H.264표준은 방송통신, 무선 네트워크를 통한 비디오 화상회의, VOD, 스트리밍 서비스, 대화형 비디오 서비스와 같은 다양한 분야에 적용을 위해 기술적으로 디자인 되었다. 또한 H.264는 저 비트율, 저 프레임율, 저 해상도에서부터 SDTV, HDTV와 같은 높은 비트율, 프레임율, 해상도를 가진 비디오까지 넓은 적용범위를 가지고 있다[2]. H.264표준에서는 높은 부호화 효율과 네트워크 환경에 강인한 부호화를 위해 다중 블록 모드를 위한 움직임

※ 본 연구는 서울시 산학연 협력사업(과제번호 10557)의 지원에 의하여 이루어진 것임.
* 세종대학교, 정보통신연구소, 정보통신공학과 (kimse@mcubeworks.com)
** 세종대학교, 정보통신연구소, 정보통신공학과 (hjk@sejong.ac.kr, 연락저자)
논문번호 : KICS2005-07-287, 접수일자 : 2005년 7월 19일, 최종논문접수일자 : 2006년 9월 4일

보상, 1/4 화소단위 움직임 보상, 다중 참조 프레임을 이용한 움직임 보상, Weighted prediction, 직접 공간 예측 (Direct spatial prediction), 루프 필터 (Loop filter)와 같은 효과적인 기술들이 채택되었다[3-5].

이러한 기술들 가운데 다중참조 프레임을 사용한 움직임 예측(MRME: Multiple reference frame motion estimation)은 H.264부호화기에서 높은 복잡도를 차지하는 반면에 부호화 효율 면에서는 높은 이득을 주기 때문에 매우 중요하다. 이러한 이유로 H.264에서 사용되는 MRME 방식의 효율성을 높이기 위하여 많은 연구가 진행 되어 왔다[12-15]. T. Wiegrand는 비디오 부호화에서 다중참조 프레임을 사용하기 위해 요구되는 Side 정보의 제어하기 위한 방식을 제안했다[12]. 또한 MRME 방식에서 고려되는 참조 프레임의 수를 줄이기 위한 방법이 제안 되었다[13]. 저자들은 블록의 변형 형태를 정수형 화소 위치, 1/2 화소 위치, 1/4 화소 위치로 분류하여, 매크로블록(Macroblock)의 변형 형태에 따라 움직임 추정을 위해 사용되는 참조 프레임의 수를 줄이기 위해 사용하였다. 예를 들어 두 번째와 세 번째 참조프레임 내에 있는 매크로 블록들 중 현재 매크로 블록과 동일 위치한 매크로블록의 변형된 형태가 같다면 현재 프레임과 가장 근접한 하나의 참조 프레임, 즉 두 번째 참조 프레임 만을 움직임 예측을 위해 사용하였다. 하지만 이 방식은 많은 참조 프레임들이 선택 되었을 때 복잡도를 크게 줄이지 못하는 단점을 가지고 있다. M. E. Al-Mualla와 N. Canagarajah, D. R. Bull는 SRME(Single Reference frame Motion Estimation) 방법을 위한 SMS(Simplex Minimization Search) 알고리즘을MRME 모듈을 위해 확장하였고, 알고리즘은 하나의 참조프레임을 이용한 움직임 예측과 유사한 복잡도를 요구되게 되었다[14]. 또한 MRME 방식의 복잡도를 줄이기 위해 Center-biased frame selection 알고리즘이 제안되었다[15]. 저자들은 각 참조프레임의 중심위치로부터 작은 탐색영역에서 움직임 벡터를 찾아, 최적의 움직임 벡터를 가지는 하나의 참조프레임 만을 선택하였다. 선택된 참조 프레임에서 더 향상된 최적의 움직임 벡터를 얻기 위해 전 탐색 영역 예측(full search) 방식이 적용되었다. 논문에서 저자들은 선택된 참조 프레임이 전역 최적 움직임 벡터를 가지고 있다고 가정하였다. 하지만 움직임 예측을 위한 비용함수는 convex 하지 않고 움직임 예측을 위한 탐색 영역은 제한 되어 있기 때문에 제안한 방법에 의해 선택된 참조 프레임은 전역 최적 움직임 벡터를 가지고 있지 않을 것이다. 비록 이 방법이 MRME 방식의 복잡도를 줄일 수 있다고 해도 알고리즘이 지역 최적화를 줄 수 있기 때문에 부호화된 영상에 화질손실을 가져다 줄 것이다.

본 논문의 목적은 H.264에서 사용되는 MRME 방식을 위한 효율적인 알고리즘을 제안하는 데에 있다. 제안한 움직임 예측 알고리즘이 MRME 방식에 적용 될 경우 적은 복잡도로 전 탐색 영역 움직임 추정방식과 유사한 화질을 나타낼 수 있다. 제안한 방식은 세 가지 단계로 구성된다. 첫 번째로 현재 프레임과 첫 번째 참조 프레임 사이의 움직임 예측 절차에서 구해진 움직임 벡터들로부터 벡터맵을 구성한다. 두 번째로, 구성된 벡터맵으로부터 움직임 벡터를 추정한다. 마지막으로, 최적의 움직임 벡터는 추정된 움직임 벡터를 중심으로 기존의 방식에서 사용되는 것 보다 작은 탐색영역을 이용한 움직임 벡터 보정과정으로부터 구해진다. H.264는 양 방향의 참조 프레임을 이용한 움직임 예측이 허락된다[11]. 본 논문에서는 알고리즘의 명확하고 간략한 서술을 위해 전 방향에 위치한 참조프레임 만을 고려한다. 게다가 제안한 방식이 양 방향에 있는 참조프레임을 이용한 움직임 예측방식을 위해 쉽게 확장 될 수 있기 때문에 본 논문에서 전 방향에 위치한 참조프레임만을 다루는 것은 알고리즘의 일반성 손실을 가져다 주지 않을 것이다.

본 논문은 다음과 같이 구성되어 있다. II장 에서는 H.264에서 사용되는 MRME 방식에 대하여 기술된다. III장에서 MRME 모듈을 위한 효율적인 방식이 제안된다. 제안한 알고리즘의 실험결과는 V장에서 기술되고, IV장에서는 결론을 맺는다.

Ⅱ. H.264의 움직임 예측 방식

2.1 기존 MRME 방식

H.264에서는 이전의 부호화된 하나이상의 영상들이 움직임 예측과정에서 참조 프레임으로 사용될 수 있다[3, 9, 11]. 현재 매크로 블록과 예측된 참조 블록의 차이는 하나의 참조프레임을 사용하는 것 보다 다중 참조 프레임을 사용함으로써 더욱 작아 질 수 있다 있다[3]. 따라서 MRME 방식은 영상의 화질 과 압축 비율 측면에서 H.264부호화기의 부호화 효율을 개선시키기 위해 사용된다.

