論文 97-22-3-7

움직임 벡터의 영역화에 의한 가변 블럭 동영상 부호화

正會員 김 진 태*, 최 종 수**

Moving Image Coding with Variable Size Block Based on the Segmentation of Motion Vectors

Jin Tae Kim*, Jong Soo Choi** *Regular Members*

※본 연구는 정보통신부의 대학기초연구지원사업의 연구비에 의해 연구되었음.

요 약

다양한 움직임을 포함하고 있는 동영상은 일정한 크기의 영역으로 나누어 부호화하는 것보다 움직이는 물체별로 영역을 구성하여 부호화하는 것이 효율적이다. 영상내에 존재하는 여러 움직임에 적합한 영역을 구성하여 움직임을 표현하므로, 복잡한 움직임에도 잘 적응할 수 있고 광범위한 형태의 움직임에도 안정적이다. 본 논문에서는 움직임 벡터의 영역화에 의한 새로운 동영상 부호화 기법을 제안한다. 우선, 움직임 벡터 필드에서 움직임 벡터를 스무딩하고 비슷한 움직임을 갖는 영역들로 분리하여 그 움직임들을 표현한다. 영역을 분리하기 위해서 영역 성장법을 사용하고, 영역의 병합여부는 해당 블럭의 움직임 벡터와 그에 따른 예측 오차를 이용한다. 이 때 영상이 갖는 상관성으로 인해 같은 영역에 포함될 수 있는 인접한 블럭들은 비슷한 움직임을 갖게된다. 실험 결과, 제안된 기법이 기존의 방법보다 부호화 측면에서 우수하였다.

ABSTRACT

For moving image coding, the variable size of region coding based on local motion is more efficient than fixed size of region coding. It can be applied well to complex motions and is more stable for wide motions because images are segmented according to local motions. In this paper, new image coding method using the segmentation of motion vectors is proposed. First, motion vector field is smoothed by filtering and segmented by smoothed motion vectors. The region growing method is used for decomposition of regions, and merging of regions is decided

*한서대학교 전산정보학과
**중앙대학교 전자공학과
論文番號:96389-1216
接受日字:1996年 12月 16日

by motion vectors and prediction errors of the region. Edge of regions is excluded because of the correlation of image, and neighbor motion vectors are used for evaluation of current block and construction of region. The results of computer simulation show the proposed method is superior than the existing methods in aspect of coding efficiency.

Ⅰ. 서 론

TV 신호와 같이 시간축상으로 변하는 영상 신호를 부호화하고자 할 때, 데이타 압축을 위해서는 시간 중복성을 제거하는 것이 필수적이다. 시간 중복성을 이용하는 영상 부호화 방법으로 움직임 보상 부호화(motion compensated coding:MCC)가 있다. 움직임 보상 부호화는 영상 신호에 존재하는 시간적, 공간적인 상관성을 이용하여 신호의 중복성을 효율적으로 줄임으로써 영상 신호를 전송 또는 저장하는데 필요한 대역폭을 감소시키는 방법이다. MCC는 높은 압축율에서도 우수한 재생 영상의 화질을 얻을 수 있다. 움직임 보상 부호화는 크게 움직임을 찾는 단위에 따라서 화소 순환법(pel recursive algorithm:PRA)과 블럭 정합법(block matching algorithm:BMA)으로 나뉠 수 있으나 계산량의 문제와 하드웨어 구현의 복잡성 때문에 대부분의 시스템에는 블럭 정합법이 쓰이고 있다.[1]

BMA는 입력 영상을 일정한 크기의 블럭으로 나누어 이전 프레임의 영상과 정합시켜 최소의 왜곡을 갖는 위치를 찾아내는 방법인데, 프레임내의 국부적 특성을 반영하는 데에는 한계가 있다. 연속 영상들에 존재하는 움직임은 복잡하게 구성되어 있으며 계속적으로 변화된다. 따라서 간단한 병진 운동만을 추정할 수 있는 블럭 정합 알고리듬에 의해서 고정된 블럭 크기 단위로 움직임을 추정하면, 영상의 국부적인 특성에 따라 발생하는 움직임을 효율적으로 추정할 수 없게 된다.[2, 3] 물론 움직임 추정시 블럭 크기를 작게 하면 국부적으로 변화하는 복잡한 움직임도 추정할 수는 있으나 추가된 움직임 정보를 전송해야 되는 문제가 발생한다. 따라서 움직임 추정의 예측 성능과 움직임 정보의 전송 부담을 모두 고려하여 움직이는 물체별로 영역화하고 영역에 의한 동영상 부호화 방법을 고려할 수 있다.[4, 5]

동영상의 영역 분할이란 시간축으로 추출되는 움직임 정보(벡터)를 이용하여 프레임의 각 영역이 의미 있는 물체를 포함하도록 분할하는 것이다. 고정 크기 블럭에 의한 영역 분할은 편의상 이용하는 것이므로 각각의 블럭들이 프레임에 내포된 물체의 형상과는 아무런 관계가 없다. 동영상을 움직이는 물체별로 부호화하고자 하는 객체 지향형 부호화 방법은 MPEG-4의 표준화와 더불어 현재 많은 연구가 진행되고 있어 기존의 고정 크기 블럭에 근거한 방법에 비해 많은 잠재력을 가졌다고 할 수 있다.[6] 기존의 연구 결과로는 임의의 영역에 대한 표면의 물레를 기호화하는 방법[7], 정사각형의 매크로블럭이 두 물체 이상의 경계면을 포함할 때 직교 변환 부호화 계수를 줄이면서 블럭마다 효과를 줄이려는 방법[8] 등이 있다.

본 논문에서는 동영상을 작은 블럭 크기로 움직임 추정을 하고 추정된 움직임 벡터를 움직임 벡터 필드에서 영역화한다. 영역화는 비슷한 움직임을 갖는 블럭들을 하나의 영역으로 병합하여 대표 움직임 벡터로 대치한다. 병합의 판단 여부는 스무딩된 움직임 벡터와 예측 오차를 이용한다. 영역화된 블럭은 움직이는 물체의 모양을 갖게 되어 영상의 국부적인 특성을 고려한 모양이 되고 전체적인 부호화 효율을 높일 수 있다. 최종적으로 전송되는 데이타는 가변 블럭의 형태와 영역별에 대한 움직임 벡터 그리고 예측 오차이다.

