논문 99-24-9A-15 한국통신학회논문지 '99-9 Vol.24 No.9A

블록 움직임 추정을 위한 2단계 고속 전역 탐색 알고리듬

정회원 정 원 식*, 이 법 기*, 이 경 환*, 최 정 현*, 김 경 규*, 김 덕 규*, 이 건 일*

Two-Stage Fast Full Search Algorithm for Block Motion Estimation

Won-Sik Cheong*, Bub-Ki Lee*, Kyeong-Hwan Lee*, Jung-Hyun Choi*, Kyeong-Kyu Kim*, Duk-Gyoo Kim*, and Kuhn-Il Lee* *Regular Members*

요 약

본 논문에서는 전역 탐색 알고리듬 (full search algorithm; FSA)과 동일한 성능을 나타내면서도 고속으로 움직임을 추정할 수 있는 블록 움직임 추정을 위한 2단계 고속 전역 탐색 알고리듬을 제안하였다. 제안한 방법에서는 첫 번째 단계에서 9:1로 부표본화된 탐색점에 대하여 블록 정합을 행하여, 여기서 얻어지는 최소 평균 절대치 오차 (mean absolute error; MAE)를 기준 MAE로 설정한다. 두 번째 단계에서는 첫 번째 단계에서 블록 정합을 행하지 않은 탐색점에 대하여 각 탐색점에서 가질 수 있는 MAE의 최소 범위를 구한 뒤, 이 값이 기준 MAE보다 작은 탐색점에 대하여서만 블록 정합을 행하였다. 이때, MAE의 최소 범위는 첫 번째 단계에서 블록 정합을 통하여 얻은 MAE들과 현재 블록 내의 화소들의 이웃 화소간의 화소 값의 차를 이용하여 구하였다. 그러므로, 제안한 방법에서는 MAE의 최소 범위를 이용하여 블록 정합이 필요한 블록에 대하여서만 정합을 행함으로써 FSA와 동일한 움직임 추정 성능을 유지하면서도 움직임 벡터의 추정을 위한 계산량을 줄일 수 있었다.

ABSTRACT

In this paper, we propose a two-stage fast full search algorithm for block motion estimation that produces the same performance to that of full search algorithm (FSA) but with remarkable computation reduction. The proposed algorithm uses the search region subsampling and the difference of adjacent pixels in the current block. In the first stage, we subsample the search region by a factor of 9, and then calculate mean absolute error (MAE) at the subsampled search points. And in the second stage, we reduce the search points that need block matching process by using the lower bound of MAE value at each search point. We get the lower bound of MAE value for each search point from the MAE values which are calculated at the first stage and the difference of adjacent pixels in the current block. The experimental results show that we can reduce the computational complexity considerably without any degradation of picture quality.

I. 서 론

통신 매체의 발달과 고성능의 계산 능력을 가진 컴퓨터의 등장으로 인하여 정지영상뿐만 아니라 동영상을 이용한 다양한 응용들이 나타나고 있다. 그러나, 디지털로 정지영상이나 동영상을 표현하기 위해서는 많은 데이터 량을 필요로 하므로, 이를 효율적으로 전송 및 저장하기 위해서는 잘 설계된 압축 방식이 필요하다.

동영상 압축을 위한 표준으로는 H.261,[1] H.263,[2]

* 경북대학교 전자전기공학부(formula@palgong.kyungpook.ac.kr)
논문번호 : 99253-0628, 접수일자 : 1999년 6월 28일

및 MPEG[3],[4] 등이 있다. 이들은 움직임 추정 및 보상과 이산 여현 변환 (discrete cosine transform)을 기반으로 하고 있다. 여기서 움직임 추정 및 보상은 연속된 프레임간의 시간적인 중복성을 제거함으로써 높은 압축율을 얻는데 핵심적인 역할을 담당하고 있다. 움직임 추정 및 보상 기법으로는 수행시간이 비교적 적게 소요되고, 하드웨어 구현이 용이한 블록 정합 알고리듬 (block matching algorithm; BMA)[5]이 많이 사용되고 있으며,[6] 정합 척도로는 평균 자승 오차 (mean squared error; MSE)와 비슷한 성능을 가지면서도 계산량이 적은 평균 절대치 오차 (mean absolute error; MAE)가 많이 사용된다.

