論文2003-40SP-2-3

움직임 특성을 이용한 새로운 고속 움직임 예측 방법

(A New Fast Motion Search Algorithm Using Motion Characteristics)

李 誠 鎬 *, 魯 大 榮 *, 張 祐 演 *, 吳 承 埈 *, 安 昌 範

(Sung-Ho Lee, Dae-Young No, Ho-Yun Jang, Seoung-Jun Oh, and Chang-Beom Ahn)

요 약

최근 들어 ASIC(Application Specific IC)이나 소형 시스템에서 사용할 수 있는 더 빠르고 정확한 움직임 벡터 예측방법이 요구되고 있다. 전역탐색(Full Search: FS) 방법은 탐색영역의 모든 화소들을 탐색하여 움직임 벡터를 예측하는 방법으로 화질과 PSNR은 좋지만 반면에 많은 계산량이 요구된다. 기존의 고속 알고리즘들은 탐색 회수를 제한함으로써 계산량을 줄였기 때문에 움직임 벡터 예측의 정확도가 낮고, 움직임 보상시 SAD(Sum of Absolute Difference) 값이 높아지는 것을 감수해야한다. 본 논문에서는 영상에서의 현재 블록과 주변 블록과의 공간적인 상관도를 고려하여 예측된 움직임 벡터 (Predicted Motion Vector: PMV)를 이용하는 고속 움직임 탐색 방법을 제안한다. PMV 방법은 주변 블록의 움직임 벡터를 이용한 기존 방법들 보다 명확하고 간결하게 탐색을 수행할 수 있다. PMV 방법이 대표적인 기존 방법인 Nearest-Neighbors Search(NNS) 방법보다 속도 및 정확도 면에서 성능이 양호함을 대표적인 실험 시퀀스를 통하여 보였다.

Abstract

Recently we need a faster and more accurate motion vector search algorithm for ASIC(Application Specific IC) or small systems. Block motion estimation using Full Search(FS) algorithm provides the best visual quality and PSNR, but it requires intensive computations. The previously proposed fast algorithms reduced the number of computations by limiting the number of searching locations. This is accomplished at the expense of less accuracy of motion estimation and gives rise to an appreciably higher SAD(Sum of Absolute Difference) for motion compensated images. In this paper we exploit the spatial correlation of motion vectors and present a fast motion estimation scheme which uses the predicted motion vector(PMV). The PMV scheme is more clear and simpler than the previously proposed algorithms which also use adjacent motion vectors. Simulation results with standard video sequences show that the PMV scheme is faster and more accurate than other algorithms such as Nearest-Neighbors Search(NNS) algorithm.

Keywords : BMA, Motion estimation, Motion vector prediction, H.263, MPEG-4

* 正會員, 光云大學校 電子工學科

(Dept. of Electronics Engineering, Kwangwoon Univ.)

※ 이 논문은 한국과학재단 목적기초연구(R01-2002-000-00179-0) 지원에 의해 수행되었습니다.

接受日字:2002年9月25日, 수정완료일:2003年3月10日

Ⅰ. 서 론

컴퓨터가 보편화되고 인터넷이 활성화되면서 멀티미디어 컨텐츠를 쉽게 접할 수 있게 되었고 제작도 보편화되고 있다. 요즘 모든 방송국의 웹 사이트(Web site)

에서는 고속 인터넷 서비스와 향상된 멀티미디어 스트리밍 서비스를 통하여 일반인에게 VOD(Video On Demand) 서비스를 지원하고 있다. 또한 차세대 휴대통신서비스인 IMT-2000에서는 기존 휴대 통신기에서 제공하지 못했던 동영상서비스를 지원하고 있다. 이런 영향으로 비디오 영상에 대한 실시간 코딩이 더욱더 요구된다.

영상압축은 디지털 비디오를 전송하고 저장하기 위하여 가장 핵심이 되는 기술이다[1]. 720×420 인 NTSC 비디오 신호를 압축하지 않을 경우 249×2^{20} 비트가 필요하다. 이러한 방대한 정보를 화상회의와 같은 실시간 서비스에 적용하는 것은 불가능하다. 따라서 MPEG, H.26x와 같은 압축방식이 필요하다[2-7]. 동영상 압축 표준방식에서는 비디오 신호의 시간상의 중복성을 제거함으로써 데이터를 압축하기 위하여 움직임 보상방식(Motion Compensation: MC)을 사용한다. MC를 위하여 비디오 영상이 가지는 움직임 벡터(Motion Vector: MV) 값을 예측하는 것이 요구된다[1]. 영상압축 인코더에서 가장 많은 계산량을 요구하는 부분이 움직임벡터를 추정하는 부분이다. 따라서 실시간으로 비디오 영상을 압축하려면 움직임 벡터를 추정하기 위하여 요구되는 계산량을 매우 줄여야 한다.

일반적인 고속 움직임 벡터 예측 알고리즘으로는 TSS(Three-Step Search)[8], LOGS(2-D Logarithm Search)[9], OSA(Orthogonal Search Algorithm)[10], CS(Cross-Search Algorithm)[11] 등이 있다. 이 방법들을 사용하면 FS보다 탐색점이 줄어들기 때문에 정확도가 다소 감소하지만 탐색 속도를 월등히 향상시킬 수 있다. NTSS(New Three-Step Search) 방법[12]과 4SS(Four-Step Search) 방법[13]은 움직임 벡터가 중앙에 치우치는 특성과 초기 탐색 종료 조건을 이용하여 기존의 고속 알고리즘들보다 정확도와 속도를 향상시켰다.