그림 1에서는 다중 참조 프레임을 이용하여 움직임 벡터를 예측하는 과정을 보여준다. 그림에서 C

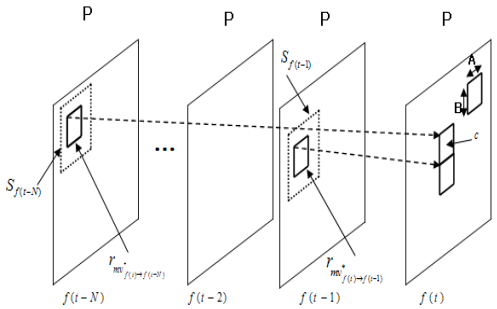


그림 1. H.264 에서 다중 참조 프레임을 사용한 움직임 벡터 예측.
Fig. 1. Motion vector estimation using multi-reference frames in H.264.

는 다중 참조 프레임으로부터 움직임 벡터를 예측하기 위해 사용되는 하나의 매크로 블록 혹은 부매크로 블록 (Sub-macroblock) 이다. 블록 C 의 크기는 A ×B {16 ×16, 16 ×8, 8 ×16, 8 ×8, 8 ×4, 4 ×8, 4 ×4}이고, f(t-k), k=1,...,N, 와 f(t)는 각각 k 번째 참조 프레임과 현재 프레임이다. 블록 c 가 7 가지 크기 중 하나인 A ×B 일 때 움직임 벡터를 구하기 위한 식은 다음과 같이 나타낼 수 있다.

$$mv^{*}_{f(t)\to f(t-k)} = \arg min_{mv_{f(t)\to f(t-k)}\in S_{f(t-k)}} \quad (1)$$
$$[\sum_{x=0}^{A-1}\sum_{y=0}^{B-1}|c(x,y)-r_{mv_{f(t)\to f(t-k)}}|+$$
$$\lambda_{motion}\times R_{motion}(pmv, mv_{f(t)\to f(t-k)})]$$

$mv_{f(t)->f(t-k)}$는 현재 프레임 f(t)와 참조 프레임 f(t-k) 사이의 후보 움직임 벡터이다. $r_{mvf(t)->f(t-k)}$는 f(t) 내의 블록 c 의 위치로부터 $mv_{f(t)->f(t-k)}$에 의해 구해진 f(t-k) 내의 참조 블록이다. 여기서 $r_{mvf(t)->f(t-k)}$와 c 모두 A ×B 의 크기를 가진다. λ_{motion}는 움직임 예측을 위한 Lagrange parameter 이고 pmv는 현재 블록의 위치에서 공간적으로 왼쪽, 위, 오른쪽 위쪽에 위치한 세 개 블록들의 움직임 벡터들의 median 연산으로부터 계산된 벡터이다.

$R_{motion}(pmv,mv_{f(t)->f(t-k)})$ 은 움직임 벡터를 부호화하기 위해 필요한 비트 수 이고, $S_{f(t-k)}$는 참조 프레임 f(t-k)을 이용한 움직임 예측에 사용되는 탐색 영역이다. 식 (1)에 대한 과정이 k=1,...,N 에 대하여 모두 수행된 후, 각 참조프레임에 대한 움직임 벡터 $\{MV^{*}_{f(t)->f(t-k)},k=1,2,...,N\}$들이 구해진다. 구해진 움직임 벡터들 중에서 최적의 움직임 벡터 $MV_{A\times B}$는 식 (1)을 최소화하는 움직임 벡터로부터 구해진다. 이 절차가 가능한 모든 블록 크기 A ×B = {16 × 16, 16 ×8, 8 ×16, 8 ×8, 8 ×4, 4 ×8, 4 ×4} 에 대하여 모두 적용된 후 H.264에서 지원되는 모든 블록 모드들에 대한 움직임 벡터 $\{MV_{16\times16}, MV_{16\times8},...,MV_{4\times4}\}$ 들이 구해진다. 이러한 움직임 벡터들로부터 최적의 모드 $MODE^{*}$ 는 다음의 비용함수를 최소화하는 모드로 선택 된다.

$$J(MODE) = D_{REC}(MODE) + \lambda \times R_{REC}(MODE) \quad (2)$$

$R_{REC}(MODE)$는 원본 블록과 하나의 모드 MODE를 사용하여 부호화된 블록의 차이를 제곱한 값들의 합이다. MODE 는 {16 ×16, 16 ×8, 8 ×16, 8 ×8, 8 ×4, 4 ×8, 4 ×4}의 모든 가능한 조합들 중 하나를 나타낸다. 예를 들어, MODE는 {하나의 16 ×16 블록}, {두개의 16 ×8 블록들}, {두개의 8 ×8 블록들과 8개의 4 ×4 블록들} 등이 될 수 있다. $R_{REC}(MODE)$는 매크로 블록을 하나의 모드를 사용하여 부호화할때에 헤더 정보, 움직임 벡터 정보 등을 부호화하는 데에 필요한 비트 수이다.

MPEG-2, MPEG-4와 같은 비디오 부호화기에서, 움직임 예측 과정은 움직임 벡터와 매크로블록 모드를 결정하기 위해 많은 계산 량을 필요로 한다. 게다가 다중 참조 프레임을 사용하는 것은 하나의 참조 프레임을 사용하는 것에 비해 더 많은 복잡도를 요구하기 때문에 움직임 벡터와 블록 모드를 빠르게 탐색하고 결정하는 것은 H.264부호화기의 복잡도를 줄이기 위한 가장 효과적인 방법 중에 하나이다.

2.2 기존의 MRME 방식에서의 복잡도 분석

이번 장에는 다중 참조프레임을 이용하여 전 탐색영역 예측을 수행하였을 경우에 요구되는 복잡도를 분석한다. 본 논문은 움직임 추정 과정에서 고려되는 블록 정합의 수를 줄이는 것에 중점을 두었기 때문에 계산량은 블록 정합 과정의 수 관점에서 구해진다. 분석을 위해 참조 프레임의 개수가 N 이고 k 번째 참조 프레임에 대한 탐색 영역 $S_{f(t-k)}$가 $(2W_{(t-k)}+1)\times(2H_{(t-k)}+1)$ 의 화소 크기로 제한 되어 있다고 가정한다. 여기서 $(2W_{(t-k)}+1)$와 $(2H_{(t-k)}+1)$ 는 탐색영역의 가로와 세로의 크기이다. 가정으로부터 기존의 MRME 알고리즘은 현재 프레임 f(t) 내에 있는 하나의 블록 c에 대하여 $\sum_{k=0}^{N}(2W_{(t-k)}+1)\times(2H_{(t-k)}+1)$ 만큼의 블록 정합을 수행하는 것을 알 수 있다. 만약 하나의 프레임 내에 있는 매크로 블록의 개수가 M

