논문 00-25-1B-20 한국통신학회논문지 '00-1 Vol.25 No.1B

탐색 영역에서의 움직임 특성을 이용한 고속 블록 움직임 추정

정회원 최 정 현*, 박 대 규*, 정 태 연*, 이 경 환*, 이 법 기*, 김 덕 규*

Fast Block Motion Estimation Using the Characteristics of the Motion in Search Region

Jung-Hyun Choi*, Dae-Gyue Park*, Tae-Yeon Jung*, Kyeong-Hwan Lee*,

Bub-Ki Lee*, Duk-Gyoo Kim* *Regular Members*

요 약

간단하고 점진적인 움직임 추정 알고리듬인 3단계 탐색 방법은 저비트율 영상 압축분야에서 널리 이용되어 왔다. 본 논문에서는 탐색 영역 내에서의 움직임 특성을 이용한 새로운 고속 블록 움직임 추정 알고리듬을 제안한다. 대부분의 움직임 벡터는 탐색 영역의 중심 부분에 존재하며, 따라서 본 논문에서는 중심 부분의 움직임을 3단계 탐색 방법보다 훨씬 세밀하게 추정한다. 또한 각 단계에서 가능한 모든 움직임 방향을 고려하면서 이전 단계내의 탐색영역과 중첩되지 않는 국부 영역에 대해서만 움직임 추정을 행한다. 따라서 제안한 방법은 기존의 방법과 비교하여 보다 우수한 움직임 추정을 얻을 수 있으며, 계산 시간을 줄일 수 있다.

ABSTRACT

The three-step search(TSS) algorithm, a simple and gradual motion estimation algorithm, has been widely used in some low bit-rate video compression. We propose a new fast block motion estimation algorithm using the characteristics of motion in search region. Most of motion vectors exist in the center region of search area, so the motion in that region is examined more closely than TSS in this paper. Also in a search step, motion vector is estimated in the local area which is not overlapped with the search area in previous step, considering the all possible direction of motion. Therefore, we get the better motion estimation and reduce computational time in compared with the conventional methods.

Ⅰ. 서 론

화상 전화, 화상 회의 및 HDTV 등에서는, 영상 및 음성 정보를 제한된 용량의 채널을 통하여 전송하거나 저장 매체에 저장하기 위해서 데이터를 압축한다. 동영상 부호화에서는, 많은 양의 영상 데이터를 압축하기 위해서 동영상에 존재하는 여러 가지 중복성들, 즉 프레임내 (intraframe)의 화소들 사이에 존재하는 공간적 중복성, 인접 프레임간 (interframe)에 존재하는 시간적 중복성, 그리고 발생 부호의 확률적 중복성 등을 제거한다. 동영상 데이터에서는, 시간 상관 계수가 공간 상관 계수보다 훨씬 크기 때문에 인접 프레임간의 시간적 중복을 제거하는 움직임 보상 부호화가 효율적이다. 대부분의 움직임 보상 부호화는, 이전 프레임에서 현재 블록의 움직임을 찾는 움직임 추정 단계와 움직임 보상된 영상과 원 영상과의 차 영상의 부호화 단계로

* 경북대학교 전자전기공학부(jhchoi@palgong.kyungpook.ac.kr)
논문번호 : 99221-0603, 접수일자 : 1999년 6월 3일

이루어진다. 움직임 추정 방법으로는 비교적 하드웨어 구현이 간단하고 성능이 우수한 블록 정합 알고리듬 (block matching algorithm; BMA)이 사용된다. 전역 탐색 블록 정합 알고리듬 (full search BMA; FSBMA)은 탐색 영역 내의 모든 탐색점들에 대해 블록 정합을 행하는 방법으로서, 움직임 벡터를 정확하게 찾을 수는 있지만 많은 계산량을 필요로 한다. 움직임 추정의 계산량을 줄이기 위해서, 2차원 로그 탐색 (two dimensional logarithmic search), 교차 탐색 (cross search), 3 단계 탐색 (three step search; TSS) 방법 등의 여러 고속 블록 정합 알고리듬들이 소개되었다[1]-[5]. TSS 방법에서는, 움직임 추정 오차는 움직임이 일어나는 방향으로 단조 감소한다고 가정하고, 전체 탐색 영역에 대해서 몇개의 참조 탐색점 (reference search point)을 정하여 단계를 진행할수록 점진적으로 움직임을 추정한다[3]. 이 방법에서는 이전 단계에서 찾은 최소 오차의 참조 탐색점을 중심으로, 현재 단계에서는 참조 탐색점의 간격을 좁혀서 탐색을 행함으로써, 많은 계산량 감소를 얻을 수 있었다. Lu 등이 제안한 움직임 추정 방법에서는 단계를 진행할 때, 4 방향의 움직임만을 고려하여 이전 단계에서 탐색하지 않은 영역에 대해서만 탐색을 행함으로써 탐색점 수를 크게 줄일 수 있었다[6]. 그러나 움직임이 비교적 작은 영상에 대해서는 TSS 방법과 비슷한 성능을 나타내지만, 움직임이 크고 움직임 벡터의 분포가 균일하지 않은 영상에 대해서는 성능이 저하되는 단점이 있다.

