움직임 벡터의 활동성과 방향성을 고려한 트랜스 부호화기의 비트율 제어

준회원 구 성 조*, 장 해 원*, 정회원 황 찬 식*

Bit-rate control of transcoder considering the activity and direction of motion vector

Sung-Jo Koo*, Hae-Won Jang* *Senior Members*, **Chan-Sik Hwang*** *Regular Member*

요 약

다양한 유무선 네트워크 환경상에서 실시간으로 보다 안정적이고 신뢰성 있는 주문형 비디오 및 실시간 멀티미디어 서비스를 사용자가 제공받기 위해서는 비디오 비트 스트림의 비트율을 다양한 채널의 대역폭에 맞추는 트랜스 부호화(transcoding) 과정이 필요하다. 본 논문에서는 트랜스 부호화 과정에서 움직임 벡터 추정시 매크로블록의 활동도와 방향성를 이용하여 계산량을 줄이면서 화질을 보장하는 방법을 제안한다. 매크로블록의 활동도와 방향성의 임계치는 복호화 과정에서 추출한 중간정보 중에서 DCT 계수를 사용하여 비디오마다 적합하게 적용이 가능하다. 그리고 제안된 비디오 트랜스 부호화 알고리즘을 이용하여 H.263 환경에서 영상의 비트율 변환을 통한 실험으로 적절한 수준의 화질 유지와 계산량의 감소를 확인할 수 있었다.

ABSTRACT

The transcoding is needed to modify the bit rate of the video bitstream in accordance with the bandwidth of a variety of channels in order to provide users with a realtime multimedia service and more stable and credible VOD(Video On Demand) service on various wire/wireless network. This paper suggests how to guarantee the video quality in the transcoding process, taking the activity and direction of the macroblock into consideration, thereby reducing the calculation. The threshold of the macroblock's activity and direction is applicable to the different videos, with the DCT coefficients from the intermediate data in the decoding process. Moreover, the suggested video transcoding algorism enables us to maintain the proper video quality and reduce the calculation through the video bit rate change experiment in H.263 environment.

Key Words : transcoding; bit rate; activity; direction; motion vector.

* 경북대학교 전자공학과 데이터 통신 시스템 연구실(jhw97@palgong.knu.ac.kr)
논문번호 : 030244-0609, 접수일자 : 2003년 6월 10일
※본 연구는 경북대학교 특성화 사업팀(KNURT) 연구과제 지원으로 수행되었습니다.

Ⅰ. 서 론

최근 화상 회의, 주문형 비디오, mobile phone,원격 강의 같은 네트워크 상의 멀티미디어 서비스들이 크게 증가하고 있다. 현재 제공되는 대부분의 비디오 서비스와 멀티미디어 응용에서는 미리 부호화 되어있는 비디오를 사용한다. 이러한 응용 서비스를 다양한 네트워크에 접속된 사용자가 이용하기 위해서는 비디오 비트 스트림의 비트율을 다양한 채널의 대역폭에 맞추는 것이 필요하다. 만약 10 Mbits/s로 부호화 되어있는 비디오를 대역폭이 좁은 채널로 그대로 전송한다면 심각한 화질의 열화가 발생한다. 그러므로 높은 비트율로 부호화 되어있는 비디오를 제한된 대역폭을 가지는 채널로 보낼 때에는 적절한 비트율로 바꾼 후 전송해 주어야 하며 이러한 과정을 트랜스 부호화(transcoding)라고 한다 [1].

최근 무선 네트워크 환경의 제약 즉, 낮은 대역폭, 이동 컴퓨터의 처리 능력 제한, 적은 메모리 공간, 디스플레이 크기 제약 등으로 인해 트랜스 부호화의 필요성이 증대되고 있다. 특정 비트율로 부호화 되어있는 비디오를 원하는 비트율로 다시 변환하기 위해서는 복호화 한 후 다시 부호화 과정을 수행해야 되기 때문에 이에 따른 계산량의 증가와 더불어 전송시간에 문제가 발생하게 된다. 일반적으로 비디오를 부호화하는 과정 중에서 계산량이 가장 많은 부분이 움직임 추정 부분이므로 이 부분에서 계산량을 줄인다면 트랜스 부호화를 수행하는데 소모되는 시간을 크게 줄일 수 있다.

