1995年度 定期總會 및 秋季學術大會 論文集 1995.11.18

영역별 특성을 이용한 적응적 움직임 벡터 추정 기법

박 태희°, 이 동욱, 김 재민*, 김 영태
동국대학교 전기공학과, 삼성전자*

Adaptive Motion Vector Estimation Using the Regional Feature

Tae-Hee Park °, Dong-Wook Lee, Jae-Min Kim*, Young-Tae Kim
Dept. of Electrical Eng. Dongguk Univ., Samsung Elec. Co.

ABSTRACT

In video image compression, it is important to extract the exact motion information from image sequence in order to perform the data compression, the field rate conversion, and the motion compensated interpolation effectively.

It is well known that the location of the smallest sum of absolute difference(SAD) does not always give the true motion vector(MV) since the MV obtained via full block search is often corrupted by noise. In this paper, we first classifies the input blocks into 3 categories : the background, the shade-motion, and the edge-motion. According to the characteristics of the classified blocks, multiple locations of relatively small SAD are searched with an adaptive search window by using the proposed method. The proposed method picks MVs among those candidates by using temporal correlation. Since temporal correlation reveals the noise level in a particular region of the video image sequence, we are able to reduce the search are very effectively.

1. 서 론

영상 부호화는 영상 신호들의 공간적·시간적 상관성을 이용하여 신호의 중복성을 줄임으로써 데이터 전송 또는 축적하는데 필요한 신호의 대역을 압축하기 위한 신호 처리 기법을 말한다. 일반적으로 연속적인 화상 신호는 현재 프레임과 이전 프레임과의 상관성이 매우 크기 때문에 이러한 상관 관계를 이용하여 필요한 정보는 그대로 유지하면서 정보를 표현하는 데이터량을 줄이는 동영상 압축 방법에 관한 연구 및 표준화가 최근 꾸준히 진행 중이다[9].

MPEG-2의 영상 압축 기법의 기본적인 부호화 원리는 움직임 추정을 이용한 시간적 중복성 제거와 변환 부호화를 이용한 공간적 중복성 제거 및 엔트로피 부호화를 이용한 통계적 중복성 제거 방식으로 나눌 수 있는데, 이들 방식중 정보 압축 효율 면에서 가장 영향을 많이 미치는 것은 시간적 중복성 제거 방식으로, 프레임과 프레임 사이의 상관 관계를 구하여 이들 상관성을 제거해서 압축 효과를 얻는 방법이다[9][10].

움직임 보상 방식은 연속되는 영상 신호에서 현재 프레임의 화소들이 이전 프레임에 비해 어느 정도 움직였는지를 움직임 벡터(motion vector : MV)로 추정하여, 전체 영상의 전송 대신 이들 움직임 벡터 정보를 전송함으로써 전송량을 줄이는 기법이다[2][3][4]. 여기에는 화소 순환식(pel-recursive)[6][7] 방법과 블록 정합(block matching)[1][3][5] 방법이 있는데, 알고리즘의 간략함과 그에 따른 적은 계산량 때문에 성능은 조금 뒤질지라도 블록 정합을 이용한 움직임 벡터 추정 기법이 더 폭넓게 이용되고 있다. 하지만 이 방법은 암시적으로 같은 블록내의 화소들은 같은 방향의 움직임을 갖는다는 가정을 가지고 출발하기 때문에 영상의 움직임이 복잡한 경우, 영상의 지역적인 변화에 따른 움직임을 효과적으로 반영할 수 없다. 또한 블록 정합의 full search, 2-D log search, three step search 와 같은 알고리즘[1] 또한 1초당 30프레임이라는 방대한 데이터 처리시 계산량의 부담 및 다양한 영상에 대한 획일적인 적용으로 영상내 움직임 상태에 따라 무시 못할 에러가 발생하여 화질을 심각히 악화시키는 주요한 요인이 된다.

본 연구는 위와 같은 문제점을 고려하여 영상내 움직임 상태에 따라 영역을 구분하여 각 영역에 가변 블록 크기를 이용한 적응적 움직임 추정 방식을 제안한다.