움직임 벡터의 시공간적인 상관도를 이용하는 방법으로는 PM(Prediction Model) 탐색 방법[14]과 NNS(Nearest-Neighbors Search) 방법[15] 등이 대표적이다. 이 방법들은 움직임 벡터의 상관도를 이용한다는 점에서 다른 고속 알고리즘보다 발전된 형태이며, 속도뿐만 아니라 정확도면에서도 좋은 성능을 보여 준다. PM 방법은 기존의 NTSS와 4SS에서 (0,0) 주변에서 탐색하던 것을 PMV(Predicted Motion Vector) 주변에서 탐색하도록 변형한 것이다. NNS 방법은 PMV를 중심으로 하여 작은 영역에서 세밀하게 탐색을 하는 것이 특징이다. NNS 방법에는 2가지 탐색 경로가 있는데 주변 블록의 코딩 모드에 따라 TSS에서 제안한 방법을 택하는 경로와 NNS에서 자체적으로 제안한 경로로 나누어진다. NNS 방법은 움직임 탐색 과정에서 선험적인 결과를 기반으로 여러 가지 경우를 정하고 이에 따라 MV를 탐색하기 때문에 브랜치(Branch) 명령들이 필요하다. 브랜치 명령은 병렬처리(Parallel processing)에서 성능을 감소시키는 주 원인이기 때문에 최근 개발되고 있는 비디오 처리용 DSP 아키텍처에 구현하기에 부적합하다고 할 수 있다. 이러한 DSP 칩들은 내부적으로 병렬처리에 기반을 두고 있기 때문이다[16,17].

본 논문은 NNS 방법에서와 같이 움직임 벡터의 예측값으로 주변 블록의 움직임 벡터들에 대한 중간값을 이용하고, 그 중간값을 TSS 방법의 첫 번째 탐색점 집합의 원소로 추가하여 중간값으로 보상된 위치의 블록에 대한 SAD 값과 나머지 후보 탐색점들의 SAD 값에 따라 탐색 경로를 결정한다. 주변 블록의 코딩 모드로 탐색 경로를 선택하는 방법보다 브랜치 명령의 수를 현저하게 줄이고 과정을 단순화시킴으로써 화질을 유지하면서 속도를 향상시킬 수 있다.

Ⅱ. 고속 움직임벡터 예측 알고리즘

고속 알고리즘은 대부분 탐색점의 수를 줄여나간다. 대표적인 알고리즘인 TSS 방법과 NNS 방법은 다음과 같다.

1. TSS 방법

TSS 방법[8]은 가장 일반적인 고속 움직임 예측 알고리즘이다. <그림 1>은 ±7의 탐색 영역에 대한 TSS 방법을 나타내고 있고 각 과정은 다음과 같다.

Algorithm TSS

/* step size = $\lceil d/2 \rceil$ 로 초기화한다. (단, d 는 최대 움직임 변위이다.) */

1단계: 초기 step size로 아홉 군데에서 SAD 값을 계산한다.

2단계: 최소 SAD 값의 위치를 구한다.

3단계: step size를 반으로 줄인 후 SAD 값이 최소되는 점을 중심으로 8개의 SAD 값을 구한다.

4단계: 이 과정을 3회 반복한다.

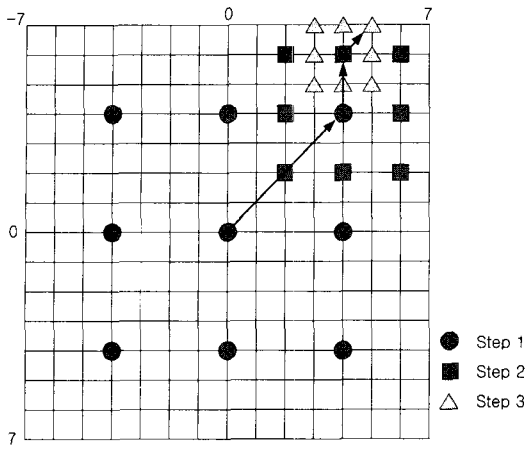


그림 1. TSS 방법의 예
Fig. 1. An example for the TSS scheme.

2. NNS 방법

NNS 방법[15]은 기존의 고속알고리즘의 단점인 PSNR 측면을 크게 향상시킨 알고리즘이다. 계산량은 TSS 방법과 거의 유사하나 PSNR은 기존의 고속 알고리즘보다 0.5dB~1dB 정도 높다.

<표 1>은 NNS 방법에서 주위 블록의 코딩 모드에 따른 PMV 및 탐색 경로(Search Path)를 나타내고 있다. NNS 방법은 주위 블록의 코딩 모드에 따라 2가지 모드로 나뉘게 된다. NNS 방법의 첫 번째 모드는 현재 블록의 왼쪽과 위쪽 블록인 MB0와 MB1이 인트라 모드(INTRA mode)인 경우로서 TSS 방법과 동일한 과정을 수행한다. 두 번째 모드는 주변 블록이 인터 모드(INTER mode)로 코딩되었을 때 주변 블록으로부터 PMV를 구하고 PMV를 중심으로 최근접(Nearest-Neighbors) 탐색을 하는 경우이다.