본 논문에서는, 실제 영상의 움직임 벡터가 탐색 영역의 중앙에 편중된다는 특성[5]과 최적 탐색점 (global minimum search point)에 대한 움직임 추정 오차의 특성[3]을 이용하여 고속 움직임 추정 방법을 제안하였다. 이 방법에서는 탐색 초기 단계에서 참조 탐색점의 간격을 작게 설정하여, 탐색 영역의 중앙 부분에 대해서 세밀하게 탐색을 행함으로써, 이들 영역에서 기존의 방법들에 비해서 더 정확하게 움직임을 추정할 수 있었다. 또한 이전 단계와 현재 단계에서의 최소 오차 탐색점의 블록 MAD (mean absolute difference)의 크기를 비교하여 세부 탐색 방향을 결정함으로써 탐색점 수를 줄일 수 있었다. 네 방향의 움직임에 대해서만 탐색을 행하는 Lu 등의 방법에 비해 제안한 방법에서는 가능한 모든 움직임 방향을 고려하므로, 움직임이 크고 움직임 벡터의 분포가 균일하지 않는 영상에 대해서도 우수한 결과를 얻을 수 있었다. 본 논문에서는 Ⅱ장에서 기존의 방법들에 대하여 설명하고, Ⅲ장에서는 제안한 알고리듬을 설명하며, Ⅳ장에서는 컴퓨터 모의 실험 결과를 보이고, Ⅴ장에서 결론을 맺는다.

Ⅱ. 기존의 움직임 추정방법

1. TSS 방법[3]

TSS 방법은 움직임 추정 오차가 움직임 방향으로 단조 감소한다는 가정을 이용하여, 전역 탐색을 행하지 않고 탐색 영역의 일부에 대해서만 탐색을 행함으로써 계산량을 크게 줄일 수 있는 기법이다. 단계를 진행하면서 참조 탐색점의 간격을 점점 줄여 점진적으로 움직임 벡터를 추정한다. 이 방법의 순서는 다음과 같이 요약 될 수 있다.

1 단계: 참조 탐색점의 간격을 4로 하여, 원점과 그 주위 8개의 참조 탐색점에 대하여 블록 정합을 행하여 최소 오차의 탐색점(SP_1)을 찾는다.

2 단계: 참조 탐색점의 간격을 2로 하여, SP_1 주위의 8개의 참조 탐색점에 대하여 블록 정합을 행하여 최소 오차의 탐색점(SP_2)을 찾는다.

3 단계; 참조 탐색점의 간격을 1로 하여, SP_2 주위의 8개의 참조 탐색점에 대하여 블록 정합을 행하여 최소 오차의 탐색점(SP_3)을 찾아서 이를 최종 움직임 벡터로 정한다.

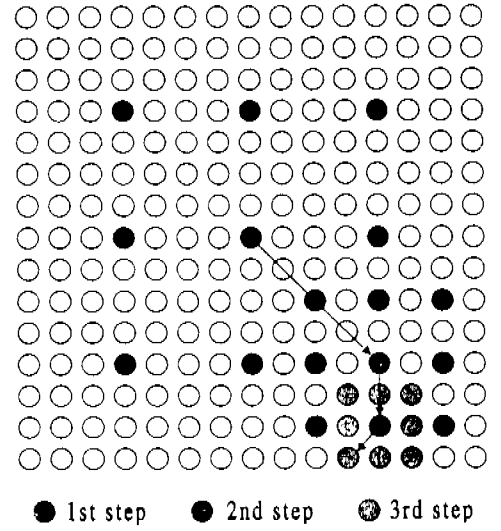


그림 1. TSS 방법의 예

If B_MAD(A)≥B_MAD(B) and B_MAD(A)≥B_MAD(C), Ⅰ is selected
If B_MAD(A)≥B_MAD(B) and B_MAD(A) 〈 B_MAD(C), Ⅱ is selected (1)
If B_MAD(A) 〈 B_MAD(B) and B_MAD(A) 〈 B_MAD(C), Ⅲ is selected
If B_MAD(A) 〈 B_MAD(B) and B_MAD(A)≥B_MAD(C), Ⅳ is selected