본 논문의 구성은 다음과 같다. Ⅰ장 서론에 이어, Ⅱ장에서는 움직임 벡터를 스무딩 시키는 방법에 대해 기술하며, Ⅲ장에서는 본 논문에서 사용한 영역 분할법을 소개한다. 움직임 벡터를 영역화하고 영역화된 움직임 벡터를 가변 블럭의 형태로 구성하는 방법을 Ⅳ장에서 제안하며, 제안한 방법의 컴퓨터 시뮬레이션과 그 결과를 Ⅴ장에서 검토하고, Ⅵ장에서 결론을 맺는다.

Ⅱ. 움직임 벡터 스무딩

실시간 구현의 용이성 때문에 현재 광범위하게 사용되고 있는 블럭 정합 알고리듬을 이용하여 구해진 움직임 벡터는 정해진 왜곡 평가 함수를 최소화시키는 방향으로 각 블럭별로 독립적으로 결정되므로, 물체의 실제 움직임을 충분히 표현할 수 없다. 이와 같이 얻어진 움직임 벡터는 회전(rotation) 운동이나 물체의 가까워짐 및 멀어짐(zooming) 운동 등을 제대로 반영하지 못하고, 하나의 블럭내에 2개 이상의 서로 다른 움직임을 갖는 물체가 포함되어 있는 경우에 블럭 정합 알고리듬으로 구해진 움직임 벡터는 신뢰성의 문제가 발생한다. 또한 잡음 등의 영향으로 물체의 실제 움직임과 관계없이 움직임 벡터가 불연속적으로 나타나기 때문에 인접한 블럭의 움직임 벡터간에도 상관성이 존재하지 않는다. 이와 같은 문제점을 해결하고 움직임 벡터의 부호화 이득을 얻고자하는 것이 움직임 벡터 스무딩 기법이다. 대표적인 움직임 벡터 스무딩 방법인 CCITT(현 ITU)의 RM7 방법을 소개한다.[9]

CCITT의 RM7에서는 배경 영역에서 불연속적이고 신뢰성 없는 움직임 벡터를 제거하기 위하여 움직임 추정후의 움직임 벡터를 스무딩한다. 현재 프레임에서 교정중인 블럭의 움직임 벡터를 MV, 현재 블럭과 이전 프레임의 같은 위치에 있는 블럭과의 차의 절대값의 평균을 FD(frame difference), 현재 블럭과 움직임 벡터에 의해 이동된 위치에 있는 이전 프레임의 블럭과의 차의 절대값의 평균을 DFD(displaced frame difference)라 할 경우, RM 7의 움직임 벡터 교정 알고리듬은 그림 1과 같다. 그림 1에서 움직임 벡터를 (0, 0)으로 한다는 것은 RM 7에서의 'No MC' 모드이며, 이것은 움직임 벡터를 부호화(전송)하지 않는다는 것을 의미한다.

```
if (FD < 1) then MV=(0,0)
else if (1 < FD < 3) then
    if (FD < DFD×2) then MV=(0,0)
else
    if (FD < DFD×1.1) then MV=(0,0)
```

그림 1. RM7의 움직임 벡터 스무딩 알고리듬
Fig. 1 Motion vector smoothing algorithm in RM7.

Ⅲ. 영역 분할

영상의 영역 분할은 영상을 유사한 성질을 갖는 화소들로 구성된 영역들로 분할하는 것을 말한다. 영역 분할은 영상 해석(image analysis)에서 영상을 이해하기 위한 전처리(preprocessing) 단계로 이용될 뿐만 아니라 영상 표현의 간략화, 즉 인간의 시각 특성을 고려한 영상 부호화 기법에서 중요한 역할을 하고 있다.[10] 영역 분할 기법은 분할과 병합(split and merge), 영역 성장(region growing)법[11], Weber의 법칙에 의한 JND(just noticeable difference) 시각 효과를 고려하여 문턱치를 변화시키는 방법[12] 등의 많은 종류가 개발되어 왔다.

벡터 필드의 영역 분할을 위해 영역 성장법을 이용한다. 본 논문에서는 움직임 벡터 필드에서 영역 분할을 수행하므로 영역 성장법의 전처리와 후처리 과정은 적용하지 않는다. 영역 성장법은 영상의 단위 변화에 충실하게 의존하여 영역을 분할하는 방법이다. 이 방법에서의 영역 분할은 전처리 필터링과 초기 분할(initial segmentation) 그리고 후처리(postprocessing) 과정으로 구성된다. 전처리 필터링은 원치 않는 소영역이 생기는 것을 방지하기 위해 사용된다. 초기 영역 분할은 다음과 같은 순서로 이루어진다.

단계 1 : 그림 2와 같이 주사 순서에 따라 중심 화소 I_0를 선택한다.

단계 2 : I_0에 인접한 I_1과 I_2중 I_0에 유사한 화소를 I로 하고 I가 속한 영역의 평균값을 $mean(I)$로 한다. 다음 식을 만족시키면 I_0를 I 영역으로 병합시키고, 아니면 새로운 영역을 구성하고 단계 3으로 간다.

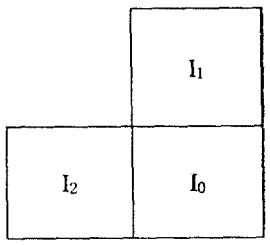


그림 2. 영역 분할을 위한 창
Fig. 2 Window for segmentation.

$$|I_0 - I| \times \beta + |mean(I) - I_0| \times (1-\beta) < \alpha$$

단계 3: I_1이 속한 영역과 I_2가 속한 영역이 다음 식을 만족시키면 두 영역을 병합시키고, 아니면 단계 1로 간다.

$$|mean(I_1) - mean(I_2)| < \alpha/2$$

위 식에서 α와 β는 영상에 의존한다. 영역 분할의 후처리 과정은 크기가 작은 국소 영역을 제거한다.