BMA를 이용하여 움직임을 추정하는 경우에 가장 좋은 성능을 얻을 수 있는 방법은 이전 프레임에 설정된 탐색영역의 모든 탐색점에 대하여 탐색을 행하는 FSA이다. 이 방법에서는 모든 탐색점에 대하여 탐색을 행하여 정합 척도가 최적인 블록을 찾기 때문에 움직임 추정 오차 측면에서 최적인 움직임 벡터를 얻을 수 있지만 계산량이 많은 단점이 있다. 이와 같은 단점을 줄이기 위하여 움직임 벡터를 고속으로 추정할 수 있는 여러 가지 고속 움직임 추정 기법들이 제안되었다.

대표적인 고속 BMA 방법으로는 2차원 로그 탐색 (two dimensional logarithm search; 2-D LOG), 3단계 탐색 (three step search; TSS), 및 교차 탐색 (cross search) 알고리듬 등이 있으며,[5],[7]-[11] 이 방법들 중 간단하면서도 성능이 좋은 3단계 탐색 알고리듬이 가장 많이 사용된다. 그러나, 이 방법들은 움직임 추정 오차는 움직임 방향으로 단조 감소한다는 가정을 이용하여 탐색영역의 일부분에 대하여서만 탐색을 행하므로, 계산량은 줄일 수 있었지만 국부 최소 (local minimum)에 빠질 수 있는 단점을 가진다. 즉, 이 방법들은 움직임 추정의 정확성을 어느 정도 희생함으로써 계산량의 감소를 얻는다.

또한, Liu 등[12]은 전역 탐색을 수행하면서 움직임 추정을 고속으로 행하기 위한 방법으로 블록의 화소들을 수직 및 수평 방향으로 부표본화 (subsampling)한 뒤 이를 이용하여 블록 정합을 행하는 방법을 제안하였다. 그러나 이 방법에서도 블록의 화소의 일부분만이 블록 정합의 계산에 사용되므로 FSA에 비하여 성능이 떨어지는 단점을 가진다.

본 논문에서는 전역 탐색 알고리듬과 동일한 성능을 나타내면서도 고속으로 움직임을 추정할 수 있는 2단계 고속 전역 탐색 알고리듬을 제안하였다. 제안한 방법에서는 먼저, 첫 번째 단계에서 9:1로 부표본화된 탐색점에 대하여 블록 정합을 행한 뒤, 여기서 얻어지는 MAE 중에서 가장 작은 MAE를 기준 MAE (reference MAE)로 설정한다. 두 번째 단계에서는 첫 번째 단계에서 블록 정합을 행하지 않은 탐색점에 대하여 각 탐색점에서 가질 수 있는 MAE의 최소 범위를 구한 뒤, 이 값이 기준 MAE보다 작은 값을 갖는 탐색점에 대하여서만 블록 정합을 행하였다. 이는 각 탐색점에서 가질 수 있는 MAE의 최소 범위가 기준 MAE 보다 크다면, 블록 정합을 행하더라도 기준 MAE보다 작은 MAE를 가질 수 없으므로 블록 정합을 행할 필요가 없기 때문이다. 이때, MAE의 최소 범위는 첫 번째 단계에서 블록 정합을 통하여 얻은 MAE들과 현재 블록 내의 화소들의 이웃 화소간의 화소 값의 차를 이용하여 구하였다. 즉, 제안한 방법에서는 MAE의 최소 범위를 이용하여 블록 정합이 필요한 탐색점에 대하여서만 블록 정합을 행함으로써 FSA와 동일한 움직임 추정 성능을 유지하면서도 움직임 벡터의 추정을 위한 계산량을 줄일 수 있었다

제안한 방법의 성능을 평가하기 위한 여러 가지 동영상에 대한 컴퓨터 모의 실험을 통하여, 제안한 방법이 FSA와 동일한 성능을 유지하면서 많은 계산량의 감소를 얻을 수 있음을 확인하였다.

Ⅱ. 블록 정합 알고리듬을 이용한 움직임 추정

BMA는 현재 입력 프레임을 일정한 크기의 블록으로 나눈 후, 이전 프레임에서 설정된 탐색영역에서 정합 척도가 최적인 위치를 찾는 것이다. 이때 사용되는 정합 척도로는 MSE와 비슷한 성능을 유지하면서도 계산량이 적고 하드웨어 구현이 용이한 MAE가 널리 이용되고 있다. 이때, MAE는

$$MAE_{(k,l)}(x,y) = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} | I_t(k+i, l+j) - I_{t-1}(k+x+i, l+y+j) | \quad (1)$$

와 같다. 여기서, N은 블록의 수직 및 수평 방향의 크기, $I_t(i,j)$는 현재 프레임의 (i,j) 좌표에서 화소의 휘도 값, $I_{t-1}(i,j)$는 이전 프레임의 (i,j) 좌표에서 화소의 휘도 값, (k,l)은 현재 블록의 위치 좌표, 그리고 (x,y)는 탐색영역에서의 탐색점의 위치를 나타낸다. 이를 이용하여 (k,l)번째 블록의 움직임 벡터는

$$\mathbf{v}(k,l) = \arg \min_{(x,y)} MAE_{(k,l)}(x,y) \quad (2)$$

와 같이 구한다.