움직임 추정 시 계산량을 줄이기 위해 입력 비트스트림으로부터 추출해낸 기저 움직임 벡터(base motion vector)를 그대로 재 사용하는 방법이 있다. 그러나 단순히 기저 움직임 벡터를 그대로 사용하게 되면 양자화 에러에 의해 영상의 열화가 발생하게 된다. 그래서 이러한 문제점을 해결하기 위해 기저 움직임 벡터를 그대로 사용하지 않고 정제(refinement)과정을 수행하는 방법이 있다. 정제 과정을 거친다면 움직임 추정에 소요되는 계산량을 상당히 줄이면서 영상의 화질 열화를 방지할 수 있다[2]. 그러나 매크로블록의 특성을 고려하지 않고 모든 매크로블록에 대해서 정제 과정을 수행한다면 비효율적이다. 왜냐하면 트랜스 부호화하기 전의 움직임 벡터와 트랜스 부호화를 수행한 후의 움직임 벡터가 같거나 비슷하다면 이러한 경우에는 움직임 추정을 할 필요가 없기 때문이다. 그러므로 기저 움직임 벡터를 재사용해도 되는 경우와 그렇지 않은 경우를 구별하는 것이 중요하다. 또한 비트량 감소로 인해 프레임 스킵이 발생하게 된다. 그렇기 때문에 생략된 프레임에 대해 움직임 벡터의 추정이 중요하다[3].

본 논문에서는 매크로블록의 활동도를 이용해서 트랜스 부호화에서의 움직임 추정 시에 기저 움직임 벡터를 재 사용할 것인지 아닌지를 정하고 매크로블록의 활동도가 비슷한 경우 움직임 벡터의 방향성을 고려하여 추정하는 방법을 제안한다.

Ⅱ. 트랜스 부호화

일반적으로 transcoding은 아래와 같은 과정을 수행하게 된다.

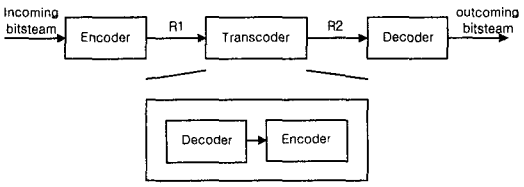

그림 1. 직렬 트랜스 부호화기
Fig. 1. Cascaded Transcoder

R1(bits/s)으로 부호화된 비디오를 R2(bits/s)로 부호화하기 위해서는 우선 이 비디오를 복호화한 후 다시 한번 더 부호화 과정을 수행해야한다. 그림 1에서 제시된 cascade로 연결된 트랜스 부호화기[4-6]는 몇 가지 문제점을 가지고 있다. 첫 번째는 수행과정이 복잡하다는 것이고 두 번째는 시간이 오래 걸린다는 것이다. 전자와 후자의 경우 모두가 부호화 과정을 다시 수행하기 때문에 그러한 단점이 발생하게 된다. 그러나 트랜스 부호화의 특성상 부호화 과정을 생략할 수는 없으므로 과정을 간략하게 할 수 있는 방법이 필요하게 되었다.

1. 트랜스 부호화에서 움직임 추정

부호화하는 과정 중에서 가장 계산량이 많고 시간이 오래 걸리는 부분이 움직임 추정부분이다. 기본적으로 움직임을 추정하는 방법은 전 탐색영역 추정(full search motion estimation)이다.

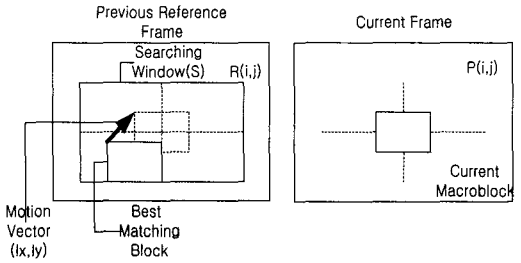

그림 2. 움직임 추정 (전 탐색영역)
Fig. 2. Motion estimation (Full search)