이라면, 하나의 프레임 내에 있는 모든 블록들에 대하여 움직임 벡터를 예측하기 위해 요구되는 총 블록 정합의 수는 다음과 같이 나타내어진다.

$$Q \times M \times \sum_{k=1}^{N} (2W_{(t-k)}+1) \times (2H_{(t-k)}+1) \qquad (3)$$

Q는 비용함수 (2)를 최소화 하기 위해 고려 되는 총 블록들의 개수 이다. 구해진 계산 량은 H.264부호화기에서 소비되는 총 계산 량에서 많은 부분을 차지 한다. 즉 식 (1)과 (2) 의 과정의 복잡도를 개선시키는 것은 H.264표준을 비디오 코덱으로 이용하여 실시간 멀티미디어 시스템을 구현 할 때에 매우 중요한 부분이다.

Ⅲ. H.264를 위해 제안하는 효율적인 MRME 알고리즘

이번 장에서는 다중 참조 프레임을 이용한 효율적인 움직임 벡터예측 알고리즘을 제안한다. 제안된 예측 절차는 기존의 방식에서 사용된 탐색영역의 크기 보다 작은 새로운 탐색영역 $S^{new}_{f(t-k)}$, k=2,3,...,N 의 사용을 제외하고 그림 1의 과정과 유사하다.

3.1 움직임 벡터맵

벡터맵 $MVMAP_{f(t-k+1)->f(t-k)}$, k=2,3,...N 들은 프레임 f(t-k+1)와 f(t-k)가 각각 현재 프레임이고 참조 프레임 이었을 때 예측 된 움직임 벡터들 $mv^{*}_{f(t-k+1)->f(t-k)}$로부터 구성된다. 즉 f(t)가 현재 프레임일 때 f(t-k+1)와 f(t-k), k=2,3,...N, 사이의 벡터맵 들은 이전의 절차에서 모두 구해진다. 움직임 벡터 $mv^{*}_{f(t-k+1)->f(t-k)}$ 들은 다음의 식에 의해 예측 된다.

$$mv^{*}_{f(t-k+1)\to f(t-k)} = \arg min_{mv_{f(t-k+1)\to f(t-k)} \in S_{f(t-k)}} [\sum_{x=0}^{A-1}\sum_{y=0}^{B-1} |c(x,y) - r_{mv_{f(t-k+1)\to f(t-k)}}| + \lambda_{motion} \times R_{motion}(pmv, mv_{f(t-k+1)\to f(t-k)})] \qquad (4)$$

식 (4)의 절차는 f(t-k+1)이 현재 프레임 일 때 SRME 방식을 적용했을 때와 동일하다. 이 절차는 f(t-k+1) 이 현재 프레임일 때인 이전의 단계에 포함되기 때문에 식 (4)를 이용한 움직임 벡터 예측 과정을 위해 계산 과정을 필요로 하지 않는다. 그림 2는 프레임 f(t-k+1) 와 f(t-k) 사이의 움직임 벡터맵 구성 과정의 예를 보여준다. 움직임 벡터들은 하나

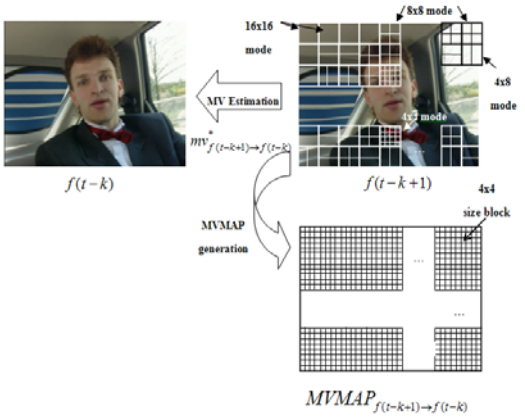


그림 2. 움직임 벡터맵의 구성 과정
Fig. 2. An example of the process to make a motion vector map.

의 참조프레임 f(t-k)을 이용하여 식 (1) 에 의해 예측되고, 각 매크로 블록의 모드는 16×16, 16×8, 8×16, P8×8 중 하나로 결정 된다. $MVMAP_{f(t-k+1)->f(t-k)}$ 내에 있는 요소 벡터들은 4×4 크기의 공간 해상도 크기로 저장되고, 같은 공간적 위치를 가진 $mv^{*}_{f(t-k+1)->f(t-k)}$의 복사 과정에 의해 생성된다. 예를 들어 만약 하나의 매크로 블록의 모드가 16×16로 결정되면, 예측된 하나의 움직임 벡터는 $MVMAP_{f(t-k+1)->f(t-k)}$내에 같은 공간적 위치를 가진 모든 16개의 요소 벡터들로 복사된다.

3.2 효율적인 움직임 예측

제안한 MRME 알고리즘의 절차는 그림 3에 보여진다. 현재프레임이 f(t) 이고 $MVMAP_{f(t-1)->f(t-2)}$이 현재 프레임 f(t-1)과 참조프레임 f(t-2)을 이용하여 식 (1)에 의해 주어 질 때, 최적의 움직임 벡터 $mv^{*}_{f(t)->f(t-2)}$는 참조프레임 f(t-2)로 부 터 구해진다. 첫 번째로 움직임 벡터 $mv^{*}_{f(t)->f(t-1)}$ 는 k=1 일 때 식 (1)에 의해 구해진다.

$$mv^{*}_{f(t)\to f(t-1)} = \arg min_{mv_{f(t)\to f(t-1)} \in S_{f(t-1)}} [\sum_{x=0}^{A-1}\sum_{y=0}^{B-1} |c(x,y) - r_{mv_{f(t)\to f(t-1)}}| + \lambda_{motion} \times R_{motion}(pmv, mv_{f(t)\to f(t-1)})] \qquad (5)$$