최종 움직임 벡터가 (3, 7)일 경우에 대해서 TSS 방법의 예를 그림 1에 나타내었다. 그림 1의 예에서와 같이 탐색 영역의 크기가 -7~+7인 경우에 25개의 탐색점에 대하여 블록 정합을 행하기 때문에, 탐색 영역 내의 모든 225개의 탐색점에 대해서 탐색하는 전역 탐색 방법에 비해 11.11% 정도(25/225)의 적은 계산량을 가진다. 그러나, 탐색 영역의 전체가 아닌 일부에 대해서만 탐색을 행하므로 국부 최소에 빠질 수 있는 문제점이 있다. 특히 영상의 움직임 추정 오차가 이 방법에서 사용된 가정과 달리 움직임 방향으로 단조 감소하지 않는 경우 움직임 추정 오차가 매우 커지는 단점이 있다.

2. Lu 등의 방법[6]

TSS 방법에서는 움직임 추정 오차가 움직임 방향으로 단조 감소한다는 가정을 이용하여 탐색점 수를 줄였다. 그러나 각 단계에서 8 방향 모두에 대하여 탐색을 행하는 것은 가정에 위배되므로 비효율적이다. 즉, 최소 오차 (minimum error)은 동시에 서로 반대 방향으로 일어나지 않는다는 가정을 전제하였으므로, TSS 방법에서의 각 단계의 8 방향의 탐색점들 중 일부는 탐색 영역이 중복된다. 그러므로, Lu 등의 방법에서는 모든 움직임 방향에 대하여 탐색을 행하는 TSS 방법과 달리, 이전 단계에서 구한 최소 오차의 탐색점을 기준으로 수평과 수직 방향의 탐색점에 대해서만 움직임 추정 오차를 구하고 그 크기를 비교하여 움직임의 진행 방향을 결정하므로 TSS 방법에 비해 계산량을 많이 줄일 수 있다.

Lu 등의 방법에 대하여 설명하면 다음과 같다. (움직임 추정 오차는 블록 MAD로 하였다.)

1 단계; 참조 탐색점의 간격을 4로 하여, 그림 2에서와 같이 원점과 수평과 수직의 참조 탐색점에 대하여 블록 MAD를 구하고, 식 (1)에서와 같이 그 크기를 비교하여 탐색 방향을 선택한다. 탐색 방향이 결정되면, 그림 3에서와 같이 탐색을 행하여 최소 오차 탐색점(SP_1)을 구한다.

2 단계; 참조 탐색점의 간격만 2로 줄이고서, SP_1을 원점으로 두고 1 단계와 같이 반복하여 최소 오차 탐색점(SP_2)을 구한다.

3 단계; 참조 탐색점의 간격만 1로 줄이고서, SP_2를 원점으로 두고 1 단계와 같이 반복하여 최소 오차 탐색점(SP_3)을 구하고 이를 최종 움직임 벡터로 정한다.

최종 움직임 벡터가 (3, 7)일 경우에 대해서 Lu 등의 방법의 예를 그림 4에 나타내었다.

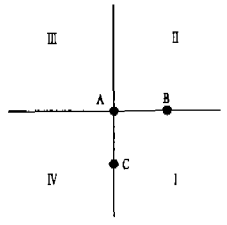


그림 2. Lu의 방법에서 각 단계에서 탐색 방향을 정하기 위한 첫 번째 탐색 패턴.

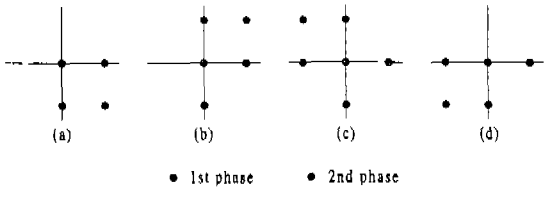


그림 3. Lu의 방법에서 각각의 선택된 탐색 방향에 대한 탐색 패턴:
(a)1사분면; (b)2사분면; (c)3사분면; (d)4사분면.