Ⅳ. 움직임 벡터의 영역화

블럭 정합 알고리듬에 의해서 구해진 움직임 벡터는 실제의 움직임(real motion)을 제대로 표현하지 못한다. 실제 배경이라 여겨지는 부분에서도 움직임 벡터가 (0, 0)으로 구해지지 않는 경우가 매우 많다. 이는 카메라의 잡음(noise) 등에서 생기는 문제인데, 이러한 잡음의 영향을 바로 잡아 주고자하는 방법이 움직임 벡터 스무딩이다. 이와 같이 블럭 정합 알고리듬에 의해 찾아진 움직임 벡터는 실제의 움직임을 표현하지 못하는 경우가 존재하므로 어떤 조건하에서는 실제의 움직임에 맞게끔 변화시키면 부호화 효율을 더욱 개선시킬 수 있게 된다. 따라서 본 논문에서는 움직임 벡터를 적당한 조건에 의해 교정시키고 이를 이용하여 영역을 분할하는 정보로 이용한다. 또한 영상이 기본적으로 갖고 있는 이웃하는 영역들간에는 상관성이 매우 크다는 성질을 이용한다. 두 물체의 경계와 배경과 움직이는 물체의 경계를 제외하면 이웃 블럭의 움직임 벡터도 비슷한 값을 가지는 것이 타당하다 할 수 있다.

영역 분할은 1차원 또는 2차원으로 수행시킬 수 있다. 한 축만을 고려하는 경우(1차원인 경우)에 영역 분할을 구성하는 순서는 다음과 같다. 입력 영상을 기본 크기 단위로 움직임 추정을 하여 움직임 벡터를 구해 움직임 벡터 필드를 구성한다. 처음 위치의 블럭과 다음 위치의 블럭에 대하여 두 개의 움직임 벡터와 예측 오차 값에 의해 두 블럭의 병합 여부를 판정한다. 처음 위치 블럭을 $Block_1$, 다음 위치 블럭을 $Block_2$, 처음 위치 블럭의 움직임 벡터를 MV_1, 예측 오차를 MAE_1, 다음 위치 블럭의 움직임 벡터를 MV_2, 예측 오차를 MAE_2로 정의하면, 각 블럭의 예측 오차를 아래 식으로 표현된다.

$$MAE_1 = Block_1(MV_1)$$
$$MAE_2 = Block_2(MV_2)$$

단계 0: 처음 위치 블럭에 (0, 0)의 움직임 벡터를 대입한 예측 오차를 계산한다. 구해진 예측 오차가 임계값 이하이면 움직임 벡터와 예측 오차를 갱신한다. 즉,

$MAE_1' = Block_1(0, 0)$
if $MAE_1' < T_1$ then $MV_1 = (0, 0)$, $MAE_1 = MAE_1'$
goto step 1

단계 1: 두 블럭의 움직임 벡터가 동일한지를 비교한다. 동일한 움직임 벡터를 갖는 블럭이면 두 블럭을 병합시켜 단계 6으로 간다. 그렇지 않으면 단계 2로 간다. 즉,

if $MV_1 = MV_2$ then *goto step* 6
else *goto step* 2

단계 2: 다음 위치의 블럭에 대해 처음 위치의 블럭에 의해 구해진 움직임 벡터를 적용시켜 그에 따른 예측 오차를 구한다. 이 때, 구해진 예측 오차가 임계값 이하이거나 원래의 예측 오차와 비교하여 그 차이가 작으면 두 블럭을 병합하고, 다음 블럭의 움직임 벡터는 처음 블럭의 움직임 벡터로 하고 예측 오차는 병합된 블럭에 맞게 단계 6에서 갱신된다. 그렇지 않으면 단계 3으로 간다. 즉,

$MV_2 = MV_1$, $MAE_2' = Block_2(MV_1)$
$D_2 = |MAE_2 - MAE_2'|$
if $MAE_2' < T_1$ or $D_2 < T_2$ then *goto step* 6
else *goto step* 3

단계 3: 단계 2에서와는 반대로 처음 위치의 블럭에 대해 다음 위치의 블럭의 움직임 벡터를 적용시켜 처음 블럭의 예측 오차를 계산한다. 구해진 예측 오차가 임계값 이하이거나 원래의 예측 오차와의 차이가 작으면 두 블럭은 병합되는 단계 6으로 간다. 그렇지 않은 경우에는

단계 4로 간다. 즉,

$MV_1 = MV_2$, $MAE_1' = Block_1(MV_2)$
$D_1 = |MAE_1 - MAE_1'|$
if $MAE_1' < T_1$ or $D_1 < T_2$ then *goto step* 6
else *goto step* 4

단계 4:두 블럭의 움직임 벡터의 평균을 구한다. 평균값에 해당하는 움직임 벡터를 각각 처음 블럭과 다음 블럭에 적용시키고 각각의 예측오차를 구한다. 두 블럭에서 평균값에 의한 움직임 벡터에 대해 구해진 예측 오차가 모두 임계값 이하이거나 새로 구한 예측 오차와 원래의 예측오차의 차이가 두 블럭 모두 임계값 이하이면 두 블럭의 움직임 벡터를 평균에 해당하는 움직임 벡터로 대치시키고 두 블럭을 병합시켜 단계 6으로 간다. 그렇지 않으면 단계 5로 간다. 즉,

$MV_1 = MV_2 = (MV_1 + MV_2)/2$
$MAE_1' = Block_1(MV_1)$, $MAE_2' = Block_2(MV_2)$
$D_1 = |MAE_1 - MAE_1'|$, $D_2 = |MAE_2 - MAE_2'|$
if ($MAE_1' < T_1$ and $MAE_2' < T_1$) or ($D_1 < T_2$ and $D_2 < T_2$)
then *goto step* 6 else *goto step* 5

단계 5:두 블럭은 병합할 수 없는 블럭으로 판단하고 두 블럭은 각각 원래의 움직임 벡터와 예측 오차를 유지하고 두 블럭을 독립된 형태로 처리된다. 그 다음에 오는 블럭과 위의 과정을 반복 처리한다. (*goto step* 0)