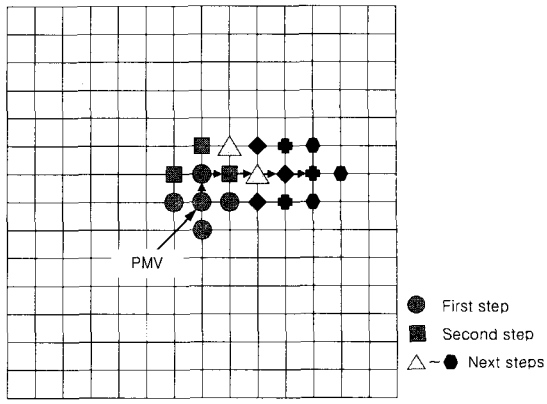


그림 2. NNS(두 번째 모드)
Fig. 2. NNS(second mode).

<그림 2>는 NNS 방법의 두 번째 모드를 나타내고 있다. 두 번째 모드의 경우 PMV를 중심으로 다이아몬드 탐색(Diamond Search)을 한다. 이때 다이아몬드 탐색점 중에서 가운데 탐색점이 최소 SAD 값을 가진다면 탐색을 중지하고 그 점을 움직임벡터로 설정한다.

Ⅲ. 제안된 방법

움직임 탐색에 있어서 시퀀스의 움직임 특성에 따라 TSS 방법과 NNS 방법의 성능이 달라질 수가 있다. 본 논문에서 제안하는 방법은 시퀀스의 움직임 특성에 따라 광역 검색과 국부 검색을 수행한다. 광역 검색으로는 TSS 방법을 사용하고, 국부 검색은 PMV를 중심으로 하는 다이아몬드 탐색(Diamond Search) 방법을 사용한다.

다음으로, 광역 검색과 국부 검색을 판단하는 기준이 필요하다. 먼저, TSS 방법 3단계까지 모든 탐색점에

표 1. NNS 탐색 경로
Table 1. NNS Search Path.

코딩 모드			Predicted Motion Vector	탐색경로
MB0	MB1	MB2		
I	I	x	(0,0)	Three-Step
I	P	x	MV1	Nearest-Neighbors
P	I	x	MV0	Nearest-Neighbors
P	P	I	Mean(MV0,MV1)	Nearest-Neighbors
P	P	P	Median(MV0,MV1,MV2)	Nearest-Neighbors

주) 1. I: INTRA, P: INTER, x: don't care
2. MB0 ~ MB2의 위치는 <그림 4> 참조
3. MV0는 MB0의 MV를 의미.

대한 SAD 값과 PMV에서의 SAD 값을 비교하는 방법을 생각할 수 있다. 이것은 TSS 방법 3단계까지의 모든 탐색점에 대해서 SAD 값을 계산해야 하므로 많은 연산량이 필요하여 적합하지 않다. 연산량을 고려하여 TSS 방법의 1단계 또는 2단계까지의 탐색점에 대한 SAD 값과 PMV에서의 SAD 값을 비교하여 탐색 방법을 정하는 것을 생각할 수 있다. 본 논문에서는 TSS 방법 1단계 까지의 탐색점에 대한 SAD 값과 PMV에서의 SAD 값을 비교하여 탐색 방법을 선택한다.

<표 2>는 CIF 크기의 시험 시퀀스들 각각에서 90개 프레임에 대하여 TSS 방법 1단계에서 취하는 9개 탐색점에 대한 최소 SAD 값과 PMV에 의한 SAD 값을 비교하여 PMV의 SAD 값이 더 작거나 같은 횟수를 측정한 결과이다. 각 프레임의 이 횟수를 불규칙 변수 r로 표시하면, 각 시퀀스에서 r에 대한 평균값을 구할 수 있고, 이 값을 r에 대한 기대값 $E[r]$로 대치할 수 있다. 그러므로 움직임 벡터를 예측하기 위한 첫 과정에서 PMV를 통해 얻은 탐색점이 TSS의 9개 탐색점보다 정확한 탐색점 후보라는 것을 R_{pmv}로 표시하면, R_{pmv}는 식 (1)로 표시할 수 있다.

$$R_{pmv} = \frac{E[r]}{N_{MB}} \times 100 \qquad (1)$$

식 (1)에서 N_{MB}는 프레임 당 매크로블록의 수이다. CIF 크기 시퀀스에서 N_{MB}=396이다. 일반적으로 R_{pmv} 값이 70% 이상이고, 움직임이 적은 경우에는 거의 100%에 근접함을 알 수 있다. 그러므로 TSS 1단계에서 고려하는 9개 탐색점에 PMV로 예측된 탐색점을 추가하여 움직임 벡터 검색을 수행하면 단순하면서도 정확한 움직임 벡터를 적은 계산량으로 구할 수 있을 것으로 예상할 수 있다.

표 2. 실험 시퀀스에 대한 R_{pmv} 값
Table 2. The values of Rpmv in test sequences.