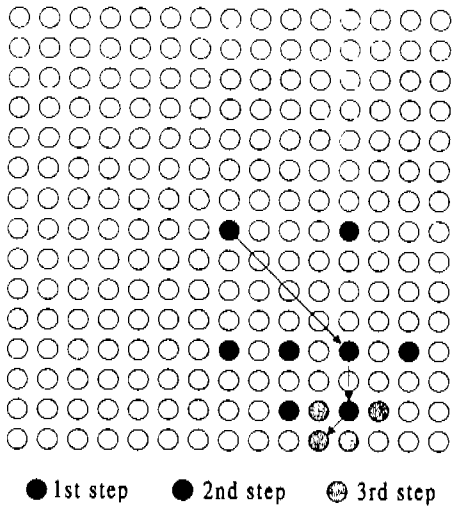


그림 4. Lu외 방법의 예

Ⅲ. 제안한 고속 움직임 추정 방법

실제 동영상에서 움직임 벡터는 탐색 영역내의 중앙 부분에서 많이 발생되며, 움직임이 큰 가장 자리 부분으로 갈수록 발생 빈도는 줄어든다.[5] 또한, 움직임 추정 오차는 움직임 방향으로 단조 감소한다[3].

제안한 방법에서는 움직임 벡터의 분포 특성을 고려하여, 초기 단계의 참조 탐색점의 간격을 작게 설정하여 세밀한 탐색을 행함으로써, 정확하게 움직임을 추정할 수 있다. 또, 최적 탐색점에 대한 각 탐색점에서의 추정 오차의 특성을 고려하여, 현재 단계에서의 세부 탐색 방향은 이미 진행된 단계에서의 최소 추정 오차와 현재 단계에서의 최소 추정 오차의 크기를 서로 비교한 후 결정함으로써 탐색점 수를 줄이면서 더 정확한 추정을 행하였다. 본 논문에서는 블록의 움직임 추정 오차를 블록 MAD로 하였다.

현재 단계에서의 탐색은 그림 5에서와 같이, 원점을 기준으로 이전 단계에서 최소 오차를 가지는 탐색점의 방향으로 행한다. 그림 6에서와 같이, 만약 더 이상의 단계를 진행할 필요가 없을 경우, 즉 i 번째 단계에서 최소 오차 탐색점(SP_i)의 블록 MAD(B_MAD(SP_i))가 i-1 번째 단계에서 최소 오차 탐색점(SPi-1)의 블록 MAD(B_MAD(SP_{i-1}))보다 클 경우에는 B_MAD(SP_i), B_MAD(SP_{i-1})과 B_MAD (SP_{i-2})를 비교해서 세부 탐색점을 결정한다.

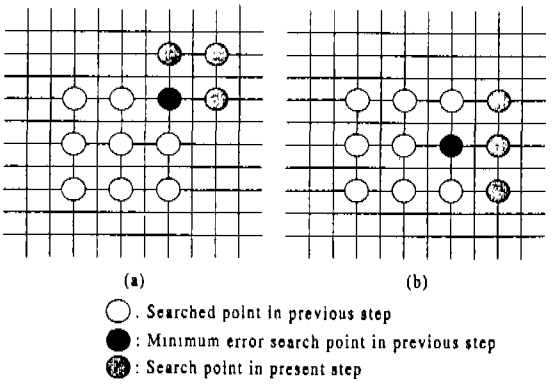
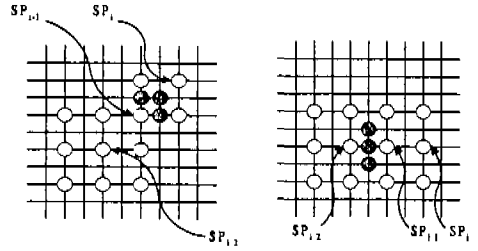


그림 5. 각 단계에서의 최소 오차 탐색점의 위치에 따른 두 가지 탐색 패턴



SP_i : minimum error search point in i-th step
(a)B_MAD(SP_{i-1})<B_MAD(SP_i)<B_MAD(SP_{i-2})
(b)B_MAD(SP_{i-1})<B_MAD(SP_{i-2})<B_MAD(SP_i)

그림 6. 더 이상 단계 진행이 없을 경우 두가지 세부 탐색 패턴:

제안한 고속 움직임 추정 방법을 단계별로 설명하면 다음과 같다.

1단계: i=1로 두고 참조 탐색점의 간격을 2로 하여 탐색영역의 원점으로부터 구성되는 정방형의 9점에 대해 정합을 행한 후, 최소 블록 MAD를 가지는 점 (SP_i)과 두 번째 최소 블록 MAD를 가지는 점(SP_{i-1})을 찾는다. 이때 원점이 SP_i일 경우 더 이상의 단계 진행 없이 이를 중심으로 탐색 변위를 1로 좁히고, SP_i와 SP_{i-1}사이의 위치관계에 따라 그림 6에서와 같이 인접 3개의 탐색점에 대해서만 세부 탐색하여 최종 움직임 벡터를 구한다. 원점이 SP_i가 아닐 경우에는 i←(i+1)로 하고, 2단계로 넘어간다.