BMA를 이용하여 움직임 추정을 행하는 경우에 가장 좋은 성능을 얻을 수 있는 방법은 FSA로서, 이 방법은 정합 될 수 있는 모든 탐색점에 대해 MAE를 구하고, 그 중 가장 작은 MAE 값을 갖는 탐색점의 좌표 (x,y)를 움직임 벡터로 결정하는 것이다. 이때 $N \times N$ 화소 크기의 블록의 움직임 벡터의 수직 및 수평 방향의 최대 변위가 w 인 경우에 탐색영역에서 정합을 행하여야 할 탐색점수는 $(2w+1)^2$다. 따라서 FSA는 움직임 추정 오차 측면에서 최적인 움직임 벡터를 추정할 수 있지만, 탐색점 수가 너무 많으므로 계산량이 많은 단점이 있다.

이와 같은 단점을 줄이기 위하여 탐색점 수를 줄이는 방법과 블록의 화소들을 부표본화한 뒤 블록 정합을 행하는 방법 등이 연구되었다. 탐색영역에서 탐색점 수를 줄이는 방법의 경우에는 움직임 추정 오차는 움직임 방향으로 단조 감소한다는 가정을 이용하여 전역 탐색을 행하지 않고 탐색영역의 일부에 대해서만 탐색을 행하므로, 계산량은 크게 줄일 수 있었지만 국부 최소에 빠질 수 있는 단점을 가진다. 즉, 움직임 추정의 정확성을 어느 정도 희생함으로써 계산량의 감소를 얻기 때문에 움직임 추정 오차가 커지며, 특히 이 방법에서 사용된 가정이 잘 맞지 않는 경우에는 움직임 추정 오차가 매우 커지는 단점이 있다. 움직임 추정 오차의 큰 증가는 복원영상에서 블록화 현상 등의 현저한 화질 열화를 일으킬 수 있으며, 움직임 보상 오차의 전송에 필요한 비트율을 크게 증가 시킬수 있다. 그러므로 고품질 동영상 부호화기에서는 FSA가 많이 사용되고 있으며,[13],[14] FSA와 동일한 성능을 유지하면서도 움직임 추정에 필요한 계산량을 줄이기 위한 방법이 연구되고 있다.[15],[16]

또한, Liu 등[12]은 탐색 영역의 모든 탐색점에 대하여 탐색을 수행하면서 움직임 추정을 고속으로 행하기 위한 방법으로 블록의 화소들을 수직 및 수평방향으로 부표본화 한 뒤 이를 이용하여 블록 정합을 행하는 방법을 제안하였다. 그러나 이 방법에서도 블록의 화소의 일부분만이 블록 정합의 계산에 사용되므로 블록내의 화소들이 비슷한 값을 가지는 평탄한 블록의 경우에는 우수한 움직임 추정 성능을 나타내지만, 복잡한 블록의 경우에는 부표본화에 의한 해상도의 저하로 인하여 움직임 추정 오차가 커지는 단점을 가진다. 그러므로 움직임 추정 오차 측면에서 최적인 FSA에 가까운 움직임 추정 성능을 유지하면서도 계산량을 줄일 수 있는 방법이 필요하다.

Ⅲ. 제안한 2단계 고속 전역 탐색 알고리듬

본 논문에서는 FSA와 동일한 성능을 나타내면서도 고속으로 움직임을 추정할 수 있는 2단계 고속 전역 탐색 알고리듬 (two-stage fast full search algorithm; TFFSA)을 제안하였다. 제안한 방법에서는 첫 번째 단계에서 9:1로 부표본화된 탐색점에 대하여 블록 정합을 행하였다. 그리고 두 번째 단계에서는 첫 번째 단계에서 블록 정합을 행하지 않은 탐색점에 대하여 각 탐색점에서 가질 수 있는 MAE의 최소 범위를 이용하여 블록 정합 수행 여부를 결정한 뒤, 블록 정합이 필요한 탐색점에 대하여서만 블록 정합을 행하였다. 이때, MAE의 최소 범위는 이웃 탐색점에서의 MAE와 현재 블록 내 화소들의 이웃 화소간의 화소값 차를 이용하여 구하였다. 즉, 두 번째 단계에서는 각 탐색점에서 가질 수 있는 MAE의 최소 범위를 이용하여 블록 정합이 필요한 탐색점의 수를 줄임으로써 FSA와 동일한 성능을 얻으면서도 고속으로 움직임을 추정을 행하였다.