시퀀스 종류	$E[r]$	R_{pmv}(%)
Stefan	304	76.8
Coastguard	302	76.3
Foreman	278	70.3
Table tennis	255	64.4
Container	394	99.4
Akiyo	392	98.9

<그림 3>은 제안하는 방법의 순서도를 나타낸다. 주요 과정은 다음과 같다.

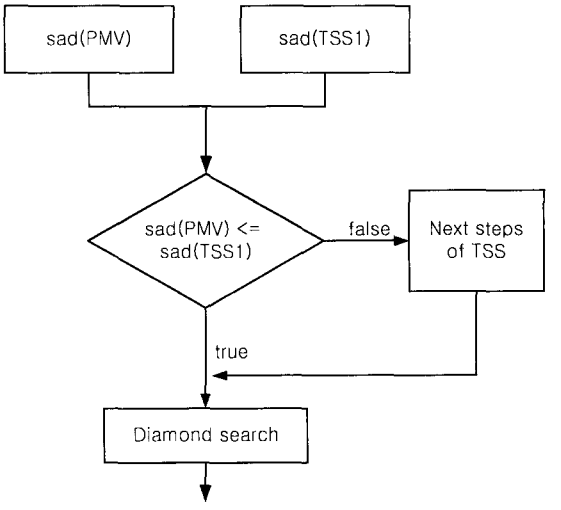


그림 3. 제안하는 방법의 순서도
Fig. 3. Flow chart of the proposed scheme.

1. PMV 구하기

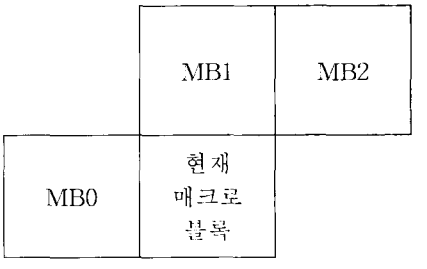


그림 4. 참조할 주변 매크로 블록
Fig. 4. Adjacent macroblocks

<그림 4>는 현재 매크로 블록과 주위의 매크로 블록들을 나타내고 있다. MB0, MB1, MB2는 각각 왼쪽, 상단, 오른쪽 상단에 위치한 매크로 블록이다. MBi (단 i=0,1,2)에 대한 PMV를 구하는 방법으로는 움직임이 적은 자연 영상에 대해서 식 (2)와 같이 중간값을 사용하는 것이 일반적으로 가장 좋은 방법으로 알려져 있다[15].

$$PMV = median(MV0, MV1, MV2) \qquad (2)$$

동영상 압축표준인 H.263+ 과 MPEG-4 에서는 움직임 벡터의 차이값을 인코딩하는데 예측값으로 MBi 들의 중간값을 사용한다. 즉, MBi 들의 중간값과의 차이

를 인코딩한다. 본 논문에서는 별도로 PMV를 구하지 않고 이 예측값을 PMV로 사용한다.

2. PMV의 신뢰도 판정

PMV가 현재 블록의 움직임 벡터와 항상 유사하다고는 할 수 없다. 예를 들면, 장면 전환이 발생했거나 블록의 위치가 물체의 경계면일 때, 또는 물체의 출현 및 소멸이 발생했을 때는 PMV는 현재 블록의 움직임 벡터와 크게 다를 수도 있다. 즉, 유사성이 없을 수도 있다. 따라서 PMV의 현재 블록의 움직임 벡터와의 유사성 즉, PMV의 신뢰도를 판단하는 규칙을 어떻게 정하느냐가 중요한 문제가 된다. PMV의 신뢰도를 판단하기 위해서 NNS 방법에서는 MB*i* 각각의 코딩 모드로 판단을 하였다.

<그림 3>의 순서도는 본 논문에서 제시하는 PMV의 신뢰도 판정 및 그에 따른 2가지 탐색경로를 나타내고 있다. PMV의 신뢰도 판정은 다음과 같다. 먼저, PMV에서의 SAD 값 sad(PMV)를 구하여 TSS 방법 1단계에서 9개 탐색점에 대한 SAD 값 sad(TSS1)과 비교한다. sad(PMV)가 더 작거나 같으면 PMV는 신뢰도가 있다고 판단하고, 그렇지 않다면 PMV는 신뢰도가 없는 것으로 판단한다.

PMV의 신뢰도에 따라 2가지 탐색경로를 갖는다. PMV가 신뢰도가 있으면 PMV 주변 영역에서 세부 검색을 수행하고, 그렇지 않으면 일반적인 TSS 탐색의 나머지 과정을 수행한다. 세부 검색으로는 3.3장에서 설명할 다이아몬드 탐색을 사용한다. 두 번째 경우에서 TSS의 나머지 과정이 끝난 후에도 다이아몬드 탐색을 하는데, 이것은 여러 영상에 대해 실험해본 결과 대체적으로 TSS 탐색 후에 다이아몬드 탐색을 해주는 것이 속도에 큰 영향을 미치지 않으면서 성능향상을 다소 얻을 수 있기 때문이다.

3. 다이아몬드 탐색

센터 주위의 상, 하, 좌, 우 4개의 탐색점을 탐색한다. 센터에서 SAD 값이 최소이면 탐색을 종료하고, 그렇지 않으면 그 점으로 이동하여 다시 주변 4개의 탐색점을 탐색하는데 최대 4번까지 반복하여 계산량 증가를 막는다. <그림 5>는 다이아몬드 탐색 과정을 나타내고 있다.