단계 6(두 블럭의 병합):두 블럭을 병합한다. 각각의 블럭이 갖는 움직임 벡터와 예측 오차를 병합된 블럭에 맞게 갱신한다. 병합된 블럭의 움직임 벡터를 MV_M, 예측 오차를 MAE_M라 하면

$$MV_M = (\sum^{k} MV_1 + \sum^{l} MV_2)/(k+l)$$

$$MAE_M = (\sum^{k} MAE_1(MV_1) + \sum^{l} MAE_2(MV_2))/(k+l)$$

여기서, k와 l은 각 블럭의 화소 수

마지막 블럭이 아니면 단계 0부터 다음 블럭을 처리한다. (*goto step* 0)

위와 같이 이웃하는 두 블럭에 대해 5가지 단계를 순서대로 검토한다. 위의 단계 1에서 단계 4까지 4가지 조건중에 어느 하나라도 만족되면 두 블럭은 병합되며 병합된 블럭은 만족되는 조건에 해당하는 움직임 벡터와 예측 오차를 갖게 된다. 4가지 조건이 모두 만족되지 않는 경우에 대해서는 처음 위치의 블럭과 다음 위치의 블럭을 독립된 움직임 성질을 갖는 블럭이라 보고 두 블럭을 별개의 영역으로 처리한다. 따라서 처음 블럭은 이미 구해진 움직임 벡터를 갖게되며, 다음 블럭은 그 다음에 오는 블럭과 위의 4가지 조건을 만족하는지의 여부 과정을 반복적으로 거치게 된다.

1차원으로 처리한 영역 분할의 효과를 더욱 확대시키기 위하여 영역 분할을 2차원으로 처리할 필요성이 있다. 위의 방법을 영역 분할 기법에 적용하여 2차원으로 확장하면 다음과 같다. 입력 영상은 기본 블럭 크기의 이전 프레임의 영상과 블럭 정합 알고리듬에 의해 움직임 벡터를 구한다. 구해진 움직임 벡터로 유사한 움직임을 갖는 블럭들을 병합시키는 영역 분할 단계를 거치는데 영역 분할은 다음과 같은 순서에 의해 이루어진다.

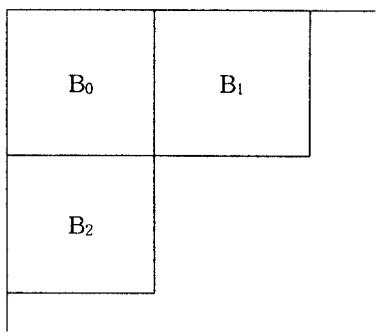


그림 3. 영역 분할에 이용한 블럭의 순서
Fig. 3 Block order for segmentation.

단계 0:주사 순서에 따라 그림 3과 같이 중심 블럭 B_0를 선택한다.

단계 1:블럭 B_0에 대해 조건에 맞으면 스무딩한다.