1. 이웃 탐색점에서의 오차 블록을 이용한 MAE 계산

수평 및 수직 방향의 크기가 N인 현재 블록 C와 탐색 영역의 (x,y) 위치에서의 후보 정합 블록 (candidate matching block) $M(x,y)$의 화소들의 휘도 값을 $N \times N$ 행렬로 나타내면 각각

$$\mathbf{C} = [\ \mathbf{C}_0 \ \cdots \ \mathbf{C}_{N-1}\ ] \quad (3)$$

$$where,\ \mathbf{C}_k = [\ c_{0,k} \ \cdots \ c_{N-1,k}\ ]^T$$

$$\mathbf{M}(x,y) = [\ \mathbf{M}_0(x,y) \ \cdots \ \mathbf{M}_{N-1}(x,y)\ ] \quad (4)$$

$$where,\ \mathbf{M}_k(x,y) = [\ m_{y,x+k} \ \cdots \ m_{y+N-1,x+k}\ ]^T$$

와 같고, 이들의 오차 블록 $\mathbf{D}(x,y) = \mathbf{M}(x,y) - \mathbf{C}$는

$$\mathbf{D}(x,y) = [\ \mathbf{D}_0(x,y) \ \cdots \ \mathbf{D}_{N-1}(x,y)\ ]$$

$$= [\ \mathbf{M}_0(x,y) - \mathbf{C}_0 \ \cdots \ \mathbf{M}_{N-1}(x,y) - \mathbf{C}_{N-1}\ ]$$

$$where,\ \mathbf{D}_k(x,y) = [\ d_{y,x+k} \ \cdots \ d_{y+N-1,x+k}\ ]^T \quad (5)$$

와 같다. 그리고, 탐색점 (x,y)에서의 MAE는

$$MAE(x,y) = \frac{1}{N^2} \| \mathbf{M}(x,y) - \mathbf{C} \| = \frac{1}{N^2} \| \mathbf{D}(x,y) \| \quad (6)$$

와 같이 구할 수 있다.

또한 탐색점 (x,y)에서 블록 정합을 행하였다면, 이 탐색점에서의 오차 블록을 이용하여 이웃 블록의 MAE를 계산할 수 있다. 예를 들어 블록 정합을 행한 탐색점 (x,y)의 수평 방향으로 이웃 탐색점 $(x+1,y)$에서의 오차 블록 $\mathbf{D}(x+1,y) = \mathbf{M}(x+1,y) - \mathbf{C}$는

$$\begin{aligned}\mathbf{D}(x+1,y) &= [\ \mathbf{D}_0(x+1,y) \quad \cdots \quad \mathbf{D}_{N-2}(x+1,y) \quad \mathbf{D}_{N-1}(x+1,y)\ ] \\ &= [\ \mathbf{M}_0(x+1,y) - \mathbf{C}_0 \cdots \mathbf{M}_{N-2}(x+1,y) - \mathbf{C}_{N-2} \quad \mathbf{M}_{N-1}(x+1,y) - \mathbf{C}_{N-1}\ ] \\ &= [\ \mathbf{M}_1(x,y) - \mathbf{C}_0 \cdots \mathbf{M}_{N-1}(x,y) - \mathbf{C}_{N-2} \quad \mathbf{M}_N(x,y) - \mathbf{C}_{N-1}\ ] \quad (7)\end{aligned}$$

와 같다. 이를 이미 블록 정합을 행한 탐색점 (x,y)에서의 오차 블록 $\mathbf{D}(x,y)$를 이용하여 표현하기 위하여 현재 블록 내의 화소들의 이웃 화소간의 수평 방향으로의 화소값의 차를

$$\mathbf{D}_{CH+} = [\ \mathbf{C}_0 - \mathbf{C}_{N-1} \quad \mathbf{C}_1 - \mathbf{C}_0 \cdots \mathbf{C}_{N-1} - \mathbf{C}_{N-2}] \quad (8)$$