2단계: 이전 단계에서 최소 블록 MAD를 가진 탐색점(SP_{i-1})을 기준으로 탐색 변위를 2로 유지하면서 그림 5에서와 같이 3개의 탐색점에 대해서만 탐색을 행하여 SP_i를 찾는다. 만약

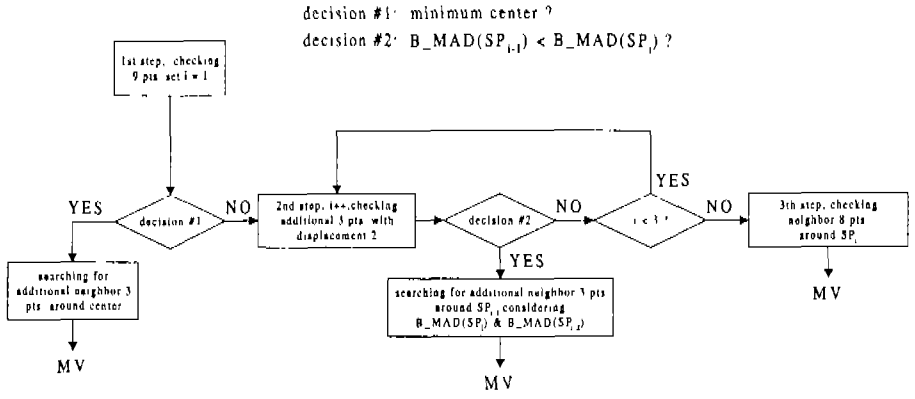


그림 8. 제안한 방법의 흐름도

B_MAD(SP$_i$) 〉 B_MAD(SP$_{i-1}$)일 때는 더 이상의 단계 진행없이 탐색 변위를 1로 좁히고 B_MAD(SP$_i$)와 B_MAD(SP$_{i-2}$)의 크기를 비교하여 작은 방향의 인접 3개의 탐색점에 대해서만 정합하고 끝낸다.(그림 6)

B_MAD(SP$_i$) 〈 B_MAD(SP$_{i-1}$)일 때는 움직임의 진행이 계속될 확률이 존재하므로, i ←(i+1)로 두고 다시 2단계를 수행한다. 단, 탐색영역내에서 더 이상 진행 할 단계가 없을 경우 3 단계로 넘어간다.

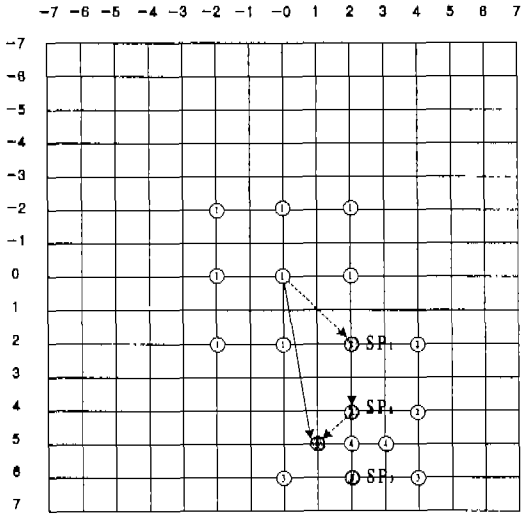
3단계: 이전 단계까지의 최소 블록 MAD를 가지는 탐색점을 중심으로, 탐색변위를 1로 두고서 인접 8개의 탐색점에 대해 정합을 행하여 최종 움직임 벡터를 결정한다.

그림 7에 움직임 벡터가 (1, 5)인 경우 제안한 방법의 움직임 추정 예를 보였다. 제안한 고속 움직임 추정 방법의 흐름도를 그림 8에 나타내었다.