현재 매크로블록(macroblock)의 움직임 벡터를 구하기 위해서는 식 (1)과 (2)에서처럼 움직임 벡터를 구하고자하는 현재 프레임의 매크로블록과 searching range 내의 이전 프레임의 블록간의 각 픽셀 값의 차의 절대값의 합(SAD)을 구해서 그 중에서 가장 작은 블록을 찾아서 두 블록간의 움직임 거리를 움직임 벡터로 정한다[2].

$$(I_x, I_y) = \arg\min_{(m,n)\in S} SAD(m,n) \tag{1}$$

$$SAD(m,n) = \sum_{i}^{i+16}\sum_{j}^{j+16} |P(i,j) - R(i+m, j+n)| \tag{2}$$

여기서 m과 n은 비교하고자하는 블록의 수직, 수평 위치를 나타내고 $P(i,j)$는 현재 프레임의 픽셀들을 나타내고 $R(i,j)$는 이전의 재생된 프레임의 픽셀들을 나타낸다.

이러한 방법으로 움직임 벡터를 구한다면 정확하게 움직임 벡터를 구할 수는 있으나 탐색 영역이 (-15, 15) 만큼 되어서 계산량이 많아진다.

2. 움직임 벡터의 정제

최적화된 움직임 벡터는 트랜스 부호화기의 인코더에서 움직임 추정에 의해서 얻을 수 있으나 이는 많은 계산량을 요구하기 때문에 바람직하지 않다. 그래서 디코더에서 추출해낸 움직임 벡터를 이용해서 움직임 벡터를 구하는 방법이 사용되어왔다. 그런데 비트율의 변화에 의한 양자화 에러의 영향 때문에 그대로 추출해낸 움직임 벡터를 사용할 수는 없고 이 벡터를 이용해서 새롭게 움직임 벡터를 찾아 내어야한다. 식 (3)과 (4)의 식에서와 같이 추출한 움직임 벡터 (B_x, B_y)를 기준으로 전 탐색 움직임 추정보다 작은 탐색 영역에서 최적화된 움직임 벡터를 얻을 수 있다[2].

$$(D_x, D_y) = \arg\min_{(m,n)\in small\ S} SAD(m,n) \tag{3}$$

$$SAD(m,n) = \sum_{i}^{i+16}\sum_{j}^{j+16} |P(i,j) - R(i+B_x+m, j+B_x+n)| \tag{4}$$

일반적으로 생략된 프레임에서의 움직임 벡터를 정확하게 구하면 정제 과정에서 탐색 영역의 크고 작음에 상관없이 비슷한 화질을 나타낸다. 따라서 정제 과정에서 계산량을 상당히 줄일 수 있다. 본 논문에서는 실험적으로 정제 과정의 탐색 영역을 (-1, 1)로 정하여 사용하였다[7].

3. 기존의 방법들

움직임 추정에 있어서 보다 정확하게 움직임 벡터를 찾고 계산량을 줄이기 위해서 기존의 몇 가지 방법들이 제안되었는데 이 중에서 가장 성능이 좋은 방법이 ADVS(Activity Dominant Vector Selection)이다[8].

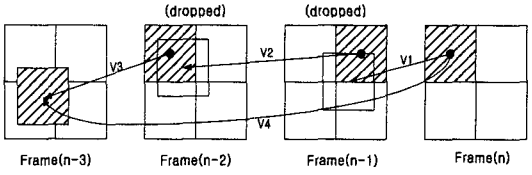

그림 3. 스킵된 매크로블록에서의 움직임 추정
Fig. 3. Motion estimation in skipped macroblock

그림 3에서 Frame(n)은 현재 프레임, Frame(n-3)은 이전 프레임이고 Frame(n-1)과 Frame(n-2)는 생략된 프레임이다.

현재 프레임의 첫 번째 매크로블록은 움직임 벡터 V_1만큼 Frame(n-1)로 이동해 갔을 때 Frame(n-1)의 이웃하는 4개의 매크로블록 중에서 활동도가 가장 큰 매크로블록의 움직임 벡터를 선택한다. 여기서 이 두 번째 매크로 블록의 움직임 벡터 V_2만큼 Frame(n-2)로 이동해가고 다시 이웃하는 매크로블록 중에서 활동도를 고려해서 움직임

벡터를 선택하여 Frame(n-3)으로 이동해간다. 결국 현재 프레임 Frame(n)에서 이전 프레임 Frame(n-3)으로 이동해 갈 때 움직임 벡터는 $V_4 = V_1 + V_2 + V_3$가 되는 것이다. 그리고 Frame(n-3)에서 V_4를 기준으로 하여 특정 탐색 영역으로 Frame(n)의 첫 번째 매크로블록과 SAD(Sum of Absolute Differences)가 가장 작은 부분을 선택한다.