그 다음으로 $r_{mv*f(t)->f(t-1)}$에 의해 부분적으로 혹은 완전히 겹쳐진 $MVMAP_{f(t-1)->f(t-2)}$ 내의 요소 벡터들은 추적 벡터 $I_{f(t-1)->f(t-2)}$를 만들기 위해 사용된다. 그림 3의 상황을 예를 들어

$$I_{f(t-1)->f(t-2)} = \text{Median } (v1,v2,....v15,v16) \qquad (6)$$

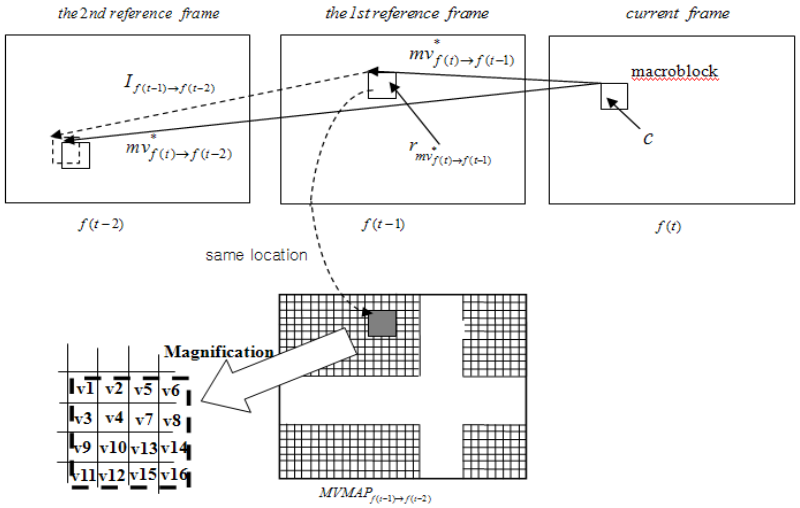


그림 3. 움직임 벡터맵을 이용하여 움직임 벡터 mv*f(t)->f(t-2) 를 예측하는 과정
Fig. 3. The proposed process to estimate mv*f(t)->f(t-2) using motion vector map.

Median 연산은 양극 혹은 단극의 임펄스 노이즈 벡터 존재제거를 위해 효과적인 방법이다. 때문에 Median 방식은 임펄스 노이즈 벡터에 영향을 받지 않은 $I_{f(t-1)->f(t-2)}$ 를 계산하는 데에 있어서 효과적으로 사용 될 수 있다. 다음으로 추정하는 움직임 벡터는 다음의 식으로 계산 된다.

$$PMV_{f(t)->f(t-2)} = I_{f(t-1)->f(t-2)} + mv^{*}_{f(t)->f(t-1)} \quad (7)$$

$PMV_{f(t)->f(t-2)}$ 는 기존의 MRME 방식에 의해 k=2 일 때 식(1)에 의해 예측 된 움직임 벡터 $mv^{*}_{f(t-1)->f(t-2)}$와 다를 것이다. 이러한 차이를 줄이기 위해 새로운 움직임 벡터는 $PMV_{f(t)->f(t-2)}$를 중심으로 탐색 영역 $S^{new}_{f(t-2)}$ 내에서 보정된다. 또한 탐색 영역 $S^{new}_{f(t-2)}$ 는 기존의 방식에서 사용되는 $S_{f(t-2)}$ 에 비해 매우 작기 때문에, 움직임 예측 과정의 복잡도는 크게 줄어 들게 된다.

제안된 알고리즘은 표 1 에 요약 된다. 첫 번째 참조 프레임 f(t-1)을 사용한 움직임 예측과정은 단계 1 에서 수행된다. 단계 2 에서, 예측된 움직임 벡터, $mv^{*}_{f(t)->f(t-1)}$ 들은 $MVMAP_{f(t)->f(t-1)}$ 에 저장 되고, 다음 단계에서 프레임 f(t+1)이 현재 프레임이 될 때에 이용된다. 단계4에서 추정되는 움직임 벡터 $PMV_{f(t)->f(t-k)}$ 는 움직임 벡터맵으로부터 계산되고 단계 5 에서 기존의 방식에서 사용되는 탐색 영역 $S_{f(t-k)}$ 보다 더 작은 영역인 $S^{new}_{f(t-k)}$ 내에서 보정된다. 단계 4 와 5 과정은(N-1) 개의 참조 프레임에 대하여 반복 된다. 단계 7에서 하나의 블록 모드에 대한 최적의 움직임 벡터가 단계 1과 5에서 예측된 $mv^{*}_{f(t)->f(t-k)}$, k=1,2,...N 들 중 비용함수 식 (1)을 최소화하는 벡터에 의해 결정된다. 이러한 과정들이 모든 가능한 블록 모드들에 대해 고려된 후 최적의 블록 모드는 단계 9에서 결정되고 움직임 예측 과정은 끝나게 된다.

f(t)가 현재 프레임일 때 $MVMAP_{f(t-1)->f(t-2)}$, $MVMAP_{f(t-2)->f(t-3)}$, $MVMAP_{f(t-3)->f(t-4)}$,…, $MVMAP_{f(t-N+1)->f(t-N)}$들은 각각 f(t-1),f(t-2),…,f(t-N+1)들이 현재 프레임이었을 때 구해지기 때문에 MVMAP 들은 참조 프레임 들의 모든 영역을 포함하게 된다. 만약 비디오 시퀀스에서 하나의 물체가 사라졌다가 나타나는 상황이 발생되면, 제안한 방식은 최소의 예측에러를 가지는 움직임 벡터를 선택한다. 심지어 예측에러가 크게 되더라도, 제안한 알고리즘은 적절하게 적용될 수 있다.

제안한 방법에서, N개의 참조 프레임들이 움직임 예측을 위해 사용될 경우, N-1개의 움직임 벡터맵들이 저장 되어야 한다. 만약 하나의 프레임 내에 있는 총 매크로 블록의 개수가 M이 라면, 제안한 방식은 벡터맵에 4×4 크기 블록에 대해 하나의 움직임 벡터를 저장 하기 때문에 하나의 벡터맵에 16×M개의 움직임 벡터를 저장하기 위한 메모리가 요구된다. 비록 제안한 알고리즘이 벡터맵 저장을 위한 메모리 공간과 추정하는 움직임 벡터 계산 과정이 필요하기는 하지만 향상된 부호화 속도에 의해 충분하게 보상 될 수 있다.

3.3 제안된 MRME 알고리즘의 복잡도 분석

제안한 MRME 알고리즘에서 요구되는 복잡도는 표 1의 단계 1과 5에서 크게 차지된다. 단계 1에서 블록 정합과정은 식 (1)에 의해 수행된다. 참조 프레임 f(t-1)을 위한 탐색영역 $S_{f(t-1)}$ 는 $(2W_{(t-1)}+1)\times(2H_{(t-1)}+1)$ 화소 크기 이기 때문에 단계 1의 움직임 예측 과정에서 $(2W_{(t-1)}+1)\times(2H_{(t-1)}+1)$만큼의 블록 정합이 요구된다. 단계 5에서 추정된 움직임 벡터는 크기가 작은 탐색영역 $S^{new}_{f(t-k)}$ 내에서 보정된다. 따라서 단계 5의 움직임 예측은 $(2W^{new}_{(t-k)}+1)\times(2H^{new}_{(t-k)}+1)$ 크기의 블록 정합 과정이 필요하다. 단계 5는 (N-1)개의 참조 프레임 f(t-k), k=2,3,...,에 대해 반복되기 때문에 요구되는 블록 정합 과정의 총 개수는 다음의 식으로 나타내어 진다.