저비트율 비디오 응용에서 탐색은 15×15크기의 탐색 영역에서 이루어지며 따라서 수평, 또는 수직 방향으로 나타날 수 있는 움직임 벡터의 최대 변위는 7이 된다. 이 경우 전역탐색방법은 225개의 탐색점수에 대해 정합을 행해야 하며, TSS 방법은 25개의 탐색점수에 대해 정합을 행해야 한다. 그러나 제안한 방법은 각 단계마다 중도 정지하는 경우가 많이 발생하므로 한 블록의 움직임을 추정하기 위한 탐색점수가 정확히 정해진 것은 아니지만 평균 14점 정도를 정합한다. 정확한 정합 회수를 수식으로 표현하면,

$$L = (9+3)P_1+(9+3+3)P_2+(9+3+3+3)P_3 + (9+3+3+8)P_4 \quad (2)$$

과 같다. 여기서, P_i는 i 단계에서 최종움직임 벡터가 결정될 확률이며, 서로 배반 사건이다. 이 확률들은 영상의 움직임 특성의 분포에 따라 달라진다. 즉, 움직임이 큰 영상에 대해서는 정합해야 할



(B_MAD(SP$_2$) < B_MAD(SP$_3$) < B_MAD(SP$_1$))

그림 7. 제안한 방법의 움직임 추정 예.

탐색점수가 많아지고, 반대로 움직임이 작은 영상에 대해서는 탐색해야 할 탐색점수가 적어진다.

제안한 방법에서는 움직임 벡터의 분포 특성, 즉 움직임 벡터가 탐색영역의 중앙에 집중되어 분포한다는 특성을 고려하여, 초기 단계에서의 참조 탐색점의 간격을 기존의 방법보다 작게 하여 세밀한 움직임 추정을 행하기 때문에, 움직임이 작은 영역에

서의 추정을 보다 정확하게 하여 성능을 개선 할 수 있다. 또한, 탐색점 사이의 블록 MAD는 탐색 영역 내의 최적 정합점에서 멀어질수록 증가한다는 특성을 고려하여, 각 단계에서 최종 움직임 결정할 때 인접 8점 중 불필요한 다섯 점을 제외함으로써 탐색점수를 줄여 보다 빠른 추정을 가능하게 하였다.

Ⅳ. 실험 결과 및 고찰

제안한 고속 움직임 추정 방법의 성능을 평가하기 위해서, 전역 탐색 방법, TSS 방법, 그리고 Lu의 방법과 비교하여 컴퓨터 모의 실험을 수행하였다. 실험 영상으로 352×240크기의 SIF 영상을 100 프레임씩 사용하였다. 실험 영상으로는 화상 전화용의 실험 영상인 SALESMAN, 전체적인 카메라 이동이 있는 비교적 움직임이 작고 분포가 일정한 FLOWER GARDEN, 사람의 신체와 공의 움직임이 급격하게 변하고 갑작스런 카메라 줌이 일어나는 비교적 움직임이 크고 그 분포가 불규칙한 TABLE TENNIS 그리고 비교적 큰 움직임이 많은 FOOTBALL 영상을 각각 100프레임씩 사용하였다. 탐색을 위한 블록의 크기는 8×8을 그리고 탐색영역은 -7에서 7까지로 하였다. 정합척도는 MAD를 사용하였다. 객관적인 성능을 비교하기 위해 화질의 척도로는 PSNR을 사용하였고, 계산량의 척도로는 움직임 추정의 대부분을 차지하는 블록 정합 회수를 사용하였다.

제안한 고속 움직임 추정 방법과 기존 방법의 성능 비교를 위해 최초 100프레임을 수행한 후 평균 PSNR을 표 1에 나타내었다. 표 1에서와 같이 FLOWER GARDEN과 TABLE TENNIS 영상 모두에 대해서 제안한 방법이 기존의 방법들보다 우수한 성능을 보인다. 영상의 움직임이 작고 움직임 벡터의 분포가 일정한 FLOWER GARDEN 영상의 경우, 제안한 방법에서는 탐색 영역의 중앙 부분에 대하여 세밀한 탐색을 행하므로, TSS 방법과 Lu의 방법보다 더 우수한 결과를 나타내었다. TABLE TENNIS 및 FOOTBALL 영상과 같이 영상의 움직임이 크고 불규칙할 경우, Lu의 방법에서는 탐색 진행 방향을 4 방향만 고려하기 때문에 TSS 방법보다 성능이 많이 저하되지만, 제안한방법에서는 모든 가능한 움직임 방향을 고려하므로, 보다 정확한 움직임 추정이 가능함을 확인할 수 있다. 또한, 움직임이 아주 작은 화상 전화용의 실험 영상인 SALESMAN 영상에 대해서는 기존의 방법과 비슷한 결과를 나타내었다. 그림 9에는 프레임별 PSNR 결과를 나타내었는데, 전 프레임에 걸쳐서 비교적 고른 PSNR 분포 Lu 등의 방법의 정합 회수는 10~16회인데, 평균 정합 회수는 14.2~15.2회로 최를 나타내었다.