활동도에 따라 움직임 벡터를 추정하는 과정은 그림 4에 나타나 있다.

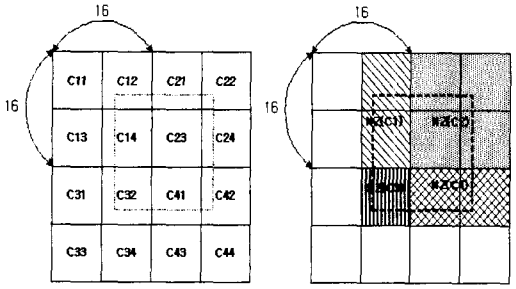

그림 4. ADVS 알고리즘
Fig. 4. Illustration of ADVS algorithm

현재 프레임이 매크로블록의 움직임 벡터만큼 이동해 왔을 때 매크로블록들을 8×8 블록들로 분해한 후 0이 아닌 AC 계수의 개수가 가장 많은 매크로블록의 움직임 벡터를 선택한다.

$$
\begin{aligned}
&NZ(C_1) = NZ(C_{12}) + NZ(C_{14}) \\
&NZ(C_2) = NZ(C_{21}) + NZ(C_{22}) + NZ(C_{23}) + NZ(C_{24}) \\
&NZ(C_3) = NZ(C_{32}) \\
&NZ(C_4) = NZ(C_{41}) + NZ(C_{42}) \\
&NZ(\cdot) : 0\text{이 아닌 } AC \text{ 계수의 개수} \\
&C_{ij} : i\text{번째 매크로블록의 } j\text{번째 블록} \\
&\qquad\qquad (i,j = 1,2,3,4)
\end{aligned}
\tag{5}
$$

그런데 여기서 고려해야 할 점은 과연 모든 프레임의 매크로블록에 대해서 움직임 추정을 해야하고 또한 정제 과정을 거쳐야만 하느냐는 것이다. 다시 말해서 추출해낸 움직임 벡터를 그대로 사용해도 되는 경우가 있을 것이다. 이러한 경우 계산량이 상당히 줄어들어 매우 효율적일 것이다.

또한 기존의 ADVS 방법에서는 0이 아닌 AC 계수의 개수를 비교하여 활동도를 결정하였는데 단순히 매크로블록의 활동도만 가지고 움직임 벡터를 만들어 내는 것은 문제가 발생할 수 있다. 예를 들어 배경의 움직임과 물체의 움직임이 반대 방향이고 블록의 활동도가 비슷한 경우 구하려고 하는 물체의 움직임과는 반대되는 배경의 움직임 부분을 선택할 수도 있다. 이를 방지하기 위해 이전 프레임의 움직임 벡터의 방향을 고려하여 움직임 벡터를 만들어 내는 것이 바람직할 것이다.

III. 제안한 방법

1. 매크로블록의 활동도에 따른 움직임 벡터 추정 과정 생략

트랜스 부호화에서 사용할 수 있는 중간 정보에는 움직임 벡터, 매크로블록 타입, DCT 계수, 양자화 파라미터 등이 있다. 이러한 것들 중에서 매크로블록의 활동도를 나타내는 요인으로는 움직임 벡터와 DCT 계수 등이 있다[9-10]. 즉 DCT 계수 중에서 DC값을 뺀 나머지 0이 아닌 AC 계수(에너지)의 개수가 작은 값을 가진다면 이것은 이전의 인코더에서 움직임 추정을 할 때 적절한 움직임 벡터 값을 구한 것이 되고, 또 그 매크로블록의 활동성이 작은 것을 알 수 있다. 그러므로 만약 매크로블록의 AC 계수의 에너지가 작은 값을 가진다면 움직임이 거의 없는(예를 들면 배경부분) 매크로블록이므로 이러한 매크로블록들은 움직임 추정과 정제 과정을 거치지 않고 추출한 움직임 벡터를 재사용해도 영상의 화질에는 크게 영향을 미치지 않을 것이다. 반면에 계산량은 상당히 줄어 실시간 전송에 있어서 효과가 있을 것이다.