<그림 6>은 ±7의 탐색영역에 대한 제안된 방법의 2가지 탐색 경로를 나타내고 있다. <그림 6>에는 화살표가 아래로 향한 경우와 위로 향한 경우 2가지 경로가 있다. 화살표가 아래로 향한 경로는 PMV가 신뢰도를 갖는 경우이고 화살표가 위로 향한 경로는 PMV가 신뢰도가 없는 경우를 나타낸다. <그림 6>에는 표시되지 않았으나 화살표가 위로 향한 경로에서 TSS 과정 후 다이아몬드 탐색을 수행하는 것도 포함된다.

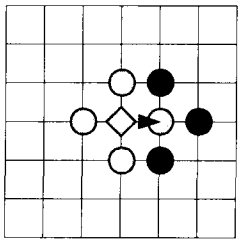


그림 5. 다이아몬드 탐색 방법
Fig. 5. The diamond search scheme.

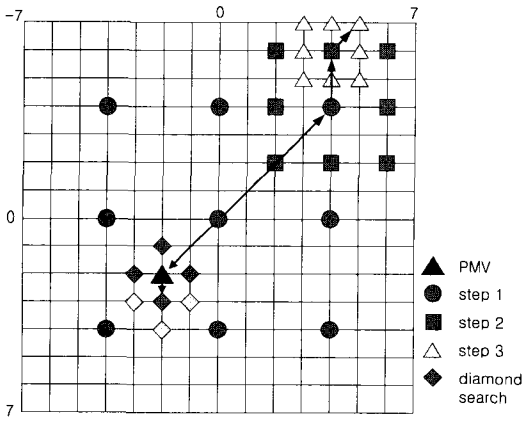


그림 6. 제안된 방법
Fig. 6. Proposed scheme.

Ⅳ. 실험 결과

H.263+ 부호화기의 움직임 예측 모듈을 기존의 방법과 제안하는 방법으로 대치하여 각 방법의 성능을 비교하였다. 사용한 비디오 시퀀스는 H.263과 MPEG-4에서 성능 검사를 위해서 가장 보편적으로 사용하는 움직임이 많은 Stefan 과 Table tennis, 움직임이 약간 있는 Coastguard 와 Foreman, 그리고 움직임이 적은 Container ship 비디오 시퀀스이다. 각 비디오 시퀀스에 대하여 CIF 크기를 사용하고 30fps로 인코딩한다. 탐색영역은 ±15 로 하였고 각 방법 모두 반화소(half-pel) 탐색을 수행한다. I와 P 프레임에 대해서 양자화 계수

QP는 13을 사용한다. 각 방법에 대한 성능은 비트율, 인코더의 움직임 벡터 모듈에서 걸린 시간(TIME), 휘도 성분의 PSNR 값 그리고 휘도성분에 대한 Q 값인 Q(Y)[18]로 표시한다. Q 값은 PSNR과 마찬가지로 원영상에 대한 영상의 시각적 화질을 나타내는 척도인데 영상의 주관적인 화질을 반영하는 값이다. Q값의 범위는 [-1,1]이고 원영상과 같을 때 값이 1이다. [18]에서는 Q를 상관도 손실(loss of correlation), 휘도 성분 왜곡(luminance distortion), 콘트라스트 왜곡(contrast distortion) 등 3가지 성분의 곱으로 표시한다. 본 실험에서는 상관도 손실과 휘도 성분 왜곡 성분의 곱으로 Q를 표시한다. 콘트라스트 왜곡은 영상 강화와 관련된 성분이기 때문에 영상의 주관적 화질을 반영하지 못한다고 판단되어 본 고에서 제외하였다.

<표 3~7>은 각 실험 비디오 시퀀스에 대하여 30fps로 인코딩한 경우에 대한 성능 평가 결과이다.

표 3. Stefan (CIF 크기 90frame사용, 30fps)
Table 3. Stefan (CIF size, 90frame, 30fps).

구 분	FS	TSS	NNS	Proposed
비트율(kbps)	761.29	956.50	800.21	791.97
TIME(ms/frame)	261.91	35.81	30.26	28.81
PSNR(Y)	29.78	29.75	29.74	29.75
Q(Y)	0.7567	0.7491	0.7505	0.7478

표 4. Table tennis (CIF 크기 90frame사용, 30fps)
Table 4. Table tennis (CIF size, 90frame, 30fps).

구 분	FS	TSS	NNS	Proposed
비트율(kbps)	411.87	534.24	461.27	451.05
TIME(ms/frame)	389.41	38.71	31.92	34.27
PSNR(Y)	31.16	31.01	31.08	31.04
Q(Y)	0.5562	0.5237	0.5441	0.5289

표 5. Coastguard (CIF 크기 90frame사용, 30fps)
Table 5. Coastguard (CIF size, 90frame, 30fps).

구 분	FS	TSS	NNS	Proposed
비트율(kbps)	531.78	574.08	531.55	531.01
TIME(ms/frame)	325.12	36.16	29.16	30.01
PSNR(Y)	29.86	29.83	29.86	29.85
Q(Y)	0.7715	0.7686	0.7717	0.7700

표 6. Foreman (CIF 크기 90frame사용, 30fps)
Table 6. Foreman (CIF size, 90frame, 30fps).