$$Q\times M\times(2W_{(t-1)}+1)\times(2H_{(t-1)}+1)+ Q\times M\times\sum_{k=2}^{N}(2W^{new}_{f(t-k)}+1)\times(2H^{new}_{f(t-k)}+1) \quad (8)$$

M은 현재 프레임 내에 있는 매크로 블록의 총 개수이다. $(2W^{new}_{(t-k)}+1)$ 와 $(2H^{new}_{(t-k)}+1)$ 는 식 (3)의 $(2W_{(t-k)}+1)$ 와 $(2H_{(t-k)}+1)$ 보다 매우 작고, 과정은 $Q\times M\times(N-1)$만큼 반복 되기 때문에 제안한 방식은 MRME 모듈의 전체적인 복잡도에서 큰 이득을 제공 할 수 있다. 제안한 알고리즘에서 W^{new}, H^{new} 의 크기는 식 (6) 으로부터 계산되어진 PMV가 최적의 MV와 비슷하다는 가정 하에 최소 1, 최대 2로 사용될 수 있다.

Ⅳ. 실험결과

제안한 알고리즘의 성능을 입증하기 위해 다양한 영상을 이용한 컴퓨터 실험이 수행된다. 실험에서 2개의 QCIF (Car phone, Foreman)와 2개의 CIF (Mobile & Calendar, Tempete) 영상들이 이용된다. 참조프레임의 개수 N=5 로 고정되고 7가지 블록 모드 {16×16, 16×8, 8×16, 8×8, 8×4, 4×8, 4×4}들이 움직임 예측을 위해 사용된다. 실험에서 GOP (Group Of Pictures) 의 구조는 "IPPP…," 즉 첫 번째 프레임을 제외한 모든 프레임은 P 프레임으로 부호화 하였다. 새로운 탐색영역 $S^{new}_{f(t-k)}$의 크기는 QCIF 와 CIF 영상에 대해 각각 $H^{new}_{(t-k)},W^{new}_{(t-k)}=1$, $H^{new}_{(t-k)},W^{new}_{(t-k)}=2$ 로 주어지고, 기존의 탐색영역 $S_{f(t-k)}$ 는 QCIF 영상에 대해 $H_{(t-k)},W_{(t-k)}=16$, CIF 영상에 대해 $H_{(t-k)},W_{(t-k)}=32$ 의 크기로 사용하였다. 표 1의 단계 4의 과정이 N-1개의 참조프레임에 대하여 반복적으로 적용될 때, 벡터 $I_{f(t-k+1)->f(t-k)}$는 각 참조 프레임에 대한 추적 정보를 만들게 된다. 이러한 효과적이 움직임 벡터 추적 과정에 의해 추정된 벡터 $PMV_{f(t)->f(t-k)}$는 최적의 움직임 벡터 $MV_{f(t)->f(t-k)}$ 와 매우 유사하게 될 수 있게 된다. 때문에 단계 5에서 참조 프레임의 시간적 거리가 증가 될 때에 탐색 영역의 크기를 크게 할 필요가 없게 된다. 제안한 알고리즘의 효율성을 입증하기 위해 다양한 비트율과 프레임율이 테스트 영상을 부호화 하기 위해서 사용된다. 테스트 영상 Car phone 은 64 kbps (bits per second)와 10 fps (frames per second), Mobile & Calendar 는 128 kbps와 15fps, Tempete 는 256 kbps와 30 fps에서 각각 부호화 된다. 실험에서 사용된 비디오 코덱은 JM74[16]이다. 그림 4, 5, 6은 원본 영상과 부호화된 영상의 PSNR (Peak-to-Peak Signal to Noise Ratio) 을 보여준다.

테스트 영상들은 각각 2장에서 기술된 방법인 Conventional MRME와 제안된 MRME, Center-biased MRME[15], 그리고 하나의 참조 프레임만을 사용하는 SRME 방식으로 각각 부호화 된다. 실험에서 제안한 알고리즘은 효율성 입증을 위해 Conventional MRME, Center-biased MRME 와 함께 비교된다. Conventional MRME 방법은 모든 참조프레임에 대하여 전 탐색 영역 움직임 예측을 이용하는 방식이다. 이 방식은 탐색 영역 $H_{(t-k)}\times W_{(t-k)}$ 내에 있는 모든 가능한 위치를 검색함으로서 화질과 압축 효율 관점에서 최적의 해법을 주지만 많은 계산 량이 요구되는 단점을 가진다. Center-biased MRME[15]방식은 첫 번째로 움직임 벡터는 모든 참조 프레임의 중심위치로부터 작은 탐색영역 내에서 검색 된다. 다음으로 찾아진 움직임 벡터 중 가장 최적의 움직임 벡터를 가지는 참조 프레임을 선택하여 전 탐색 영역 움직임 예측을 수행한다. 이러한 방법은 선택된 참조 프레임이 지역 최적화된 움직임 벡터를 가지고 있다는 가정에 의한 것이다. 하지만 지역 최적화된 움직임 벡터를 가진 참조 프레임은 대부분 전역 최적화 된 움직임 벡터를 가지고 있지 않게 될 것이고 부호화 된 영상의 화질은 감소 되게 될 것이다. 본 논문에서는 움직임 벡터맵에 의해 계산된 추정된 움직임벡터를 이용한다. 계산된 추정벡터와 최적의 움직임벡터의 상관도는 매우 높기 때문에 제안된 알고리즘은 Center-biased MRME[15] 방식보다 더 좋은 성능을 나타낼 수 있다.

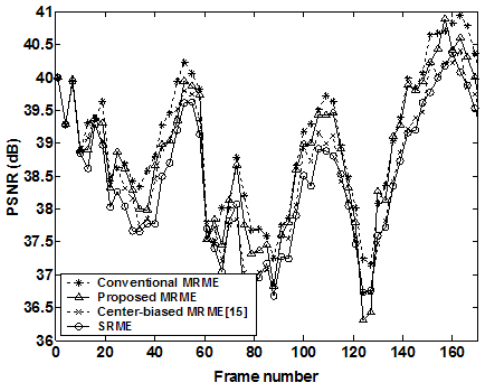


그림 4. 테스트 영상 QCIF 'Car phone' 에 대한 PSNR 그래프 (GOP=IPPP.., 64kbps, 프레임율rate=10Hz).
Fig. 4. PSNR for a QCIF 'Car phone'sequence, (GOP=IPPP.., 64kbps, Frame rate=10Hz).