즉, 그림 3에서 현재 프레임의 매크로블록의 에너지가 특정 임계치보다 작은 경우에는 움직임 벡터 추정 과정이 생략될 뿐만 아니라 정제 과정 또한 탐색 영역이 (-1, 1)로 간략하게 되어 계산량에서 상당한 이득을 얻을 수 있으며 다음 식(6)으로 나타낸다.

$$
V_4 = \begin{cases} V_1 + V_2 + V_3, & \text{if Energy} > \text{threshold} \\ V_1, & \text{otherwise} \end{cases}
\tag{6}
$$

2. 움직임 벡터의 방향성 고려

기존의 ADVS 방법에서는 0이 아닌 AC 계수의 개수를 비교하여 활동도를 결정하였는데 단순히 매크로블록의 활동도만 가지고 움직임 벡터를 만들어

내는 것은 문제가 발생할 수 있다. 왜냐하면 인접한 매크로블록들의 활동도가 비슷하다면 어떤 것이 dominant하다고 판단하기가 쉽지 않다. 즉 에너지가 dominant하지 못하다면 단순히 작은 수치적인 차이에 의해서 움직임 벡터를 선택하는데는 오류가 따를 것이다. 대부분의 경우에 있어서 매크로블록의 움직임은 이전 매크로블록과 비슷한 방향으로 이동할 가능성이 크다. 그러므로 움직임 벡터의 방향성을 이용하여 움직임 추정을 한다면 원, 타원형, 또는 지속적인 움직임의 영상에서 매우 효과적일 것이다. 따라서 매크로블록들 간에 0이 아닌 AC 계수의 개수 차이가 거의 없는 경우에는 이전 프레임의 움직임 벡터의 방향을 고려하여 기저 움직임 벡터를 만들어 내는 다음과 같은 방법을 제안한다.

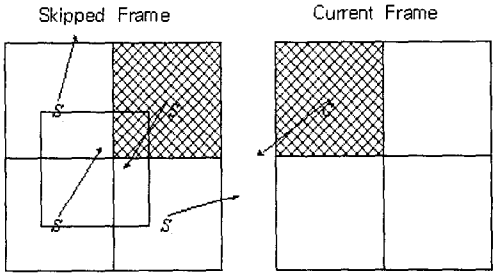

그림 5. 움직임 벡터의 방향성을 고려한 방법
Fig. 5. The method of motion vector direction

모든 움직임 벡터의 방향은 8방향으로 나누고 두 방향의 경계치 근처에 있을 때에는 두 방향 모두를 고려하여 방향성의 범위를 넓히도록 한다.

처음 ADVS 방법에 의해 선택한 매크로블록의 0이 아닌 AC 계수의 개수가 다른 매크로블록에 비해 dominant하지 못하다면 즉, 임계치보다 작다면 움직임 벡터의 방향성을 고려한다. 방향성을 고려하는 방법은 그림 5에서와 같이 $S_1 \sim S_4$ 각 매크로블록 중 현재 프레임의 움직임 벡터와 방향이 같은 매크로블록을 선택하여 이들 중에서 에너지가 가장 큰 매크로블록의 움직임 벡터를 선택한다. 이 방법에 의해서는 비록 S_3이 가장 활동도가 크다 하더라도 dominant하지 못하다면(활동도의 차이가 임계치보다 작다면) 이전 프레임의 움직임 벡터와 방향이 비슷한 S_2가 선택될 가능성이 가장 크다. 이렇게 하면 단순히 0이 아닌 AC 계수의 개수만을 비교했을 때 나오는 오류를 보완할 수 있을 것이다.