2. 영역별 특성에 따른 가변 블록 크기 설정

먼저 분류기를 이용하여 배경 영역(background)과 그림자 영역(shade region), 그리고 에지 영역(edge region)으로 나눈다. 배경 영역은 움직임이 없는 영역으로 간주하여 알고리즘 적용 없이 움직임 벡터 값을 (0,0)으로 정하고, 그림자 영역은 움직임이 작은 영역으로 보고 계산상의 복잡성을 고려하여 부샘플링(subsampling)을 이용한 알고리즘을 적용한다. 에지 영역에 대해서는 움직임이 많은 영역으로 보고 계산 시간 단축보다는 정확한 움직임 벡터를 찾는데 중점을 둔 전체 탐색(full search)을 이용한 알고리즘을 적용한다[8][11].

2.1 배경 영역 분류기

다음 식을 이용해서 영상을 배경 영역과 움직임 영역으로 구분한다.

$$bd = \sum_{i=0}^{m}\sum_{j=0}^{m} (x(i,j) - r(i,j)), \quad 0 \le i,j \le m. \tag{2.1}$$

m : 블록 크기, $x(i,j)$: 현 프레임 데이터
$r(i,j)$: 기준 프레임 데이터

bd(background distortion)이 미리 정해진 기준값보다 작으면 해당 블록을 배경 영역으로 분류하고 그렇지 않으면, 움직임 영역으로 분류한다.

2.2 그림자 영역과 에지 영역 분류기

움직임 영역을 그림자 영역과 에지 영역으로 분류하기 위해 먼저 다음과 같이 정의하자.

$$d(i,j) = x(i,j) - \eta, \quad 0 \le i,j \le m \tag{2.2}$$

m : 블록 크기, $x(i,j)$: 현 프레임 데이터,
(i,j) : 화소 위치 η : 블록 평균

움직임 영역 중에 $d(i,j)$가 D의 원소라 할 때, D의 모든 원소가 다음 식을 만족하면 그림자 영역(shade region)영역으로, 그렇지 않으면 에지 영역(edge Region)으로 구분한다.

$$|d(i,j)| < \tau \quad i,j = 0,\cdots,m. \tag{2.3}$$

τ : 가변 기준값

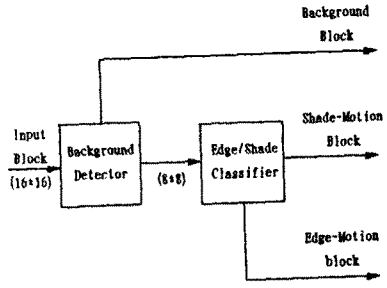


그림 1 영역 분류기 블록 선도
Fig. 1 The block diagram of the classifier

3. 적응적 움직임 추정 알고리즘

3.1 알고리즘의 필요성

블록 정합 기법에서 이용되는 SAD(sum of absolute difference)는 다음과 같이 정의된다.

$$SAD=\frac{1}{|m|^2}\sum_{i=0}^{m}\sum_{j=0}^{m}|f_2(i-\alpha,j-\beta)-f_1(i,j)| \quad (3.1)$$

여기서 m은 블록의 크기(block size), $f_k(i,j)$는 프레임 k 내 위치 (i,j)에서의 화소의 밝기(intensity), (α,β)는 움직임 변위이다.

일반적인 블록 정합 기법에 기초한 움직임 벡터 추정 알고리즘에서는 최소 SAD값을 갖는 블록의 위치로부터 움직임 벡터를 추출하게 되는데 항상 최소 SAD값의 위치가 최선은 아니다. 비록 연속적인 두 프레임이 매우 짧은 시간 간격으로 진행된다고 할지라도 어떤 위치에 있는 화소의 그레이 레벨 값은 조금씩 변하고, 잡음의 영향으로 두 번째, 세 번째로 작은 SAD값을 갖는 블록의 위치가 실제 움직임 벡터 검출에 이용될 수 있는 좋은 정보가 되기도 한다. 만일 두 번째로 작은 SAD값을 갖는 후보 블록이 실제 움직임 벡터 정보라면 기존의 블록 정합 알고리즘에 따른 최소 SAD값에 대한 움직임 정보는 잘못 찾아낸 것이 되고, 잘못된 추정 결과로 블록화 현상(block effect)이 발생한다. 이런 결과로 위 가정은 타당하다고 볼 수 있으며 움직임 벡터 필드 상에서 시간상의 상관관계를 추출하여 움직임 벡터 추정 방법으로 이용한다[12].