단계 2:중심 블럭 B_0에 인접한 B_1과 B_2 블럭의 움직임

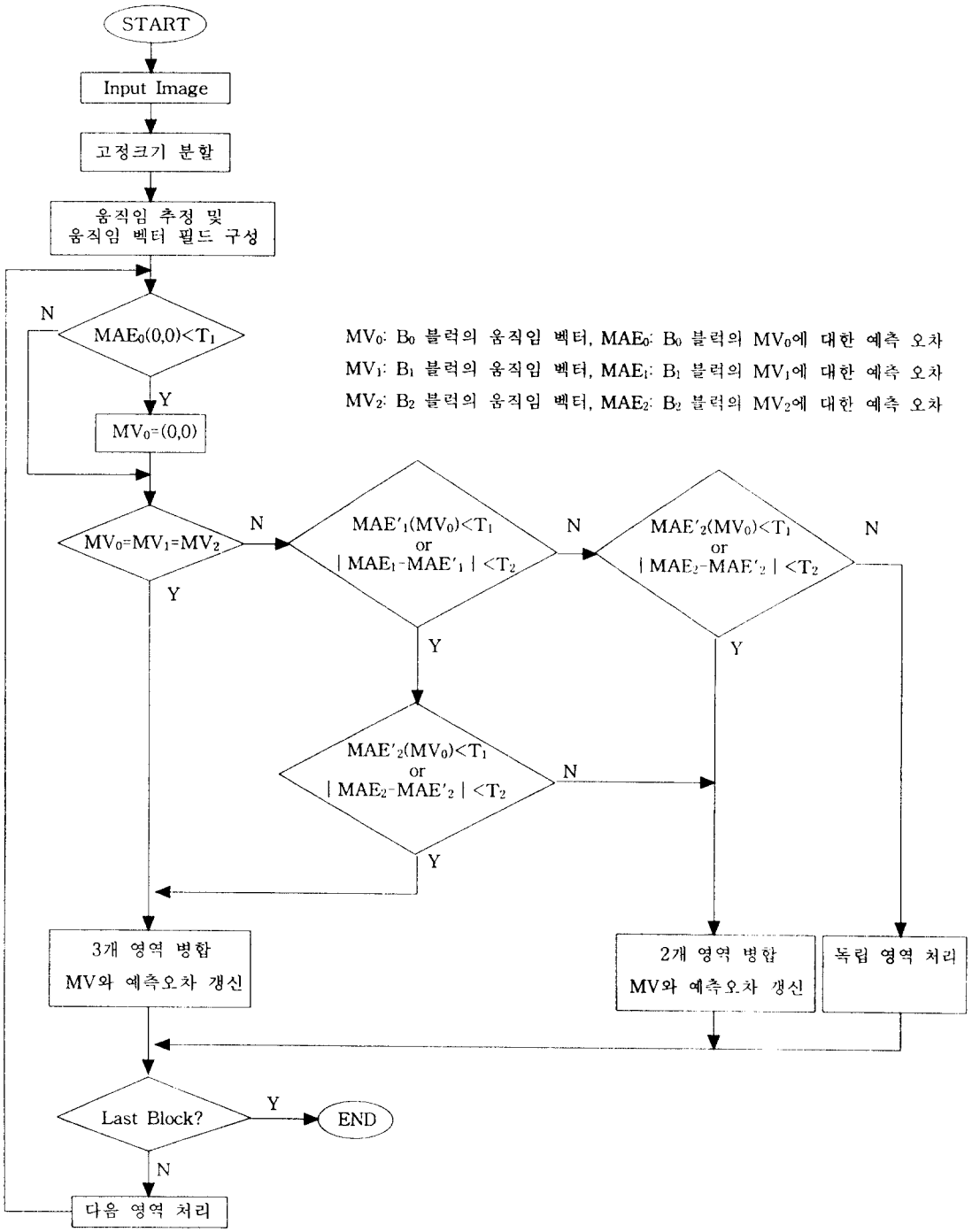


그림 4. 제안한 방법의 흐름도
Fig. 4 Flowchart of the proposed method.

벡터를 검색한다. 세 블럭이 모두 같은 움직임 벡터를 갖으면 하나의 블럭으로 병합시킨다. 단계 6으로 간다.

단계 3: 블럭 B_0의 움직임 벡터를 각각 블럭 B_1, B_2에 적용시켜 원래의 움직임 벡터에 의한 예측 오차와 비교한다. 두 블럭의 예측 오차의 차이가 임계값 이하이면 두 블럭을 블럭 B_0에 병합시키고 단계 6으로 간다.

단계 4: 블럭 B_1의 움직임 벡터를 블럭 B_0에 적용시킨다. 블럭 B_0의 예측 오차의 차이를 비교하여 병합 여부를 결정한다. 병합 조건이 맞으면 블럭 B_1에 병합시키고 단계 6으로 간다.

단계 5: 블럭 B_2의 움직임 벡터를 블럭 B_0에 적용시킨다. 예측 오차의 차이를 비교하여 병합 여부를 결정한다. 병합 조건이 맞으면 블럭 B_0를 블럭 B_2에 병합시키고 단계 6으로 간다.

단계 6: 병합한 블럭의 움직임 벡터와 예측 오차를 갱신시킨다.

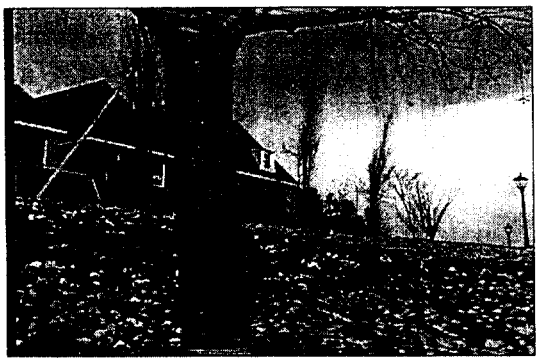
영역 분할에 의한 방법은 한 축으로만 병합하는 방법과 비슷하게 블럭별로 블럭 정합 알고리듬에 의해 구해진 움직임 벡터와 예측 오차를 영역 분할의 정보로 이용한다. 2차원으로 수행하는 경우에서 단계 3과 단계 4의 순서를 바꾸면 영역의 구성은 다른 모양을 갖게 되나 전체적인 부호화 효율 측면에서는 큰 차이가 없고, 이는 결국 병합하는 방향의 선택인데 영상 내의 움직이는 방향에 따라 결정하면 보다 비슷한 형태로 영역이 구성될 수 있다. 병합의 결과 영역은 임의의 모양을 지니게 되므로 이를 효율적으로 전송시켜야 된다. 그림 4는 제안한 방법의 흐름도이다.

V. 컴퓨터 시뮬레이션 및 검토

컴퓨터 시뮬레이션에는 MPEG(moving picture experts group) 지정 표준영상인 "Football", "Popple", 그리고 "Flower Garden" 등이 사용되었다. "Football"은 움직임이 전체적으로 상당히 크고 배경에 고주파 성분을 많이 포함한 영상이며, "Popple"은 매우 정적이고 극히 일부 영역에서만 규칙적인 회전 움직임이 있는 영상이며, 그리고 "Flower Garden"은 카메라의 패닝(panning) 현상으로 구성된 영상이다. 대상 영상 신호들은 모두 CCIR 601 형식인 4:2:2의 디지탈 신호로 휘도 신호 Y와 색차 신호 Cb, Cr로 구성되어 있다. 본 시뮬레이션은 휘도 신호(Y)만을 대상으로 한다. 세 영상신호 모두 비월 주사(interlaced scanning)된 신호이기 때문에 홀수 번째 필드만 세로축으로 보간(interpolation)하여 순차 주사(progressive scanning)된 형태로 바꾸어 프레임으로 처리하였다. 보간에 사용된 필터는 MPEG의 영상 해상도 변화에 사용되는 보간 필터[13]를 사용했으며 필터의 계수는 그림 5에 보인다. 실험에 사용된 영상의 크기는 720×480이고, 그림 6은 본 시뮬레이션에서 사용한 원영상들이다.

-12	0	140	256	140	0	-12	/256

그림 5. 보간에 사용한 필터
Fig. 5 Interpolation filter.



(a) "Flower Garden" 영상

(b) "Football" 영상

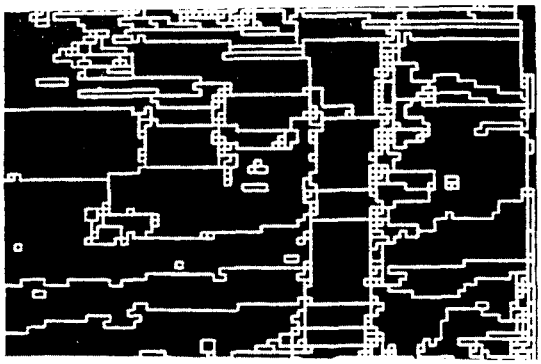
(c) "Popple" 영상

그림 6. 원 영상
Fig. 6 Original images.