구 분	FS	TSS	NNS	Proposed
비트율(kbps)	255.64	281.13	246.56	256.54
TIME(ms/frame)	275.92	36.71	30.71	30.71
PSNR(Y)	32.49	32.20	32.29	32.21
Q(Y)	0.6254	0.6162	0.6220	0.6141

표 7. Container (CIF 크기 90frame사용, 30fps)
Table 7. Container (CIF size, 90frame, 30fps).

구 분	FS	TSS	NNS	Proposed
비트율(kbps)	131.87	130.83	130.78	130.54
TIME(ms/frame)	330.23	36.14	30.38	27.27
PSNR(Y)	31.85	31.83	31.83	31.82
Q(Y)	0.5092	0.5033	0.5047	0.4993

제안된 방법은 TSS 방법과 비교해서는 모든 비디오 시퀀스에서 비트율 및 속도면 에서 우수한 성능을 나타냈다. 따라서 제안된 방법이 TSS 방법에 비해서는 확실히 우수한 방법임을 알 수 있다.

NNS 방법과 비교해서는 비트율 및 속도면 에서 고려해볼때 실험 비디오 시퀀스의 종류에 따라서 성능이 다르다는 것을 알 수 있다. 비트율 면에서 Stefan, Table, Coast guard, Container 시퀀스에서는 제안하는 방법의 성능이 더 우수하였고 Foreman 시퀀스에서는 NNS 방법이 더 우수하였다.

속도측면에서는 Stefan, Container 시퀀스에서는 제안하는 방법이 NNS 방법보다 속도가 빨랐으며, Table tennis, Coast guard 시퀀스에서는 NNS 방법이 빨랐다. Foreman 시퀀스에서는 속도가 같았다. PSNR과 Q면에서는 비트량 증감의 영향으로 모든 비디오 시퀀스에서 유사한 수치를 나타냈다.

표 8. NNS 방법과의 성능 비교
Table 8. Performance comparison between NNS and Proposed schemes.

구 분	Stefan	Table	Coast	Foreman	Container
비트율	P	P	P	N	P
속도	P	N	N	X	P

각 비디오 시퀀스에 대해서 비트율과 속도 면에서 어느 방법이 더 우수하였는지를 <표 8>에 정리하였다. P는 제안된 방법이고 N은 NNS 방법을 의미하고 X는 두 방법의 성능이 같은 것을 의미한다.

<그림 7>은 Stefan 시퀀스에 대한 탐색 방법들의 프레임별 비트량을 비교한 것이다. TSS 방법이 가장 많은 비트를 발생시켰고, FS 방법이 가장 적은 비트를 발생시킨다. 제안한 방법은 NNS 방법과 유사한 결과를 보였고 FS와 큰 차이를 보이지 않는 것을 알 수 있다.

<그림 8>은 Stefan 비디오 시퀀스에 대해서 제안된 방법과 NNS 방법과의 프레임별 처리속도를 비교하고 있다. 프레임 23과 35, 프레임 59와 71 구간에서 제안하는 방법이 NNS 방법보다 비교적 속도가 빠른 것을 알 수 있다. 비디오 시퀀스의 어떤 부분에서 제안하는 방법의 성능이 우수한지를 알아보기 위해서 Stefan 비디오 시퀀스의 프레임 23, 35, 59, 71을 <그림 9>에 나타냈다. 프레임 23과 35 사이는 주로 Stefan의 팔이 수직 방향으로 움직였다. 프레임 59와 71 구간에서는 카메라가 왼쪽으로 이동하는 것이 특징이다. 물체의 수평 이동이나 카메라의 이동 같은 움직임을 갖는 비디오 시퀀스에 대해서 제안한 방법이 NNS 방법보다 빠르게 움직임 벡터를 탐색한다는 것을 알 수 있다.

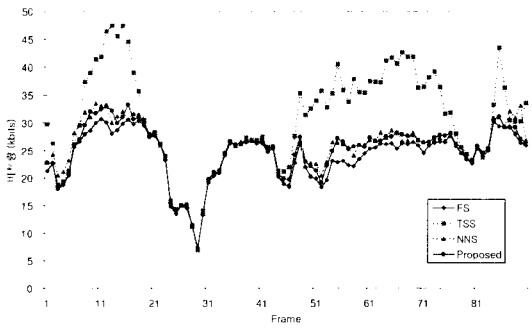


그림 7. Stefan 시퀀스에 대한 비트량 측면에서의 탐색 방법 비교

Fig. 7. Comparison of search schemes in terms of bits per frame: Stefan.