와 같이 나타내면, $\mathbf{D}(x,y)$와 $\mathbf{D}_{CH+}$의 합은

$$\begin{aligned}\mathbf{D}(x,y) + \mathbf{D}_{CH+} &= [\ \mathbf{M}_0(x,y) - \mathbf{C}_{N-1} \\ &\quad \mathbf{M}_1(x,y) - \mathbf{C}_0 \cdots \mathbf{M}_{N-1}(x,y) - \mathbf{C}_{N-2}\ ] \quad (9)\end{aligned}$$

와 같이 나타낼 수 있다. 이때 이 식을 식 (7)과 비교해 보면, 식 (7)의 마지막 열을 제외한 나머지 부분과 식 (9)의 첫 번째 열을 제외한 나머지 부분이 동일함을 알 수 있다. 그러므로 $\mathbf{D}(x,y)$의 첫 번째 열 $\mathbf{M}_0(x,y) - \mathbf{C}_0$를 $\mathbf{M}_N(x,y) - \mathbf{C}_0$로 대치시킨 새로운 오차 블록 $\hat{\mathbf{D}}_{H+}(x,y)$를

$$\begin{aligned}\hat{\mathbf{D}}_{H+}(x,y) &= [\ \mathbf{M}_N(x,y) - \mathbf{C}_0 \\ &\quad \mathbf{M}_1(x,y) - \mathbf{C}_1 \cdots \mathbf{M}_{N-1}(x,y) - \mathbf{C}_{N-1}\ ] \quad (10)\end{aligned}$$

와 같이 구한 뒤, 이를 $\mathbf{D}_{CH+}$와 더하면

$$\begin{aligned}\hat{\mathbf{D}}_{H+}(x,y) + \mathbf{D}_{CH+} &= [\ \mathbf{M}_N(x,y) - \mathbf{C}_{N-1} \\ &\quad \mathbf{M}_1(x,y) - \mathbf{C}_0 \cdots \mathbf{M}_{N-1}(x,y) - \mathbf{C}_{N-2}\ ] \quad (11)\end{aligned}$$

와 같다. 이를 식 (7)의 $\mathbf{D}(x+1,y)$와 비교해 보면, 각 열이 오른쪽으로 순환 이동 (circular shift)된 형태임을 알 수 있다. 그러므로 탐색점 $(x+1,y)$에서의 MAE는

$$MAE(x+1,y) = \frac{1}{N^2} \| \mathbf{M}(x+1,y) - \mathbf{C} \| = \frac{1}{N^2} \| \hat{\mathbf{D}}_{H+}(x,y) + \mathbf{D}_{CH+} \| \quad (12)$$

와 같이 구할 수 있다. 이 식으로부터 $MAE(x+1,y)$는 미리 구해진 탐색점 (x,y)에서의 오차 블록으로부터 구할 수 있음을 알 수 있다.

2. MAE의 최소 범위

앞 절에서 살펴본 바와 같이 각 탐색점에서의 MAE는 이웃 탐색점에서의 오차 블록과 현재 블록 내의 화소들의 이웃 화소간의 화소값의 차를 이용하여 식 (12)에서와 같이 구할 수 있다. 그러나 이와 같은 방법으로 $MAE(x+1,y)$를 구한다면 $\mathbf{D}_{CH+}$를 구하는 과정과 $\hat{\mathbf{D}}_{H+}(x,y)$의 첫 번째 열을 대치시키는 과정으로 인하여 계산량의 이득을 얻을 수 없다. 그러므로 본 논문에서는 삼각 부등식 (triangular inequality)을 이용하여 $MAE(x+1,y)$의 최소 범위를 구한 뒤 이를 이용하여 움직임 추정을 위한 계산량을 줄인다.

임의의 $N \times N$ 행렬 $\mathbf{A}$의 합의 놈 (sum norm)은

$$\| \mathbf{A} \| = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} | a_{ij} | \quad (13)$$

와 같이 정의된다.[15] 그리고, 삼각 부등식은

$$\begin{aligned}|\ \| \mathbf{A}_1 \| - \| \mathbf{A}_2 \|\ | &\le \| \mathbf{A}_1 + \mathbf{A}_2 \| \\ &\le \| \mathbf{A}_1 \| + \| \mathbf{A}_2 \| \quad (14)\end{aligned}$$

와 같고, 이를 식 (12)에 적용하면

$$\begin{aligned}|\ \| \hat{\mathbf{D}}_{H+}(x,y) \| - \| \mathbf{D}_{CH+} \|\ | &\le \| \hat{\mathbf{D}}_{H+}(x,y) + \mathbf{D}_{CH+} \| \\ &\le \| \hat{\mathbf{D}}_{H+}(x,y) \| + \| \mathbf{D}_{CH+} \| \quad (15)\end{aligned}$$