3.2 적응적 움직임 벡터 추정 알고리즘

전체 탐색으로부터 얻은 최소 SAD값의 위치가 실제 움직임 벡터에 대해 잡음의 영향을 받은 것으로 본다. 이때 잡음은 연속적인 영상 프레임에서 화소 블록의 가속이나 감속으로 생긴 것이거나 카메라 상에서 작동 오류(poor camera handling, misfocus, camera zooming), 갑작스런 조명 밝기의 변화 등으로 볼 수 있다. 이러한 잡음의 영향 속에서 실제 움직임 벡터를 추정하는 것이 우리의 목적으로, 최소 SAD값을 통해 움직임 벡터를 찾기 보다 실제 움직임 벡터를 추정하는 이유는 실제 움직임 벡터가 여러 프레임에 걸쳐 강한 상관관계를 갖기 때문이다.

각 블록의 왼쪽 윗 부분 화소를 블록 포인터(block point : BP)라하여 그 블록을 표시하는 기준 좌표로 이용한다. 즉 $BP(i,j,k)$는 k번째 프레임에서 위치가 (i,j)인 블록의 위치를 나타낸다. 간단한 예로 한 프레임 시간(1/30 sec) 동안 $BP(i,j,k)$가 $(\triangle x_1,\triangle y_1)$만큼 움직이고 $BP(i,j,k+1)$가 $(\triangle x_2,\triangle y_2)$만큼 움직였다고 했을 경우에, 잡음에 대한 영향이 전혀 없는 경우에는 다음과 같은 결과를 보여야만 한다.

$$\delta=\|(\triangle x_1,\triangle y_1)-(\triangle x_2,\triangle y_2)\|=0. \quad (3.2)$$

$(\triangle x_1,\triangle y_1)$: 움직임 변위

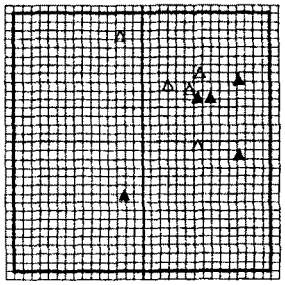
K개의 프레임이 주어졌을 경우에, $k=2,3,\cdots,K$에 대해 $BP(i,j,k)$의 움직임 벡터 후보의 위치를 $(\triangle x_{1,k},\triangle y_{1,k})$, $(\triangle x_{2,k},\triangle y_{2,k})$, $\cdots$, $(\triangle x_{C,k},\triangle y_{C,k})$ 으로 나타내고, 이때 $BP(i,j,k)$와 $BP(i,j,k+1)$사이의 움직임 벡터의 프레임 상호간 차이(interframe differential)를 아래 식과 같이 쓸 수 있다.

$$\delta_{k,k+1,C_a,C_b}=\|(\triangle x_{C_a,k},\triangle y_{C_a,k})-(\triangle x_{C_b,k+1},\triangle y_{C_b,k+1})\| \quad 1\le C_a,C_b\le C \quad (3.3)$$

여기서 C는 $BP(i,j,k)$에서 찾은 움직임 후보 벡터의 개수이고, C_a와 C_b는 각 프레임에서 C개의 후보 중에서 a번째와 b번째 후보를 나타낸다.

연속적인 두 프레임에서 물체의 움직임이 거의 일정하고, 잡음에 대한 영향이 없을 경우에는 식 (3.3)에서 $\delta_{k,k+1,C_a,C_b}$도 식 (3.2)에서와 마찬가지로 '0'의 값을 가져야 한다. 그러나 실제 비디오 시퀀스에서는 잡음을 포함하고 있으며, 일정한 속도를 갖는다는 가정 또한 성립하지 않으므로 $\delta_{k,k+1,C_a,C_b}$의 값은 더 이상 '0'일 수 없다. 하지만 서로 다른 K개 프레임 사이에서 임의의 한 블록의 움직임 벡터 후보들을 일정 평면 위에 표시하면 어떤 작은 지역에 모여 있는 것을 볼 수 있다. 이때 그림 2에서 처럼 움직임 벡터의 프레임 상호간 차이(interframe differential)를 계산해서 가장 작은 값을 갖는 움직임 벡터 군(set)을 실제 움직임 벡터 정보를 갖는 것으로 인정한다.