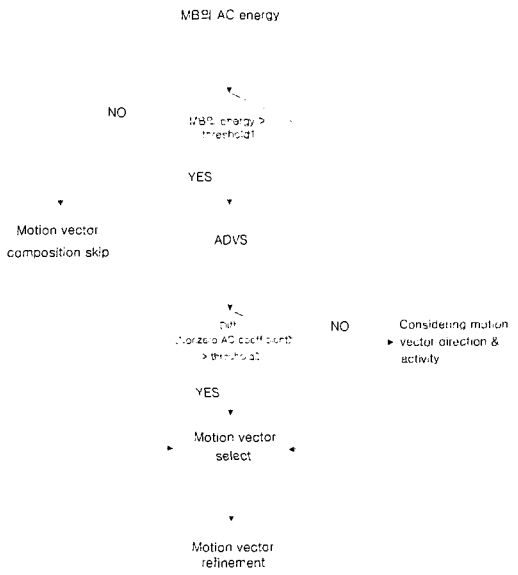

그림 6. 제안한 방법의 순서도
Fig. 6. Flow chart of proposed method

그림 6은 제안한 방법의 전체적인 순서도이다. 그림에서 보는 것과 같이 활동도와 방향성을 고려하기 위해서 각각 임계치1과 임계치2를 사용하였다. 활동도의 척도는 현재 프레임의 매크로블록의 에너지 즉, 0이 아닌 AC 계수의 개수를 사용하였다. 또한 이 에너지를 이용하여 방향성 임계치의 척도로써 사용하였다.

Ⅳ. 실험 및 고찰

실험에서 사용한 시퀀스는 'susie', 'stefan', 'table tennis' 의 QCIF 영상과 'bbc', 'children' 의 CIF 영상들이다. 실험은 H.263환경에서 비트율 변환에 근거하여 ADVS 방법과 함께 제안한 방법을 비교하여 화질과 계산량을 기준으로 성능 평가를 하였다. QCIF 영상은 128kbps로 부호화된 비트스트림을 64kbps로 변환하였고, CIF 영상은 512kbps로 부호화된 비트스트림을 256kbps로 변환하여 비교하였다. 기저 움직임 벡터를 재 사용할 것인지 아닌지와 방향성을 고려할지의 임계값은 영상 특성에 맞게 차별적으로 적용하기로 한다.

실험 결과를 좀더 자세히 살펴보기 위해 실험 시퀀스 중 stefan과 children 영상의 각 프레임에 대한 PSNR의 값을 그림 7과 8에 나타내었다.

표 1. 비트율 변환시 각 방식 결과의 평균 PSNR
Table 1. Results of different methods through the bit rate change

	ADVS	제안한 방법	Number of skipped MB / total MB
susie	37.50	37.42	321/1287 (25%)
stefan	25.57	25.63	549/2673 (21%)
table tennis	31.86	31.87	450/1386 (32%)
bbc	28.53	28.95	1084/3564 (30%)
children	32.11	32.13	7041/11088 (64%)

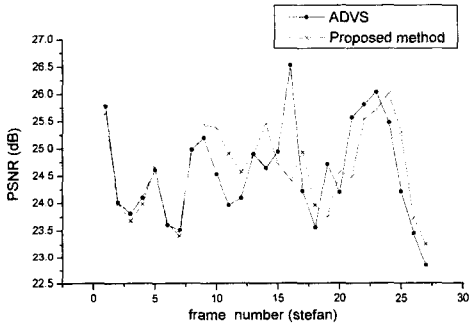

그림 7. stefan 영상의 화질 비교 (128kbps → 64kbps)
Fig. 7. Video quality comparison of stefan (128kbps → 64kbps)

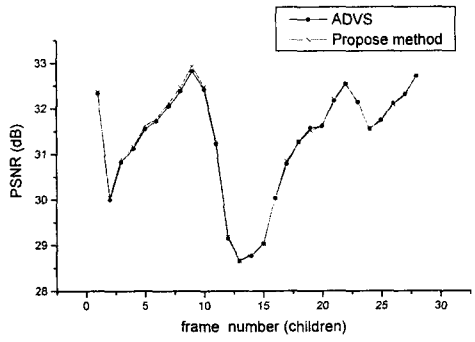

그림 8. children 영상의 화질 비교 (512kbps → 256kbps)
Fig. 8. Video quality comparison of children (512kbps → 256kbps)

표 1에서 살펴보면 기저 움직임 벡터를 재 사용하는 경우가 총 매크로블록의 거의 절반 이상이어도 PSNR은 거의 떨어지지 않는 것을 볼 수 있다. 그리고 각 방법에 대한 계산량을 비교해 보았다. 화질뿐만 아니라 트랜스 부호화 과정 시 걸리는 시간도 중요한 요소가 되기 때문에 PSNR과 함께 비교 요소로 설정하였다.