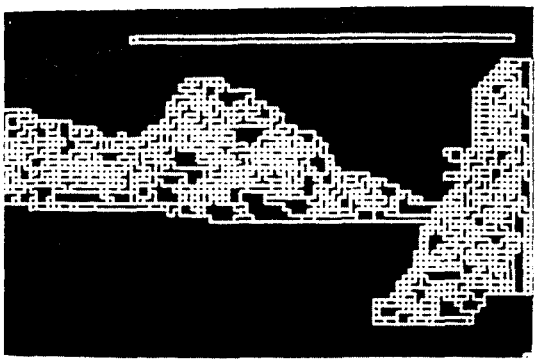
본 컴퓨터 시뮬레이션에서 첫 번째 프레임은 이미 복구되어 프레임 메모리에 있다고 가정하고 두 번째 프레임부터 15프레임을 처리하였다. 영역 분할을 구성하는 움직임 벡터 필드의 구성은 8×8 블럭 크기로 움직임 추정을 한후 형성하였다. 움직임 보상후의 예측 오차는 8×8 크기로 MPEG-2의 TM4형태로 DCT 부호화된다.[13] 움직임 추정은 전역 탐색법을 사용하고, 탐색 범위는 ±15pel/frame이다. 가변 크기 영역에 대한 부가 정보는 quadtree 형태로 구성하여 전송한다. 실험 결과를 다른 방법과 비교하기 위하여 8×8과 16×16의 고정 블럭 크기에 의한 방법과 참고문헌[3]의 가변 크기의 방법, 그리고 영역 분할에 의한 제안한 방법을 실험하였다. Kim 등의 방법[3]의 경우는 기준 블럭 크기를 8×8로 하고 세로축의 크기를 고정시킨 경우만 모의 실험을 하였다.

제안한 방법으로 영역이 분할된 형태를 그림 7에 보인다. 그림 7에서 보는 바와 같이 실험에 사용한 3가지 영상 모두에서 움직이는 물체별로 영역이 분할되고 있음을 알 수 있다.

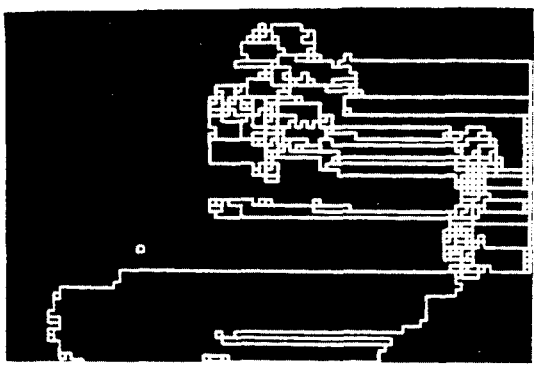
표 1~표 3에서 Fl_Gn은 "Flower Garden" 영상, Ftball은 "Football" 영상, Pop는 "Popple" 영상을 각각 의미한다. 표 1은 두 복구된 영상의 PSNR을 나타내고, 표 2는 각 프레임에서 구성된 영역의 개수를 표시하며, 표 3은 가변 블럭 크기에 대한 부가 정보를 나타낸다. Kim 등의 방법에서는 블럭의 크기에 대해 엔트로피를 구했고, 영역 분할에 의한 방법에서는 누



(a) "Flower Garden" 영상



(b) "Football" 영상



(c) "Popple" 영상

그림 7. 영역 분할된 영상의 형태
Fig. 7 Segmented images.

어진 영역의 quadtree 형태에 대해 엔트로피를 구한 다음 영역 수를 곱한 영상의 부가 정보를 구했다. 이때 사용된 식은 다음과 같다.

$PSNR = 10 \times \log_{10}(255^2/MSE)$ [dB]

$Entropy = -\sum p(i) \times \log_2 p(i)$ [bits]

여기서, MSE는 원영상과 복구된 영상의 평균 제곱 오차이고, p(i)는 영역의 크기에 대한 확률이다.

표 1, 2, 3의 고정 크기에 의한 방법의 비교 대상은 16×16에 의한 경우이다. 복구된 화질 측면에서, 가장 우수한 결과는 영역의 블럭 수가 가장 많은 8×8의 고정 크기의 경우이다. 표 1의 PSNR 측면에서 3가지 방법에서 비슷하게 세 영상 모두 1dB 이내로 비슷한 결과를 보였다. "Flower Garden"과 "Football" 영상은 16×16의 고정 크기에 의한 경우가 "Popple" 영상은 영역 분할에 의한 방법이 근소하나마 우수한 결과를 나타내고 있으나 시각적으로는 구분하지 못할 정도이고, Kim 등의 방법은 객관적인 화질 기준에서 두 방법의 중간 정도의 결과를 나타낸다. 영역 분할을 적용하는 방법 등과 같은 가변 블럭 크기의 경우 나누어지는 영역의 수를 증가시키면 화질도 같이 증가하지만 전체적인 화질을 고정 블럭 크기의 경우와 비슷하게 유지되도록 임계값을 조정하였다.

표 2에서 보면, 움직임 추정을 수행하는 블럭(영역)의 수는 고정 블럭 크기의 경우 8×8 크기이면 프레임당 5400개 16×16 크기이면 프레임당 1350개로 구성되고, 이에 대한 움직임 벡터의 전송이 필요하다. 영역 분할에 의한 방법의 경우에 블럭의 개수는 고정 블럭 크기의 경우에 비하여 모두 적게 구성되어 있다. 비슷한 화질 결과를 갖는 16×16의 고정 크기에 의한 경우와 영역 분할에 의한 경우를 비교하면, 영역 분할에 의한 경우가 약 400개에서 1100개 정도까지 적게 구성되어 있어 움직임 벡터를 전송할 때 상당한 비트를 절약할 수 있다.