표 2은 제안한 고속 움직임 추정 방법과 기존의 고속 움직임 추정 방법의 계산량 비교를 나타낸 것이다. 표 2의 결과에서, Lu 등의 방법의 정합 회수는 10~16회인데, 평균 정합 회수는 14.2~15.2회로 최대 정합 회수에 치우치게 되고, 제안한 방법의 정합 회수는 12~23회인데, 평균 정합 회수는 12.9~14.5회로서 최소 정합 회수에 치우쳐서 구해진다. 그 이유는 Lu 등의 방법과 달리 제안한 방법에서는, 각 단계에서 움직임이 진행될지의 여부를 판단하여, 움직임이 더 이상 진행되지 않는다고 판단되면, 단계 진행을 중단하고 그 단계에서 세부 탐색하여, 움직임 벡터를 구하기 때문이다. 제안한 방법의 경우는 탐색점 수가 12점과 15점일 때, 즉 1단계와 2단계의 탐색에서 중도 정지할 경우가 가장 많이 발생하므로, 블록당 평균 블록 정합 회수는 표 2에서와 같다. 움직임이 작은 영상뿐만 아니라 움직임이 크고 불규칙한 영상에 대해서도, 제안한 방법의 블록 정합 회수가 TSS 방법보다는 훨씬 적으며, Lu 등의 방법에 비해서도 비슷하거나 더 적었다.

이상의 결과들로부터, 제안한 방법은, 기존의 고속 움직임 추정 방법들에 비해서 더 적은 계산량으로 우수한 PSNR을 나타냄을 확인할 수 있다.

한편 TSS 방법의 경우는 비교 연산이 필요 없으나, Lu의 방법 및 제안한 방법에서는, 단계를 진행하면서, 두 값의 크기를 비교하는 연산이 필요하다. Lu의 방법의 경우는 각 단계를 진행할 때마다 2번씩의 비교 연산이 필요하며, 3 단계의 움직임 추정 방법이므로 블록 당 6번의 비교 연산이 필요하다. 제안한 방법에서는 각 단계에서 중도 정지하는 경우가 많으므로, 단계마다 비교하는 연산 회수가 다르다. 제안한 방법의 비교 연산 회수를 구해본 GARDEN 영상에 대해서는 1.68회가 필요하다. 그러나 비교 연산의 계산량 부담은, 결과, TABLE TENNIS 영상에 대해서는 평균 비교 연산 회수가 1.18회, FLOWER 움직임 추정을 위한 블록 정합의 계산량에 비해 극히 작다. 즉, 8×8 블록에 대해 블록 정합을 1번 행하는 것은, 64개 화소와 64개 화소의 차를 각각 계산하여 이를 더해서 평균을 내는 과정이다. 그러므로, 1번의 비교 연산은 1회 블록

정합의 1/64에 해당하는 계산량 부담을 가지므로 무시할 수 있다.

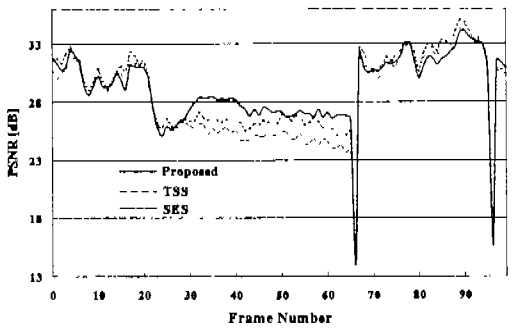
동영상 부호화에서는 기본적으로 탐색 영역의 크기를 블록 크기의 4배로 정하고 있다. 이는 동영상의 대부분은 움직임이 작으므로, 움직임 벡터가 이 영역 내에 존재할 확률이 약 90 %이다. 또한 움직임이 커서 움직임 추정이 정확하게 이루어지지 않은 영역이 약 10%에 해당되는데, 이들 영역의 보상 영상의 화질은 크게 떨어진다. 이를 위해서 탐색 영역의 크기를 크게 하는 것은, 계산량과 PSNR을 동시에 고려하는 동영상 부호화의 효율성에 어긋난다. 그러므로, 대부분의 동영상 부호화에서는, 움직임이 커서 움직임 추정이 정확하게 이루어지지 않은 영역에 대해서, 따로 프레임내 부호화를 하거나 움직임 보상된 차 성분에 대한 부호화 과정이 뒤따른다.