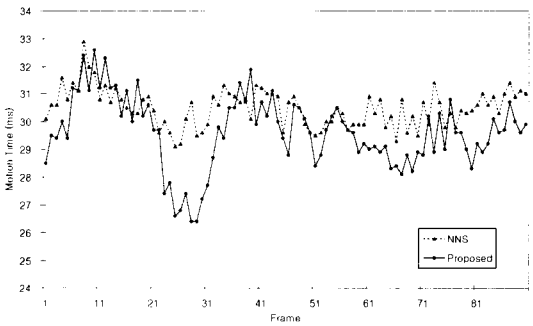
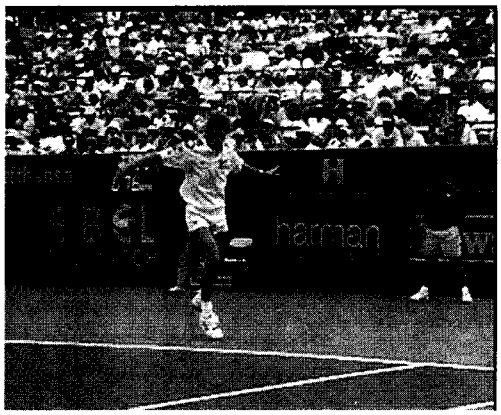
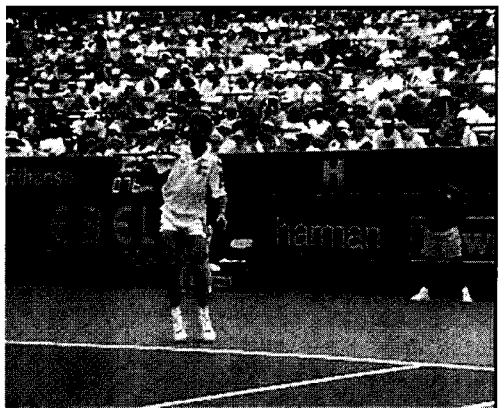


그림 8. Stefan 시퀀스에 대한 움직임 벡터 탐색 속도 측면에서의 탐색 방법 비교

Fig. 8. Comparison of search schemes in terms of motion vector search times: Stefan.

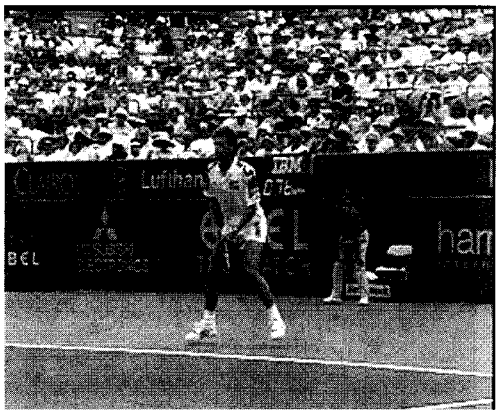


(a) 프레임 23



(b) 프레임 35

(c) 프레임 59



(d) 프레임 71
그림 9. Stefan 시퀀스의 각 프레임
Fig. 9. Frames in the Stefan sequence.

V. 결 론

본 논문에서는 주위의 움직임벡터를 이용한 고속 움직임 벡터 탐색 방법을 제안하였다. 일반적인 영상에서 주위 블록의 움직임 벡터들은 현재 블록의 움직임 벡터와 공간적인 유사성이 많다. 이러한 움직임 벡터의 공간적인 상관도를 이용하면 영상 압축에 있어서 압축률을 높일 수 있을 뿐만 아니라 속도 개선도 이룰 수 있다.

제안하는 방법의 경우 모든 실험 비디오 시퀀스에서 기존의 TSS 방법에 비해서 비트율과 속도 면에서 우수한 성능을 나타냈다. NNS 방법과는 비트율 면에서는 대체적으로 제안하는 방법이 우수하였으며 속도 면에서는 비디오 시퀀스의 종류에 따라 서로 비슷한 속도를 나타냈다. NNS 방법에 비해 프로그램구조가 간단하며 부가적으로 필요한 메모리 요구량이 없기 때문에 구현상에 있어서도 효과적이다.

이러한 결과에 따라 본 논문에서 제시하는 방법은 크기가 작은 저 비트율의 영상뿐만 아니라 고화질 영상의 압축에도 적용시킬 수 있으며 또한 속도의 향상으로 인해 실시간으로 인코딩하는 비디오 폰, 영상회의와 같은 분야에도 적용시킬 수 있다.

향후 연구 과제로는 PMV를 구할 때 이전 프레임의 움직임 벡터를 이용하여 더욱 최소점에 근접하는 PMV를 찾는 것과 PMV의 신뢰도 판단에 있어서 좀더 효과적인 기준을 연구하는 것이다. 속도측면에서 비디오 시퀀스의 종류에 관계없이 빠른 속도를 유지하도록 보완하는 것도 연구과제이다.

참 고 문 헌

[1] A.M. Tekalp, *Digital Video Processing*, Prentice Hall. 1995.

[2] ISO/IEC JTC1/SC29/WG11, *Generic coding of moving pictures and associated audio*, Part 2:Video, IS 11172-2, March 1995

[3] ISO/IEC JTC1/SC29/WG11, *Generic coding of moving pictures and associated audio*, Part 2:Video, IS 13818-2, March 1995

[4] ISO/IEC JTC1/SC29/WG11, *Generic coding of moving pictures and associated audio*, Part 2:Visual, IS 14496-2, Nov 1998

[5] ITU-T, "Video Coding for Low Bitrate Communication", Draft Recommendation H.263, July, 1995

[6] ITU-T, "Video Coding for Low Bitrate Communication", Draft Recommendation H.263+, Jan., 1998.

[7] ITU-T, "Video Coding for Low Bitrate Communication", Draft Recommendation H.26L, July, 2001.