와 같다. 또한 식 (12)로부터 $\| \hat{\mathbf{D}}_{H+}(x,y) + \mathbf{D}_{CH+} \|$

$= \| \mathbf{M}(x+1,y) - \mathbf{C} \|$이므로 이를 식 (15)에 대입한 뒤, 식의 양변을 N^2으로 나누면 식 (15)는

$$\left| \frac{1}{N^2} \| \hat{\mathbf{D}}_{H+}(x,y) \| - \frac{1}{N^2} \| \mathbf{D}_{CH+} \| \right| \le MAE(x+1,y) \le \frac{1}{N^2} \| \hat{\mathbf{D}}_{H+}(x,y) \| + \frac{1}{N^2} \| \mathbf{D}_{CH+} \| \quad (16)$$

와 같다. 이 식을 살펴보면 구하고자 하는 탐색점 $(x+1,y)$에서의 MAE의 최소 및 최대 값을 현재 블록 내의 화소들의 이웃 화소간의 화소값의 차 $\mathbf{D}_{CH+}$와 탐색점 (x,y)에서의 오차 블록 $\mathbf{D}(x,y)$의 첫 번째 열을 대치한 블록 $\hat{\mathbf{D}}_{H+}(x,y)$로부터 구할 수 있음을 알 수 있다. 이때 식 (16)에서 $\| \mathbf{D}_{CH+} \|$는 전체 탐색 영역에 대하여 한번만 계산하면 되고, $\| \hat{\mathbf{D}}_{H+}(x,y) \|$는 N개의 열 중에서 첫 번째 열을 대치시키는 과정이 필요하므로, 블록 정합을 행하는 경우에 비하여 매우 적은 계산량으로 MAE의 최소 범위를 구할 수 있다. 그러므로 이를 이용하여 블록 정합이 필요한 탐색점에 대하여서만 정합을 행한다면 계산량을 크게 줄일 수 있다. 그리고 이것은 수직 및 대각선 방향의 탐색점에 대하여서도 동일하게 적용할 수 있다.

3. 2단계 고속 전역 탐색 알고리듬

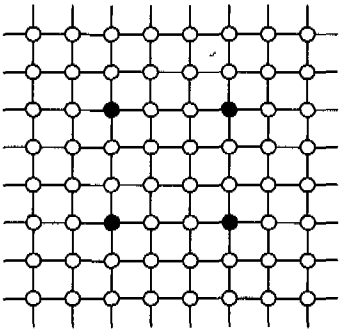
제안한 TFFSA에서는 첫 번째 단계에서 9:1로 부표본화된 탐색점에 대하여 블록 정합을 행하였다. 그리고 두 번째 단계에서는 각 탐색점에서 가질 수 있는 MAE의 최소 범위를 이용하여 블록 정합이 필요한 탐색점의 수를 줄임으로써 FSA와 동일한 성능을 얻으면서도 고속으로 움직임을 추정을 행하였다.

3.1. 1단계-탐색영역 부표본화 및 기준 MAE 계산

각 탐색점에서의 MAE의 최소 범위는 현재 블록 내 화소들의 이웃 화소간의 화소 값 차와 이웃 탐색점에서의 MAE로부터 구할 수 있다. 그러나, 현재 탐색점에서의 MAE의 최소 범위를 구하기 위해서는 이웃 탐색점에서의 MAE를 이용하여야한다. 그러므로 제안한 방법의 첫 번째 단계에서는 탐색영역의 탐색점을 그림 1에서와 같이 9:1로 부표본화하여 이들에 대한 MAE들을 구한다. 또한, 두 번째 단계에서 블록 정합을 행할 것인지를 결정하기 위하여 사용되는 기준 평균 절대치 오차 $MAE_{Ref.}$를

$$MAE_{Ref.} = \min MAE(x,y) \quad (17)$$

where, (x,y): The subsampled search points for the first stage.



●: The subsampled search points for the first stage.
○: The search points for the second stage.

그림 1. 탐색영역 부표본화

와 같이 부표본화 된 탐색점들의 MAE 중에서 최소 MAE로 정한다.