표 1. 제안한 방법과 기존 방법의 PSNR[dB] 비교

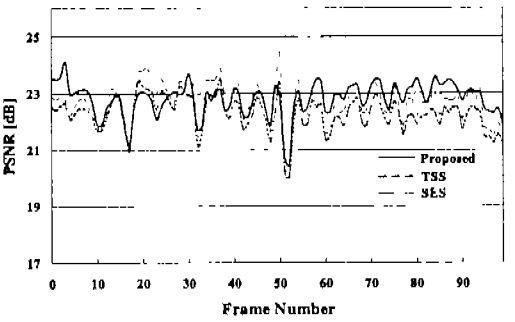
Sequences	FSBMA	TSS	Lu's	Proposed
FLOWER GARDEN	26.08	22.20	22.65	22.80
TABLE TENNIS	31.02	28.94	28.29	29.14
FOOTBALL	24.65	23.34	22.54	22.80
SALESMAN	36.42	36.05	35.91	35.85

표 2. 제안한 방법과 기존 방법의 평균 정합 회수 비교

Sequences	FSBMA	TSS	Lu's	Proposed
FLOWER GARDEN	225	25	14.2	14.5
TABLE TENNIS	225	25	14.7	13.4
FOOTBALL	225	25	14.7	13.9
SALESMAN	225	25	15.2	12.9



(a) TABLE TENNIS



(b) FLOWER GARDEN

그림 9. 제안한 방법과 기존 방법의 프레임별 PSNR[dB] 비교

V. 결 론

본 논문에서는 탐색 영역 내에서의 움직임 벡터의 분포 특성과 최적 정합점에 대한 각 탐색점에서의 블록 MAD의 특성을 고려한 새로운 고속 움직임 추정 방법을 제안하였다.

제안한 방법에서는 움직임 벡터의 분포 특성을 고려하여, 초기 단계의 참조 탐색점의 간격을 작게 설정함으로써, 정확한 움직임 추정을 행하였다. 또, 최적 탐색점에 대한 각 탐색점에서의 추정 오차의 특성을 고려하여, 현재 단계에서의 세부 탐색 방향은 이미 진행된 단계에서의 최소 추정 오차와 현재 단계에서의 최소 추정 오차의 크기를 서로 비교한 후 결정함으로써 탐색점 수를 줄인다. 모의 실험 결과, 움직임이 작은 영상에 대해서는 TSS 방법에 비해서 훨씬 적은 계산량으로 비슷하거나 우수한 화질을 얻을 수 있었으며, 움직임이 크고 분포가 불규칙한 영상에 대해서 성능이 저하되는 Lu 등의 방법에 비해서 화질과 계산량 측면에서 모두 우수한 결과를 얻을 수 있었다.

참 고 문 헌

[1] J. R. Jain, A. K. Jain, "Displacement measurement and its application in interframe image coding," IEEE Trans. on Commun.., vol. COM-29, no. 12, pp. 1799-1808, Dec. 1981.

[2] T. Koga, K. Iinuma, A Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," Proc. Nat. Telecommun. Conf., pp. G5.3.1-G5.3.5, Nov./

Dec. 1981.
[3] R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," IEEE Trans. Commun., vol. COM-33, pp. 1011-1014, Sept. 1985.
[4] M. Ganbari, "The cross search algorithm for motion estimation," IEEE Trans. Commun., vol. COM-38, no. 7, pp. 950-953, July 1990.
[5] R. Li, B. Zeng, M. L. Liou, "A new three-step search algorithm for block motion estimation," IEEE Trans. Circuits Syst., Video Technol., vol. 4, no. 4, pp. 438-442, August 1994.
[6] J. Lu, M. L. Liou, "A simple and efficient search algorithm for block-matching motion estimation," IEEE Trans. Circuits Syst., Video Technol., vol. 7, no. 2, pp. 429-433, April 1997.

최 정 현(Jung-Hyun Choi) 정회원
통신학회논문지 제24권 제1B호 참조

박 대 규(Dae-Gyue Park) 정회원
1972년 2월 21일생
1992년 3월~1998년 2월 : 경북대학교 전자공학과 학사
1998년 3월~현재 : 경북대학교전자공학과 석사과정
<주관심 분야> HDTV, 영상 신호 처리 및 압축

정 태 연(Tae-Yeon Jung) 정회원
통신학회논문지 제24권 제8B호 참조

이 경 환(Kyeong-Hwan Lee) 정회원
통신학회논문지 제24권 제1B호 참조

이 법 기(Bub-Ki Lee) 정회원
통신학회논문지 제24권 제1B호 참조

김 덕 규(Duk-Gyoo Kim) 정회원
통신학회논문지 제24권 제1B호 참조