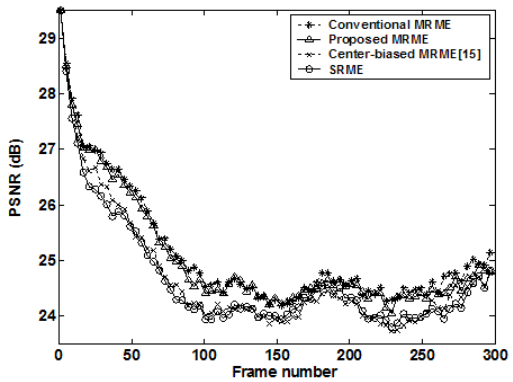


그림 5. 테스트 영상 CIF 'Mobile & Calendar' 에 대한 PSNR 그래프 (GOP=IPPP.., 128kbps, 프레임율 15Hz).
Fig. 5. PSNR for a CIF 'Mobile & Calendar' sequence, (GOP=IPPP.., 128kbps, Frame rate=15Hz).

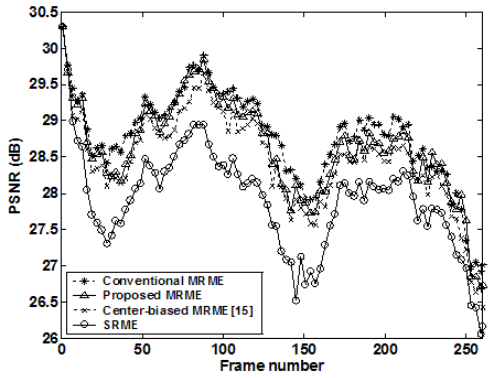


그림 6. 테스트 영상 CIF 'Tempete'에 대한 PSNR 그래프 (GOP=IPPP.., 256kbps, 프레임율 rate=30Hz).
Fig. 6. PSNR for a CIF 'Tempete' sequence, (GOP=IPPP.., 256kbps, Frame rate=30Hz).

그림 4, 5, 6에서 MRME 방식에 의해 부호화된 영상의 PSNR 은 SRME 방식에 의한 것 보다 더 높은 것을 확인 할 수 있다. 이 결과에서 H.264부호화기에서 다중 참조 프레임을 사용하는 것은 부호화 효율을 증가 시키는 것을 알 수 있다. 또한 제안한 MRME 방식의 PSNR은 Conventional MRME에 의한 결과와 매우 유사하고, Center-biased MRME 방식보다는 높게 나오는 것을 확인 할 수 있다. 이 결과는 제안한 방식이 Center-biased MRME 보다 화질 측면에서 더 우수하다는 것을 보여준다.

각 부호화 방식의 복잡도를 비교하기 위해 그림 7에서는 MVE(Motion Vector Estimation) 모듈에 의해 소비된 시간이 msec/frame 단위로 측정된다. 제안한 MRME 방식과 Conventional MRME 방식에 의해 소비된 복잡도의 비율은 다음의 식으로 나타낼 수 있다.

$$\frac{\begin{array}{l} Q\times M\times(2W_{(t-1)}+1)\times(2H_{(t-1)}+1)+ \\ \qquad Q\times M\times\sum_{k=1}^{N}(2W^{new}_{f(t-k)}+1)\times(2H^{new}_{f(t-k)}+1) \end{array}}{Q\times M\times\sum_{k=2}^{N}(2W_{(t-k)}+1)\times(2H_{(t-k)}+1)} \quad (9)$$

식에서 분자와 분모는 각각 식 (8)과 식 (3)으로부터 구해진다. QCIF 와 CIF 영상들이 부호화 될 때, $\{H^{new}_{(t-k)}=1,\ W^{new}_{(t-k)}=1,\ H_{(t-k)}=16,\ W_{(t-k)}=16\}$ 와 $\{H^{new}_{(t-k)}=2,\ W^{new}_{(t-k)}=2,\ H_{(t-k)}=32,\ W_{(t-k)}=32\}$ 들이 각각 식 (9)에 적용되면 비율은 약 21% 로 계산된다. 이 수치는 그림 7의 결과와 일치 한다, 또한 그림에서 제안한 방식은 Conventional MRME와 center-biased MRME[15] 방식보다 더 적은 복잡도를 요구함을 알 수 있다. 이 결과는 제안한 방식이 움직임 벡터맵과 시간적 예측 움직임 벡터등과 같은 과거 정보를 효과적 이용하는 반면 기존의 방식들은 이러한 정보를 이용하지 않기 때문이다. 실험 결과들에서 알 수 있듯이 제안한 알고리즘은 기존의 방식들과 비교 할 때 더 낮은 복잡도를 요구하는 반면 부호화된 영상의 화질은 유지됨을 알 수 있다.

제안된 방식의 유용성을 확인하기 위해, 율-왜곡 곡선 (Rate-Distortion curve)을 그림 8에 나타내었다. 그림에서 제안한 방식의 PSNR 은 다양한 비트율에서 Conventional MRME 방식과 매우 유사하고 MRME 방식을 사용할 경우 SRME 방식보다 더 우수함을 알 수 있다. 다양한 테스트 영상에 대해 움직임 예측 모듈에 의해 소비된 총 시간은 표 2에

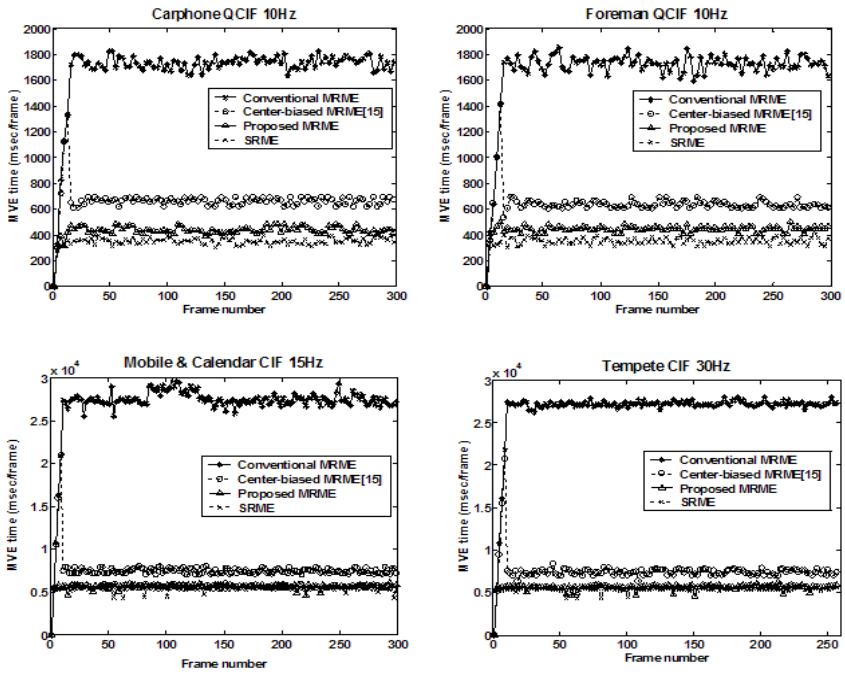


그림 7. 다양한 영상에 대해 H.264부호화기에서 움직임 예측 방식에 의해 소비된 CPU 시간 그래프.
Fig 7. Motion vector estimation time consumed by the H.264 encoders for various sequences.