3.2. 2단계-MAE의 최소 범위를 이용한 탐색점 줄임

두 번째 단계에서는 첫 번째 단계에서 구한 $MAE(x,y)$를 이용하여 $\| \hat{\mathbf{D}}_k(x,y) \|$를 구한 뒤, (x,y)의 8개 이웃 탐색점 $(x\pm1,y)$, $(x,y\pm1)$ 및 $(x\pm1,y\pm1)$에서의 MAE의 최소 범위 $\left| \frac{1}{N^2} \| \hat{\mathbf{D}}_k(x,y) \| - \frac{1}{N^2} \| \mathbf{D}_{Ck} \| \right|$를 구한다. 제안한 방법에서는 이렇게 구한 MAE의 최소 범위를 이용하여

$$IF \left(MAE_{Ref.} < \left| \frac{1}{N^2} \| \hat{\mathbf{D}}_k(x,y) \| - \frac{1}{N^2} \| \mathbf{D}_{Ck} \| \right| \right)$$
No block matching
$$ELSE \{$$
Block matching
$$IF(MAE(i,j) < MAE_{Ref.})$$
$$MAE_{Ref.} = MAE(i,j)$$
$$\} \quad (18)$$

where, (i,j): The search points for the second stage.
$k = H+, H-, V+, V-, D_1, D_2, D_3, D_4$

와 같이 MAE의 최소 범위가 기준 MAE인 $MAE_{Ref.}$보다 작은 탐색점에 대하여서만 블록 정합

을 행한다. 즉, 제안한 방법에서는 식 (17)에서와 첫 번째 단계에서 블록 정합을 행한 탐색점을 제외한 나머지 탐색점에 대하여 그 탐색점에서 가질 수 있는 MAE의 최소 범위를 이웃 탐색점에서의 MAE와 현재 블록 내의 화소들의 이웃 화소간의 화소값의 차를 이용하여 구한 뒤, 이 값이 기준 평균 절대치 오차 $MAE_{Ref.}$보다 큰 경우에는 블록 정합을 수행하지 않는다. 또한, MAE의 최소 범위가 $MAE_{Ref.}$보다 작은 경우에는 이 탐색점에서의 MAE가 $MAE_{Ref.}$보다 작을 가능성이 있으므로 블록 정합을 행하여 이 탐색점에서의 MAE인 $MAE(i,j)$를 구한 뒤 $MAE_{Ref.}$와 비교한다. 이때 $MAE(i,j)$가 $MAE_{Ref.}$보다 작으면 $MAE_{Ref.}$는 $MAE(i,j)$로 갱신되고, 이것의 최종값을 지닌 탐색점이 최종 움직임 벡터로 결정된다. 이와 같은 방법으로 제안한 방법에서는 현재 블록의 움직임 탐색 영역의 모든 탐색점에 대하여 탐색을 행하지 않고도, 움직임 추정 오차 측면에서 최적인 움직임 벡터를 찾을 수 있다.

4. 제안한 방법의 계산량

FSA는 이전 프레임에 설정된 모든 탐색점에 대하여 MAE를 계산하며, 식 (1)로부터 MAE 계산에는 $2N^2$번의 덧셈과 뺄셈의 계산이 필요함을 알 수 있다. 그러므로 현재 프레임의 블록의 전체 개수가 T이고, 움직임 탐색 영역이 상하, 좌우의 최대 변위가 w이면 한 프레임에 대한 전역 탐색 블록 정합 알고리즘의 총 계산량은

$$C_{FSA} = 2T(2w+1)^2N^2 \tag{19}$$

와 같다.

반면, 제안한 방법의 계산량은 식 (17)의 계산량과 식 (18)의 계산량의 합으로 볼 수 있다. 먼저 식 (17)의 계산량을 살펴보면, 이 식을 이용하여 전체 탐색점의 1/9에 해당하는 부표본화된 탐색점에 대하여 블록 정합을 행하여야하므로 FSA의 계산량의 1/9에 해당하는 $2T(2w+1)^2N^2/9$의 계산량이 필요하다. 식 (18)의 계산에서는 전체 탐색 영역의 8/9에 해당하는 탐색점에 대하여 MAE의 최소 범위를 구하기 위한 계산량과 MAE의 최소 범위가 $MAE_{Ref.}$보다 작은 탐색점에서의 블록 정합을 위한 계산량이 필요하다. 여기서, MAE의 최소 범위를 구하기 위해서는 수평, 수직 및 대각선 방향에 대응되는 $\| \hat{\mathbf{D}}_k \|$와 $\| \mathbf{D}_{Ck} \|$를 구하여야 한다. 이때, $\| \hat{\mathbf{D}}_k \|$의 계산에는 수평 및 수직 방향의 탐색점에 대하여서는 탐색점 당 $4N$번의 덧셈과 뺄셈의 계산이 필요하고, 대각선 방향의 탐색점에 대하여서는 수평 및 수직 방향의 계산결과를 이용하면 이에 필요한 계산량은 거의 무시할 수 있다. 그리고, $\| \mathbf{D}_{Ck} \|$의 계산을 위해서는 전체 탐색영역에 대하여 각 방향에 대한 $2N^2$번의 덧셈과 뺄셈의 계산이 필요하지만 $\| \mathbf{D}_{Ck} \|$는 180° 의 차이를 갖는 경우에 대하여서는 동일한 값을 갖는다. 즉, $\| \mathbf{D}_{CH+} \|$와 $\| \mathbf{D}_{H-} \|$는 동일한 값이다. 또한, MAE의 최소 범위가 $MAE_{Ref.}$보다 작은 탐색점에 대하여 블록 정합을 행하기 위한 계산량은 탐색점당 $2N^2$이다. 그러므로 제안한 TFFSA의 계산량은