가능하면 많은 프레임을 이용하여 움직임 벡터에 대해 연속적으로 추정하면 할수록 더욱더 정확한 추정 결과가 나오겠지만 데이터 저장 및 계산상의 복잡성 때문에 이를 고려하여, 3개 프레임만으로 움직임 벡터 추정을 수행하였다.



△ : f_{k-1}와 f_k 사이 후보 벡터들의 상대적인 위치

▲ : f_k와 f_{k+1} 사이 후보 벡터들의 상대적인 위치

그림 2 움직임 벡터 후보들의 위치도

Fig. 2 Location of motion vector candidates

다음은 적응적으로 움직임 벡터를 찾는 과정을 설명한다.

1) 먼저 k_1, k_2 두 프레임을 읽어서 k_2 프레임의 각각의 블록에 대해 k_1 프레임에서 C번째까지의 작은 SAD 값을 갖는 움직임 벡터 후보를 찾는다. 이때 초기 탐색 영역은 $W_{max}\times W_{max}$ 최대로 한다.($W_{max}=31$, $C=3$)
2) 다음 프레임 k_3를 읽어서 k_3 프레임의 각각의 블록에 대해 k_2 프레임에서 C번째까지의 작은 SAD 값을 갖는 움직임 벡터 후보를 찾는다. 이때 초기 탐색 영역은 $W_{max}\times W_{max}$ 최대로 한다.($W_{max}=31$, $C=3$)
3) 단계 1), 2)에서 찾은 C개의 후보 벡터들 사이의 거리를 구한다. (absolute distance 이용)
4) 단계 3.에서 가장 짧은 거리(D_{min})를 갖는 벡터 쌍을 선택한다. k_1, k_2 프레임에서 찾은 것은 두 프레임 사이의 실제 움직임 벡터로 확정 짓고, k_2, k_3에서 찾은 것은 k_4 프레임 처리시 탐색 영역의 위치로 이용한다. 이때 D_{min} 값에 따라 새로운 탐색 영역 범위를 설정한다.
5) 다음 프레임 k_4를 읽는다.
   단계 3-5를 계속해서 반복 실행한다.

이론상 단계 4)를 통해 프레임이 연속적으로 진행됨에 따라 탐색 영역의 크기는 그림 3처럼 점점 줄게 된다.

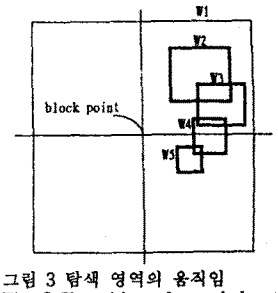


그림 3 탐색 영역의 움직임

Fig. 3 Transition of search location

4. 모의 실험 결과

움직임 정보에 따른 가변 블록 크기 적용 및 적응적 움직임 추정 알고리즘의 성능 평가를 위해 352×240 크기의 table tennis, foreman, claire 영상 55프레임에 대해 다음 4가지 알고리즘의 컴퓨터 시뮬레이션을 행하였다.

- 전체 탐색(F.S.)
- 가변 블록 크기를 이용한 전체 탐색(VBS)
- 적응적 움직임 추정(AD)
  가변 블록 크기를 이용한 적응적 움직임 추정(VBS_AD)