[8] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, T. Ishiguro, "Motion-compensated interframe coding for video conferencing", Proc. NTC81, New Orleans, LA, pp. C9.6.1~9.6.5, Nov. 1981.

[9] J. R. Jain, A. K. Jain, "Displacement measurement and its application in interframe image coding", IEEE Trans. Commun., Vol. COM-29, pp. 1799~1808, Dec. 1981.

[10] A. Puri, H. M. Hang, and D. L. Schilling, "An efficient block matching algorithm for motion compensated coding", Intl. Conf. Acoust., Speech, and Signal Process, pp. 1063~1066, Dallas, TX, April. 1987.

[11] M. Ghanbari, "The cross-search algorithm for motion estimation", IEEE Trans. Commun., Vol. 38, No.7, pp. 950~953, July 1990.

[12] R.Li, B.Zeng, and M. L. Liou, "A New Three-step Search Algorithm for Block Motion Estimation" *IEEE. Trans. on Circuits Syst.*

Video Tech, Vol 4, pp. 438~442, Oct.1994

[13] L. M. Po and W. C. Ma, "A novel four-step search algorithm for fast block motion estimation", *IEEE. Trans. on Circuits Syst. Video Tech,* Vol 6, No. 3, pp. 313~317, Jun. 1996.

[14] J. B. Xu, L. M. Po, C. K. Cheung, "A new prediction model search algorithm for fast block motion estimation", Proc. ICIP, pp. 610~613, 1997.

[15] M. Gallant, G. Côté, and F. Kossentini, "An Efficient Computation-Constrained Block-Based Motion Estimation Algorithm for Low Bit Rate Video Coding," IEEE Tr. on Image Processing, Vol. 8 , No. 12, December 1999.

[16] G. Slavenburg et al., "TriMedia, TM1300 Preliminary Data Book," Philips Electronics North America Corporation, TriMedia Product Group, 811 E. Arques Avenue, Sunnyvale, CA, 1999.

[17] "Equator Hardware Reference MAP-CA DSP Datasheet", Equator Technologies, Inc., and Hitachi, Ltd., 2001.

[18] Zhou Wang and Alan C.Bovik, "A universal image quality index," IEEE Signal Processing Letters, vol. 9, no. 3, pp. 81~84, MARCH 2002.

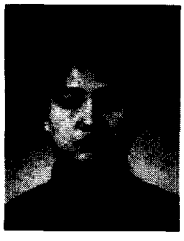
저 자 소 개

李 誠 鎬(正會員)

1998년 2월 : 광운대학교 전자공학과 졸업 (학사). 2002년 8월 : 광운대학교 대학원 전자공학과 졸업 (석사). 1998년 1월~2000년 7월 : LG산전. 2001년 9월~2002년 8월 : 인티스(주) 정보통신연구소 연구원. 2003년 2월~현재 : LG전자기술원. <주관심분야 : 영상압축, Motion Estimation>

魯 大 榮(正會員)

2001년 2월 : 광운대학교 전자공학부 졸업 (학사). 2001년 3월~현재 : 광운대학교 대학원 전자공학과 석사과정. 2001년 6월~현재 : 인티스 (주) 정보통신연구소 연구원. <주관심분야 : Motion Estimation, 영상 처리 시스템>

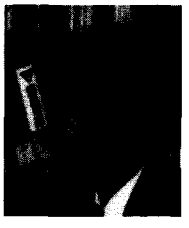


張 祐 演(正會員)

2001년 2월 : 광운대학교 전자공학부 졸업 (학사). 2003년 2월 : 광운대학교 대학원 전자공학과 졸업 (석사). 2001년 6월~2003년 2월 : 인티스(주) 정보통신연구소 연구원. 2003년 3월~현재 : 청남디지탈 (주) 연구원. <주관심분야 : Motion Estimation, 영상 압축>

吳 承 埈(正會員)

1980년 2월 : 서울대학교 전자공학과 졸업(학사). 1982년 2월 : 서울대학교 전자공학과 대학원 졸업(석사). 1986년 7월~1986년 8월 : NSF Supercomputer Center 초청 학생연구원. 1987년 5월~1988년 5월 : Northeast Parallel Architecture Center 학생연구원. 1988년 5월 : 미국 Syracuse University 전기 및 컴퓨터공학과 졸업(박사). 1982년 3월~1992년 8월 : 한국전자통신연구원 근무(멀티미디어연구실 실장). 1992년 9월~현재 : 광운대학교 전자공학부 및 정보통신연구원 교수 (멀티미디어연구실). 2000년 3월~현재 : (주)인티스 정보통신연구소 연구소장. <주관심분야 : 비디오처리, 비디오 및 영상압축, 멀티미디어시스템>



安 昌 範(正會員)

1981년 : 서울대학교 전자공학과 공학사. 1983년 : 한국과학기술원 전기 및 전자공학과 공학석사. 1986년 : 한국과학기술원 전기 및 전자공학과 공학박사. 1986년~1991년 : University of California, Irvine 연구조교수. 1991년~1992년: 생산기술연구원 부교수. 1992년~현재: 광운대학교 전기공학과 교수. <주관심분야 : 영상처리, 영상압축, 다차원신호처리, 의학영상시스템>