인접한 4개의 매크로블록에서 움직임 벡터를 선택하는 과정을 살펴보면 제안한 방법에 ADVS보다 약간의 계산 과정이 더 첨가되어 있다. 매크로블록의 에너지 차이가 많이 나지 않을 경우 움직임 벡터의 방향성을 고려하는 알고리즘 때문이다. 그러나 이러한 단순 비교 과정은 계산량에 큰 영향을 주는 것은 아니다. 실제로 계산량을 좌우하는 것은 덧셈이나 곱셈이다. 즉, 움직임 벡터를 선택하는 알고리즘에서의 계산량 증가는 전 탐색 영역으로 정제 과정을 수행하지 않고 (-1, 1)의 간략화 된 탐색 영역으로 정제 과정을 수행함으로써 전체 계산량의 관점에서 본다면 무시할 수 있을 만큼의 값이 된다.

제안한 방법에 의해 기저 움직임 벡터를 재 사용함으로 인한 계산량의 감소 정도를 살펴보기 위해 프레임 당 계산량을 표 2에 나타내었다. M은 프레임당 매크로블록의 개수이고, N은 생략되는 매크로블록의 개수를 의미한다.

표 2. 프레임당 계산량
Table 2. Computation per frame

	정제 과정 (add=1, shift=1, mult.=3)	복잡성
ADVS	±4 pels, 62208a per 16×16 block	62208×M
제안한 방법	±4 pels, 62208a per 16×16 block	62208×(M-N)

기존 ADVS의 움직임 추정 방법과 제안된 방법의 전체 계산량은 식 (7)로 비교해 볼 수 있다.

전체계산량
= 총 프레임 중 정제과정을 거치는 MB의 개수
× 탐색 영역에서의 계산량 (7)

각각의 실험 비디오들에 대해 제안된 방법에 의한 전체 계산량을 표 3에 나타내었다. 기존의 ADVS 방법에 의한 계산량을 100%로 정하고 이와 비교해서 제안한 방법의 계산량을 나타내었다.

표 3. 전체 프레임에 대한 계산량 (%)
Table 3. Computation for total frames (%)

	비디오 시퀀스에 대한 전체 계산량			
	susie	stefan	table tennis	children
ADVS	8.0×10^{4}	16.7×10^{4}	8.6×10^{4}	69.0×10^{4}
제안한 방법	6.0×10^{4} (25%)	13.2×10^{4} (21%)	5.8×10^{4} (33%)	25.2×10^{4} (64%)

위의 결과를 통해서 살펴보면 제안한 방법은 기존의 움직임 추정 방법과 비교하여 화질에서는 거의 열화가 없이 전체 계산량에서 약 30% 이상 개선된 것을 볼 수 있다. 이것은 동일한 화질을 유지하면서 수행 시간의 감소로 인해 유무선 네트워크 환경에서 데이터의 실시간 처리에 있어서 상당한 이득을 얻을 수 있을 것이다.

V. 결 론

현재 유무선 네트워크 환경에서 제공되는 대부분의 비디오 서비스와 멀티미디어 응용 서비스를 사용자가 받아볼 때 다양한 채널의 대역폭에 맞게 적절한 비트율로 바꾸어주어야만 화질의 열화가 적게 발생한다. 이를 위해서 여러 가지 방법들이 제안되었는데 본 논문에서는 트랜스 부호화 과정에서 움직임 벡터 추정 시 계산량을 줄이면서 보다 정확하게 추정할 수 있는 방법을 제안하고 있다.