영역별 특성을 분류할 때의 블록 크기는 16×16이지만 그림자 영역과 에지 영역에서 적응적 움직임 추정 알고리즘을 적용할 때는 다시 블록 크기를 8×8로 나누어 움직임 벡터를 추정하였다. 그림 8, 9는 각각 table tennis와 foreman에 대한 움직임 벡터도이다. 그림 8을 보면 탁구공 부분과 오른쪽 구석의 왼쪽 팔 부분이 움직임이 큰 에지 영역이고, racket 과 오른팔 부분이 움직임 적은 그림자 영역이다. 이들 영역에 대한 움직임 벡터도를 보면 움직임 정도를 볼 수 있다. 이 벡터도에서 table의 경계 부분을 보면 실제로 움직임 없는 부분이지만 프레임 진행 중 미세한 gray 레벨 값의 변화를 순수한 전체 탐색에서는 최소 SAD 값을 이용해 찾아 생긴 불필요한 움직임 정보다. 하지만 가변 블록 크기를 이용한 적응적 움직임 추정(VBS_AD)에서는 그런 미세한 화소 값 변화를 가변 블록 크기(VBS)에서

미리 배경 영역으로 분류하였기 때문에 table의 에지 부분에서는 전혀 움직임 벡터를 찾아 볼 수 없다. 그림 10, 11은 순수한 전체 탐색에서의 탐색 횟수를 '1'로 보았을 경우에 대한 위 4가지 방법을 이용하여 탐색 횟수를 상대적으로 나타낸 그림이다. 여기서 VBS는 움직임 벡터를 찾는 처리 속도가 전체 탐색에 비해 거의 일정하게 빠름을 알 수 있고, 이들 두 알고리즘에 비해 AD와 VBS_AD는 첫 번째 움직임 벡터 탐색 후 다음 프레임으로 진행됨에 따라 탐색 횟수가 줄어들을 볼 수 있다. 이것은 MV 추정시 잡음의 영향에 따라 탐색 영역을 설정하고 현 프레임과 다음 프레임에서의 움직임 좌표 값을 탐색 시작점으로 이용하기 때문이다.

5. 결론

본 논문은 영상 신호의 움직임 정도에 따라 영역을 분류하고 각각의 영역에 대해 수행 시간과 정확도를 고려하여 빠르고 정확하게 움직임 벡터를 추정하는데 목적이 있다. 알고리즘은 크게 영역 분할과 적응적 움직임 벡터 추정으로 나눌 수 있는데, 먼저 영역별 특성 분류기를 이용하여 입력 신호를 움직임이 거의 없는 배경 영역과 작은 움직임을 갖는 그림자 영역, 그리고 움직임이 복잡하던가 아니면 움직임 정도가 큰 움직임 영역으로 나눈다. 이들 영역 중 배경 영역은 추정 알고리즘 적용 없이 움직임 벡터를 (0,0)으로 정하고, 그림자 영역은 부샘플링을 이용하여 적응적 움직임 추정 알고리즘을 통해 움직임 벡터를 결정한다. 그리고 움직임 영역은 정확한 움직임 추정을 위해 전체 탐색을 이용한 적응적 움직임 추정 알고리즘을 적용하여 움직임 벡터를 찾아낸다.

이 제안된 알고리즘은 다음과 같은 장점을 갖고 있다. 처리 속도는 전체 탐색보다 월등하게 빠르면서 움직임 추정 결과는 유사하거나 뛰어나다는 점과, MPEG-2에서 매크로 블록(16×16) 층 대신 직접 DCT 블록 크기인 8×8을 이용하여 움직임 벡터를 추정함으로 인해 비트 수를 줄일 수 있으며, 프레임 처리 순서도 MPEG-2에서의 비순차적인 것과는 달리 순차적으로, 정보 처리와 전송이 가능하다. 추정 알고리즘 적용시 프레임이 진행됨에 따라 처리 속도가 빨라짐을 볼 수 있었으나, 만일 영상 신호가 갑자기 전환되는 상황이 발생한다면 그 순간에는 움직임 추정 결과 및 처리 속도가 현저하게 떨어질 것으로 보인다. 그러나 MPEG-2에서 처럼 11프레임마다 초기화를 시켜 준다면 장면 전환시 문제점 및 잘못 추정된 정보로 인해 발생할 수 있는 누적 에러를 11프레임마다 다시 초기화되기 때문에 큰 문제는 되지 않는다. 또한 기존의 블록 정합 방법의 복잡인 알고리즘의 반복성을 그대로 유지하고 있기 때문에 하드웨어 구현을 간단하게 할 수 있으며 프레임 간의 상관성을 적절하게 이용할 수 있으면 비트율 조절에도 상당한 이득이 있을 것으로 보인다.