표 3에서, 영역 분할에 의한 방법은 나누어진 영역 크기에 대한 부가 정보가 필요한데 본 시뮬레이션에서 나타난 부가정보는 프레임당 약 1400 비트(0.00405 bits/pel) 정도이다. 그러나 전체적인 영역의 수가 고정 크기의 경우보다 평균적으로 상당히 적기 때문에

표 1. 복구된 영상의 PSNR (dB)
Table 1 PSNR of the reconstructed images. (dB)

frame No.	고정 크기에 의한 방법						가변 크기에 의한 방법					
	8×8 크기			16×16 크기			Kim 등의 방법[3]			영역 분할의 방법		
	Fl_Gn	Ftball	Pop	Fl_Gn	Ftball	Pop	Fl_Gn	Ftball	Pop	Fl_Gn	Ftball	Pop
2	30.626	30.992	32.348	28.015	28.705	30.620	27.345	28.520	30.631	28.644	27.342	31.255
3	30.523	29.076	30.956	28.063	26.916	29.302	28.063	26.684	29.566	27.748	25.901	30.195
4	29.868	27.864	30.152	27.451	25.780	28.541	27.307	25.598	28.924	27.257	25.518	29.734
5	28.977	26.426	30.500	26.717	24.426	28.871	26.407	24.205	29.041	26.276	24.456	29.545
6	30.539	26.467	31.244	27.700	24.412	29.575	27.177	24.205	29.881	26.530	24.370	29.832
7	30.452	26.761	30.896	27.696	24.643	29.246	27.145	24.662	28.885	26.586	24.762	29.753
8	30.739	26.980	31.112	27.888	24.854	29.450	27.154	24.685	29.472	26.311	24.801	30.110
9	30.798	26.708	30.040	27.822	24.575	29.382	27.333	24.147	29.675	26.193	24.150	29.787
10	28.469	27.487	31.460	26.812	24.984	29.779	26.351	24.429	30.204	26.115	24.628	29.419
11	27.818	27.526	30.699	26.242	25.012	28.849	26.152	24.831	28.625	26.186	24.349	29.566
12	27.901	27.450	31.413	26.301	24.943	29.905	26.102	24.074	29.555	26.381	25.107	29.456
13	30.912	26.778	31.204	28.445	24.454	29.341	27.092	24.147	29.656	27.612	24.385	30.014
14	31.475	26.317	30.913	28.838	24.292	29.292	27.252	24.804	29.051	27.200	24.174	29.556
15	30.036	26.329	31.193	27.820	24.305	29.713	27.226	24.350	29.330	26.745	24.204	29.460
평균	29.938	27.369	31.009	27.558	25.164	29.419	27.008	24.953	29.464	26.842	24.868	29.834

표 2. 각 프레임에서 구성된 영역 수
Table 2 The number of regions in each frame.

frame No.	고정 크기에 의한 방법						가변 크기에 의한 방법					
	8×8 크기			16×16 크기			Kim 등의 방법[3]			영역 분할의 방법		
	Fl_Gn	Ftball	Pop	Fl_Gn	Ftball	Pop	Fl_Gn	Ftball	Pop	Fl_Gn	Ftball	Pop
2	5400	5400	5400	1350	1350	1350	1202	945	987	356	849	223
3	5400	5400	5400	1350	1350	1350	1225	1027	660	501	868	260
4	5400	5400	5400	1350	1350	1350	1169	1093	1067	534	891	267
5	5400	5400	5400	1350	1350	1350	1180	1233	1020	586	901	278
6	5400	5400	5400	1350	1350	1350	1229	1260	1058	603	928	259
7	5400	5400	5400	1350	1350	1350	1207	1235	655	691	937	246
8	5400	5400	5400	1350	1350	1350	1218	1264	626	590	982	252
9	5400	5400	5400	1350	1350	1350	1242	1302	1088	618	1025	245
10	5400	5400	5400	1350	1350	1350	1240	1404	1087	555	1158	271
11	5400	5400	5400	1350	1350	1350	1240	1244	1081	690	867	260
12	5400	5400	5400	1350	1350	1350	1130	1184	956	516	1092	273
13	5400	5400	5400	1350	1350	1350	1200	1114	987	514	1124	276
14	5400	5400	5400	1350	1350	1350	1176	1269	908	587	914	233
15	5400	5400	5400	1350	1350	1350	1248	1233	911	628	960	275
평균	5400	5400	5400	1350	1350	1350	1208	1201	935	569	964	258

표 3. 각 프레임의 부가 정보 (bits/frame)
Table 3 Overhead information in each frame.

frame No.	고정 크기에 의한 방법						가변 크기에 의한 방법					
	8×8 크기			16×16 크기			Kim 등의 방법[3]			영역 분할의 방법		
	Fl_Gn	Ftball	Pop	Fl_Gn	Ftball	Pop	Fl_Gn	Ftball	Pop	Fl_Gn	Ftball	Pop
2	0	0	0	0	0	0	3523	2563	2296	1420	1180	1020
3	0	0	0	0	0	0	3527	2742	2016	1636	1296	1204
4	0	0	0	0	0	0	3527	2927	2291	1568	1340	1132
5	0	0	0	0	0	0	3536	3026	2399	1532	1488	1312
6	0	0	0	0	0	0	3680	2972	2383	1512	1556	1308
7	0	0	0	0	0	0	3638	3036	2116	1628	1388	1228
8	0	0	0	0	0	0	3686	3069	2033	1592	1564	1212
9	0	0	0	0	0	0	3702	3164	2393	1552	1516	1196
10	0	0	0	0	0	0	3642	3260	2283	1600	1636	1272
11	0	0	0	0	0	0	3699	2929	2308	1635	1313	1250
12	0	0	0	0	0	0	3675	2917	2176	1543	1472	1129
13	0	0	0	0	0	0	3579	3014	2326	1595	1585	1269
14	0	0	0	0	0	0	3683	3059	2169	1596	1526	1173
15	0	0	0	0	0	0	3683	3039	2195	1554	1356	1254
평균	0	0	0	0	0	0	3627	2980	2242	1569	1444	1211

움직임 벡터 전송시 비트율을 절감할 수 있다. 따라서 영상의 국부적인 움직임 등을 고려한 영역 분할에 의한 동영상 부호화 방법이 기존의 고정 블럭 크기에 의한 부호화 방법보다 화질 향상을 기대할 수도 있고 전송량을 절감할 수도 있다.