움직임 벡터 추정 시 활동도가 작은 매크로블록은 복호화 과정에서 추출한 기저 움직임 벡터를 재사용하고, 또한 전 탐색 영역보다 훨씬 작은 (-1, 1)의 탐색 영역으로 정제 과정을 거치면 화질에 큰 영향 없이 계산량을 줄일 수 있다. 그리고 보다 정확한 움직임 추정을 위하여 움직임 벡터의 방향성까지 고려하였다. 본 논문에서는 그 판단 기준으로 복호화 과정에서 뽑아낼 수 있는 중간 정보 중 매크로블록의 AC 계수들의 에너지를 사용하였다. 이러한 방법으로 트랜스 부호화를 실행한다면 계산량이 작기 때문에 실시간 전송이 가능하다. 또한 AC 계수의 에너지를 이용해서 임계치를 적용하므로 비디오마다 적합하게 적용할 수 있는 장점이 있다.

제안한 방법의 타당성을 제시하기 위해 기존의 움직임 추정 기법인 ADVS 방법과 비교하여 실험하였다. 성능 평가 방법은 전체 프레임의 평균 화질과 움직임 추정 부분의 전체 계산량으로 하였으며 그 결과 제안한 방법이 ADVS와 비교하여 유사한 화질에 계산량에 있어서 약 30%이상의 이득을 보였다. 이것은 유무선 네트워크 환경에서의 실시간 처리 능력에 향상을 기할 수 있을 것이다.

참 고 문 헌

[1] M. Ghabari, "Two-layer coding of video signals for VBR networks", *IEEE Journal on Selected Areas in Communications*, vol. 7, pp. 771-781, June 1989.

[2] Jeongnam Youn, Ming-ting Sun, Chia-Wen Lin, "Motion Vector Refinement for High Performance Transcoding", *IEEE Trans. on Multimedia*, Vol. 1, No. 1, pp. 30-40, Mar 1999

[3] Kai-Tat Fung, Yui-Lam Chan, and Wan-Chi Siu, "New architecture for dynamic frame-skipping transcoder", *IEEE Trans. on Image Processing*, Vol. 11, No. 8, pp. 886-900, Aug. 2002

[4] N. Bjork and C. Christopoulos, "Transcoder Architecture for Video Coding", *IEEE Trans. Consumer Electron*, Vol. 44, pp. 88-98, Feb. 1998

[5] G. Keesman, R. Hellinghuizen, F. Hoeksema, G. Heidema, "Transcoding of MPEG bitstreams", *Signal Processing : Image Communication 8*, pp. 481-500, 1996

[6] P. A. A. Assuncao, M. Ghanbari, "Optimal transcoding of compressed video", *IEEE Int. Conf. on Image Processing 1997*, Vol. 1, pp. 739-742, Oct. 1997

[7] Tamer Shanableh, M. Ghabari, "Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding fromats", *IEEE Trans. on Multimedia*, Vol. 2, No. 2, pp. 101-110, June 2000.

[8] Mei-Juan Chen, Ming-Chung Chu, and Chih-Wei Pan, "Efficient Motion-Estimation Algorithm for Reduced Frame-Rate Video Transcoder", *IEEE Trans. on Circuits and Systems for Video Technology*, Vol.4, pp. 269-275, April 2002

[9] Bo Shen, Ishwar K. Sethi, Bhaskaram Vasudev "Adaptive Motion-Vector Resampling for

Compressed Video Downscaling", *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 9, No. 6, pp. 929-936, Sept. 1999

[10] Kai-Tat Fung, Wan-Chi Siu, "DCT-based video frame-skipping transcoder", *Proc. Int. Symposium on Circuits and Systems 2003*, Vol. 2, pp. 656-659, May 2003

구 성 조(Sung-Jo Koo) 준회원

2000년 2월 : 경북대학교 전자공학과 졸업

2002년 2월 : 경북대학교 전자공학과 석사

2002년 3월~현재 : 경북대학교 전자공학과 박사과정

<주관심분야> 영상신호처리, 트랜스코딩

장 해 원(Hae-Won Jang) 준회원

2002년 2월 : 경북대학교 전자공학과 졸업

2002년 3월~현재 : 경북대학교 전자공학과 석사과정

<주관심분야> 영상신호처리, 트랜스코딩

황 찬 식(Chan-Sik Hwang) 정회원

1977년 2월 : 서강대학교 전자공학과 졸업

1978년 2월 : 한국과학기술원 석사

1996년 2월 : 한국과학기술원 공학박사

1980년 3월~현재 : 경북대학교 전자공학과 교수

<주관심분야> 데이터 통신, 영상신호처리, 암호통신