참고 문헌

[1] H. G. Musmann, P. Pirsch, and H. J. Grallert, "Advances in picture coding," Proc. IEEE, vol. 73, pp. 523-548, Apr. 1895

[2] R. Srinvasan and K. R. Rao, "Predictive Coding Based on Efficient Motion Estimation," IEEE Trans. Commun., vol. COM-33, no. 8, pp. 888-896, Aug. 1985

[3] Y. Ninomiya and Y. Ohtsuka, "A Motion Compensated Interframe Coding Scheme for Television Pictures," IEEE Trans. Commun., vol. COM-30, no. 1, pp. 201-211, Jan. 1982.

[4] R. Srinibasan and K. R. Rao, "Motion compensated coder for videoconferencing," IEE Trans. Commun., vol. COM-35, pp. 297-304, Mar. 1987

[5] J. R. Jain and A. K. Jain, "Displacement and its application in interframe image coding," IEEE Trans. on Commun., vol. COM-29, no. 12, DEC. 1981

[6] D. R. Walker and K. R. Rao, Senior member, "Improved Pel-Recursive Motion Compensation," IEEE Trans. on Commun., vol. COM-32, no. 10, Oct. 1984

[7] A. N. Netravali and J. D. Robbins, "Motion compensated television coding : Part I," Bell Syst. Tech. J., vol. 58, pp. 631-670, Mar. 1979

[8] M. H. Chan, Y. B. Yu, and A. G. Constautinides, "Variable size block matching motion compensation with application to video coding," Proc. IEE, Pt. I, vol. 137, pp. 205-212, Aug. 1990

[9] MPEG, "Coding of Moving Pictures and Associated Audio," tech. rep., ISO JTC1/SC2/Wg1190/176, 1990.

[10] Grand Alliance HDTV System Specification version 2.0 Dec. 1994

[11] S. Y Huang, J. R. Chen, J. S. Warg "Classified Variable Block Size Motion Estimation Algorithm for Image Sequence Coding," Proc. ICIP, Nov., 13-16, 1994

[12] S. G. Kim, C. J. Kuo, "Adaptive Motion Estmation in Video Coding with a Stochastic Model," Proc. ICIP, Nov., 13-16, 1994

그림 4. table tennis 1

그림 5. table tennis 2

그림 6. foreman 1

그림 7. foreman 2

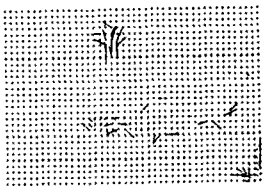


그림 8. tennis MV

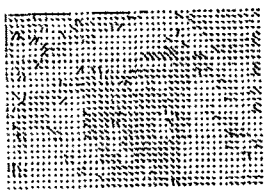
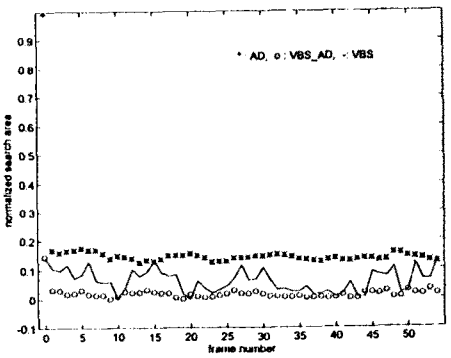
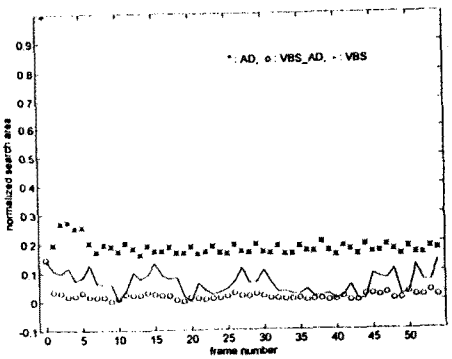


그림 9. foreman MV



- foreman -
그림 10. search area



- claire -
그림 11. search area