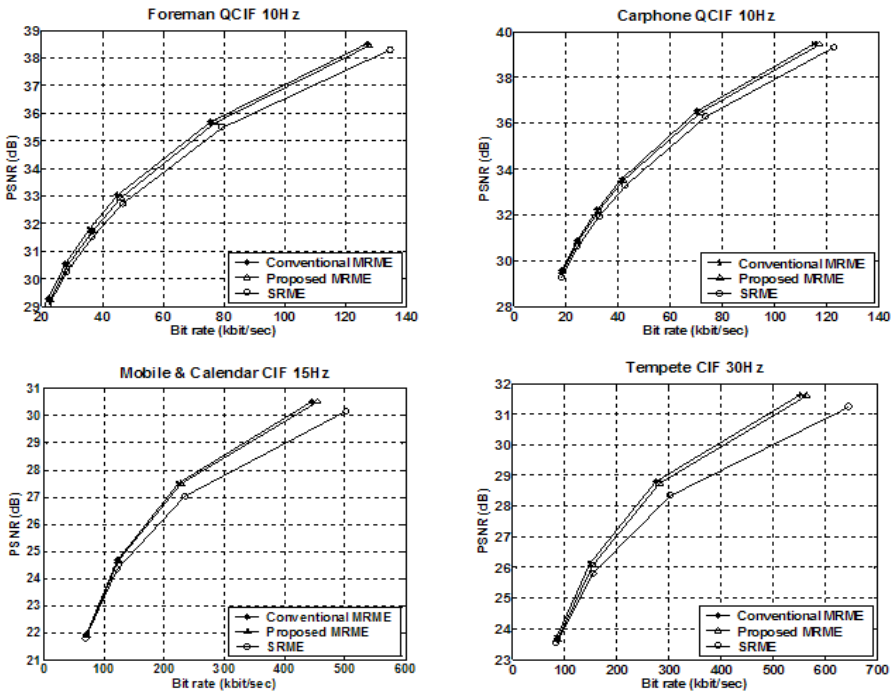


그림 8. 다양한 영상에 대한 율-왜곡 곡선.
Fig. 8. Rate-Distortion curves for various video sequences.

표 1. 제안하는 MRME 알고리즘.
Table 1. The proposed algorithm for MRME (Multi-reference Frame Motion Estimation).

N = The number of reference frames. A current frame is $f(t)$. $MVMAP_{f(t-k+1)\to f(t-k)}$, $k = 2,3,...,N$ are given by the scheme of Section III-A.
Step 0 : The mode $A\times B$ is one of {16×16,16×8, 8×16,8×8,8×4,4×8,4×4}. Step 1 : Estimation of $mv^{*}_{f(t)\to f(t-1)}$ for a frame $f(t)$ by (1) with $k=1$. Step 2 : Make a $MVMAP_{f(t)\to f(t-1)}$ from $mv^{*}_{f(t)\to f(t-1)}$ by the scheme of Section III-A, and save the $MVMAP_{f(t)\to f(t-1)}$ for a frame $f(t+1)$ which will be a current frame in the next phase. Step 3 : $k=2$. Step 4 : $PMV_{f(t)\to f(t-k)} = I_{f(t-k+1)\to f(t-k)} + mv^{*}_{f(t)\to f(t-k+1)}$ for all blocks in $f(t)$. Step 5 : Estimate $mv^{*}_{f(t)\to f(t-k)}$ by refinement of $PMV_{f(t)\to f(t-k)}$. Step 6 : If $k<N$, then $k=k+1$ and goto Step 4. Step 7 : Among the all $mv^{*}_{f(t)\to f(t-1)}$, $k=1,2,...,N$, $MV_{A\times B}$ is selected by minimizing cost function of (1). Step 8 : Unless all block types of {16×16,16×8, 8×16,8×8,8×4,4×8,4×4} are considered, then goto Step 1 with the next mode. Step 9 : Decide a mode by minimizing (2), and stop.

서 나타내었다. 제안한 방식과 Conventional MRME 방식의 복잡도는 아래의 식으로 비교된다.

(10)

이 결과는 제안한 방식이 H.264를 이용한 부호화 시간을 크게 단축됨을 보여준다. 이 장에서 보여진 실험 결과들은 제안한 방식과 Conventional MRME 방식에 의해 생성된 bitstream 의 화질은 매우 유사한 반면 제안한 방식은 Conventional MRME 방식에 비해 약 21% 의 복잡도 만을 요구함을 보여준다.

표 3 은 $A\times B$ 블록 크기에 대한 움직임 예측을 위해 요구되는 연산 횟수를 보여 준다. 식 (1)의 비용함수를 $A\times B$ 블록크기에 대해 계산하기 위해 $A\times B$ 의 곱셈과 $A\times B\times 3$의 덧셈이 요구된다. 또한 크기가 $(2W_{(t-k)}+1)\times(2H_{(t-k)}+1)$인 탐색영역을 사용할 때에 하나의 움직임 벡터를 예측하기 위해 요구되는 곱셈과 덧셈의 수는 각각 $A\times B\times(2W_{(t-1)}+1)\times(2H_{(t-1)}+1)$ 와 $A\times B\times 3\times(2W_{(t-1)}+1)\times(2H_{(t-1)}+1)$로 구해질 수 있다. 제안한 방식은 블록정합과정에서뿐만 아니라 추정하는 움직임 벡터를 계산하기 위해 추가적인 연산을 필요로 한다. 만약 L 이 median 연산을 위해 고려되는 요소벡터의 개수라면 median 벡터는 $L\times(L-1)$ 개의 덧셈연산과 한 번의 곱셈연산에 의해 구해진다. 비록 제안한 방식

표 2. 움직임 예측 모듈에 의해 소비된 총 시간 비교
Table 2. Total motion vector estimation time for various sequences.

Sequence	Total MVE time (sec) for a sequence		Reduction ratio (%)
	Conventional MRME	Proposed MRME	
Car phone (64kbps, 100 frames)	183.871	39.576	21.52
Foreman (64kbps, 100 frames)	181.561	37.800	20.81
Mobile & Calendar (128kbps, 150 frames)	4009.108	820.008	20.45
Tempete (256kbps, 260 frames)	6836.275	1421.359	20.79

표 3. 움직임 벡터를 예측하기 위해 요구되는 연산의 횟수비교.
Table 3. The number of arithmetic operations required to estimate a motion vector of a AxB block.