가변 크기 블럭으로 수행하는 Kim 등의 방법은 표 1에서 표 3을 통한 실험 분석 결과, 고정 블럭 크기(16×16)와 영역 분할에 의한 경우의 중간 정도의 성능을 나타내고 있어 기존의 가변 블럭 크기에 의한 방법보다 제안한 방법의 성능이 보다 우수함을 알 수 있다. 이는 Kim 등의 방법이 1차원적(가로축 혹은 세로축 고정)으로 가변 블럭의 형태를 만들기 때문에 영상내에 존재하는 2차원적 상관성에 대한 고려가 미흡했기 때문이라 생각된다.

그림 8은 "Popple" 영상의 각 프레임에서 발생한 전송량을 나타낸다. 전송량은 예측 오차를 DCT하고 양자화한후 구한 비트량과 움직임 벡터, 부가 정보에 대한 총 비트량을 화소 단위로 나타내었다. 그림 8에서 fix_8×8은 8×8의 고정 크기 경우이고, fix_16×16은 16×16의 고정 크기 경우이고, Kim et. al.은 참고문헌[3]에 의한 방법이며, segment는 영역 분할에 의한 방법을 나타낸다. 그림에서 보는바와 같이 영역 분할을 하여 부호화하면 기존의 고정 크기에 의한 방법보다 영역(블럭)의 수가 적음으로 인해 전송해야할 움직임 벡터의 수가 절감되고, 영역의 벡터가 스무딩으로 인해 벡터의 상관도가 높아져서 전체 전송 비트량을 절감할 수 있다.

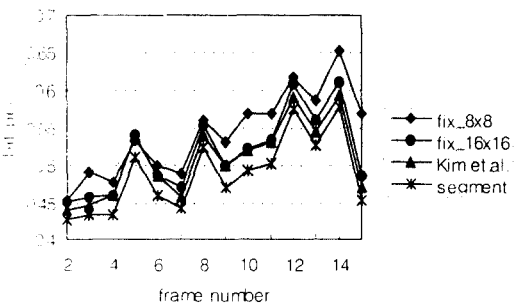


그림 8. "Popple" 영상에서의 발생 비트량
Fig. 8 Transmitted bits for "Popple" image.

Ⅵ. 결 론

동영상 부호화에서 프레임을 고정 크기 블럭으로 나누어 부호화하면 영상의 국부적인 특성을 고려하는 데는 한계가 있다. 배경 영역에서는 영역의 크기를 크게 할 필요성이 있으며, 움직임이 작고 빠른 영역에서는 영역의 크기를 작게 해야 부호화 효율을 높일 수 있다.

제안한 기법은 움직임 벡터 필드에서 영역을 분할하여 가변 블럭을 구성했다. 가변 블럭의 형태는 영상의 국부 특성을 반영할 수 있어 부호화 측면에서 효율적이다. 가변 블럭의 구성은 움직임 벡터 스무딩 기법을 영역 분할 기법에 적용시켜 비슷한 움직임을 갖는 영역으로 분할하였고 분할된 영역은 대표 움직임 벡터로 표현된다. 가변 크기의 영역 수는 일반적인 고정 크기의 경우보다 상당히 적은 수로 구성되며, 전체적인 화질은 비슷한 정도이다. 가변 크기의 영역에 대한 부가 정보도 크지 않으므로 전체적인 부호화 효율을 향상시킬 수 있었다.

참 고 문 헌

1. H. S. Musmann, P. Pirsch, and H. J. Grallert, "Advanced in picture coding," *Proc. IEEE*, vol. 73, no. 4, pp. 523-548, 1985.
2. A. Puri, H. M. Hang, and D. L. Schilling, "Interframe coding with variable block size motion compensation," *Proc. GLOBECOM'87*, pp. 2.7.1-2.7.5, 1987.
3. 김진태, 최종수, 박래홍, "영상의 국부적 성질을 이용한 가변 크기 블럭 정합 알고리듬," 대한전자공학회논문지, 제29-B권, 제7호, pp. 62-69, 1992.
4. T. B. Yu, M. H. Chan, and A. G. Constantinides, "Low bit rate video coding using variable block size model," Proc. ICASSP'90, pp. 2229-2232, 1990.
5. G. J. Sullivan and R. L. Baker, "Rate-distortion optimized motion compensation for video compression using fixed or variable size blocks," *Proc. GLOBECOM'91*, pp. 85-90, 1991.
6. E. H. Adelson and J. Y. Wang, "Representing moving images with layer," Technical Report Media

Lab, Vision and Modeling Group, TR No. 228, 1993.

7. Sharp Corporation, "Temporal scalability algorithm based on image content," ISO/IEC JTC1/SC29/WG11 MPEG95/0377, 1995.
8. M. Banham and J. Brailean, "Motorola MPEG-4 video submission technical description," ISO/IEC JTC1/SC29/WG11 MPEG95/0324, 1995.
9. CCITT SG X V, *Description of reference model 7*, 1988.
10. M. Kunt, A. Ikonomopoulos, and M. Kocher, "Second-generation image coding technique," *Proc. IEEE*, vol. 73, no. 4, pp. 549-574, 1985.
11. R. M. Haralick and L. G. Shapiro, "Survey: Image segmentation techniques," *Computer Vision, Graphics, and Image Processing*, vol. 29, pp. 100-132, 1985.
12. S. A. Rajala, M. R. Civanlr, and W. M. Lee, "A second generation image coding technique using human visual system based segmentation," *Proc. ICASSP'87*, pp. 1362-1365, 1987.
13. Test Model Editing Committee, "Coded representation of picture and audio information," ISO-IEC/JTC1/SC29/WG11 MPEG93/225, 1993.

김 진 태(Jin Tae Kim) 정회원

1995년 3월~현재 : 한서대학교 전산정보학과 전임강사

한국통신학회 논문지 제21권 제5호 참조

※주관심분야 : 영상부호화, MPEG, 영상통신 등임

최 종 수(Jong Soo Choi) 정회원

1981년 9월~현재 : 중앙대학교 전자공학과 교수

한국통신학회 논문지 제21권 제2호 참조

※주관심분야 : 컴퓨터시각, 영상부호화, 적외선신호처리 등임