Function	The number of the required additions	The number of the required multiplications
Conventional MRME	$BM_{conventional}\times(A\times B\times 3)$	$BM_{conventional}\times(A\times B)$
Median operation for L element vectors	$L+(L-1)$	1
Proposed MRME	$\{BM_{proposed}\times(A\times B\times 3)\}+\{(N-1)\times(L+(L-1))\}$	$\{BM_{proposed}\times(A\times B)\}+(N-1)$

* Assumed that $BM_{A\text{scheme}}$ is the number of the required block matchings in a scheme.

$$BM_{conventional}=\sum_{k=1}^{N}(2\times H_{(t-k)}+1)\times(2\times W_{(t-k)}+1)$$

$$BM_{proposed}=(2\times H_{(t-1)}+1)\times(2\times W_{(t-1)}+1)+\sum_{k=2}^{N}(2\times H^{sch}_{(t-k)}+1)\times(2\times W^{sch}_{(t-k)}+1)$$

이 median 연산을 요구하지만, 블록 정합의 수는 기존의 방식들에 비해 크게 줄어들기 때문에 빠른 움직임 예측이 가능하게 된다.

V. 결 론

본 논문에서는 H.264부호화기에서 움직임 예측을 위한 효율적인 방법을 제안하였다. 제안한 방식은 세가지 단계, 움직임 벡터맵의 구성, 시간적 예측 움직임 벡터 생성, 움직임 벡터의 재정의 등으로 구성된다. 제안한 방식은 이전 단계에서 구성된 움직임 벡터맵으로부터 추정하는 움직임 벡터를 만들고 계산된 추정 벡터는 기존의 MRME 방식에 의해 예측된 움직임 벡터와 매우 유사하기 때문에 다중참조 프레임을 이용한 움직임 예측 과정에서 비트율과 화질을 유지시키면서 속도를 개선시키기 위해 사용될 수 있다.

다양한 컴퓨터 실험 결과들은 본 논문에서 제안한 알고리즘이 H.264부호화기에서 화질 손실 없이

소비되는 복잡도를 줄이는 것을 보여준다. 이러한 결과들은 제안한 알고리즘이 기존의 방식들 보다 우수하다는 것을 보여준다.

참 고 문 헌

〔1〕 A. Luthra, G.J. Sullivan, and T. Wiegand, "Introduction to the special issue on the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video technol,* vol. 13, pp. 557-559, July 2003.

〔2〕 I. E. G. Richardson, *H.264 and MPEG-4*, John Wiley & Sons Ltd, 2003.

〔3〕 T.Wiegand, G.J. Sullivan, G. Bjntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video technol,* vol. 13, pp. 560 - 576, July 2003.

〔4〕 T. Stockhammer, M.M. Hannuksela, and T. Wiegand, "H.264/AVC in wireless environments," *IEEE Trans. Circuits Syst. Video technol,* vol. 13, pp. 657-673 , July 2003.

〔5〕 S. Wenger, "H.264/AVC over IP," *IEEE Trans. Circuits Syst. Video technol,* vol. 13, pp. 645-656, July 2003.

〔6〕 D. Marpe, H. Schwarz, and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," *IEEE Trans. Circuits Syst. Video technol,* vol. 13, pp. 620-636, July 2003.

〔7〕 M. Wien, "Variable block-size transforms for H.264/AVC," *IEEE Trans. Circuits Syst. Video technol,* vol. 13, pp. 604-613, July 2003.

〔8〕 H.S. Malvar, A. Hallapuro, M. Karczewicz, and L. Kerofsky, "Low-complexity transform and quantization in H.264/AVC," *IEEE Trans. Circuits Syst. Video technol,* vol. 13 , pp. 598-603, July 2003.

〔9〕 T. Wiegand, H. Schwarz, A. Joch, F. Kossentini and G.J. Sullivan, "Rate-constrained coder control and comparison of video coding standards," *IEEE Trans. Circuits Syst. Video technol,* vol. 13, pp. 688-703, July 2003.

〔10〕 P. List, A. Joch, J. Lainema, G. Bjntegaard, and M. Karczewicz, "Adaptive deblocking filter," *IEEE Trans. Circuits Syst. Video technol,* vol. 13, pp. 614 - 619, July 2003.

〔11〕 M. Flierl and B. Girod, "Generalized B pictures and the drift H.264/AVC video-compression standard," *IEEE Trans. Circuits Syst. Video technol,* vol. 13, pp. 587 - 597, July 2003.

〔12〕 T. Wiegand, X.Zang, and B.Girod, "Long-term memory motion-compensated prediction," *IEEE Trans. Circuits Syst. Video technol,*, vol.9, pp. 70-84, Feb.1999.

〔13〕 A.Chang, O.C. An, and Y. M. Yeung, "A novel approach to fast multi-frame selection for H.264 video coding," in *Proc. IEEE Int. Conf. Acoustics, speech, Signal Processing ICASSP'03* , vol. 3, pp.III-413-416 , April 2003.

〔14〕 M. E. Al-Mualla, N. Canagarajah, and D. R. Bull, "Simplex minimization for multiple-reference motion estimation," *in Proc. IEEE Int. Symp. ISCAS'00*, vol. 4, Geneva, pp. 733 - 736, May 2000.

〔15〕 C. W. Ting, L. M. Po, and C. H. Cheung, "Center-biased frame selection algorithms for fast multi-frame motion estimation in H.264," *in Proc. 2003 Int. Conf. Neural Networks, Signal Processing*, vol. 2, pp. 1258 - 1261, Dec. 2003.

〔16〕 JVT codec reference software, http://iphome.hhi.de/suehring/tml/download/old_jm/jm74.zip

김 성 은 (Sung-Eun Kim) 정회원

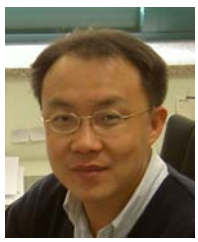
2004년 2월 세종대학교 정보통신공학과 공학사

2006년 2월 세종대학교 정보통신공학과 공학석사

현 재 (주) 엠큐브웍스 근무

<관심분야> H.264, Transcoder

한 종 기 (Jong-Ki Han) 종신회원



1992년 2월 KAIST 전기 및 전자공학과 학사

1994년 2월 KAIST 전기 및 전자공학과 석사

1999년 2월 KAIST 전기 및 전자공학과 박사

1999년 3월~2001년 8월 삼성전자 디지털 미디어 연구소 책임연구원

2001년 9월~현재 세종대학교 정보통신공학과 부교수

<관심분야> H.264, Transcoder, Scalable Video Codec (SVC)