$$C_{TFFSA} = \frac{2}{9}TN^2(2w+1)^2 + \frac{16}{9}TN(2w+1)^2 + 8TN^2 + 2PN^2 \tag{20}$$

와 같다. 여기서 F는 식 (18)에서 MAE의 최소 범위가 $MAE_{Ref.}$보다 작은 탐색 점의 총 개수이다. 이 식으로부터 제안한 방법의 계산량은 MAE의 최소 범위가 기준 평균 절대 오차 $MAE_{Ref.}$보다 큰 탐색점 수가 많을수록 전역 탐색 블록 정합 알고리듬의 계산량에 비하여 많이 감소됨을 알 수 있다.

Ⅳ. 실험 결과 및 고찰

본 논문에서 제안한 방법의 성능을 평가하기 위하여 컴퓨터 모의실험을 행하였다. 본 실험에서는 720×480의 공간해상도를 가지는 FOOTBALL, FLOWER GARDEN, MOBILE, 및 SUSIE 영상 각 40프레임을 사용하였으며, 블록 정합에 사용된 블록의 크기는 16×16, 탐색 영역의 크기는 수직과 수평방향으로 -16~15로 MPEG의 권고안과 동일하게 하였다.[17] 본 실험에서 사용한 FOOTBALL 영상은 운동장을 배경으로 가지는 전체적으로 복잡하고, 물체 및 카메라의 움직임이 매우 빠른 영상이며, FLOWER GARDEN 영상은 꽃밭과 같은 복잡한 부분이 많이 존재하고, 정지된 물체에 카메라가 빠르게 움직이는 영상이다. 그리고 MOBILE 영상은 고정된 배경에 기차가 움직이는 영상으로서 움직임이 거의 없는 배경과 물체의 움직임이 빠른 영역이 혼재 하는 영상이며, SUSIE 영상은 비교적 단순한 영상으로 큰 움직임이 없는 영상이다. 이들 영

상을 그림 2에 나타내었다. 그리고 정합 척도로는 계산량이 적은 MAE를 사용하였다. 또한 본 실험에서는FSA와 제안한 TFFSA에 대한 결과를 계산량 및 PSNR을 이용하여 평가하였다.

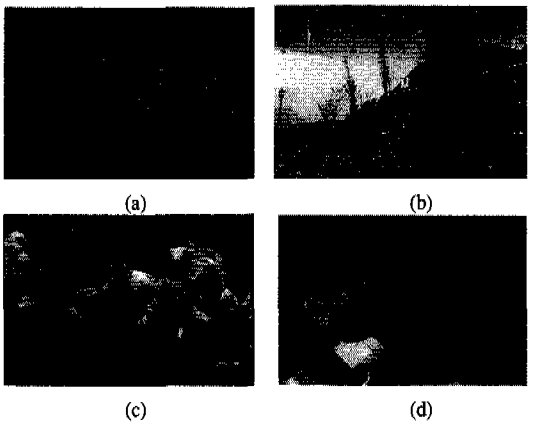


그림 2. 본 실험에서 사용된 (a) MOBILE, (b) FLOWER GARDEN, (c) FOOTBALL, 및 (d) SUSIE 영상

표 1. 평균 PSNR [dB] 비교

Sequences	MOBILE	FLOWER GARDEN	FOOT BALL	SUSIE
FSA	32.66	26.67	25.98	35.52
TFFSA	32.66	26.67	25.98	35.52

MOBILE, FLOWER GARDEN, FOOTBALL 및 SUSIE 영상에 대한 40 프레임 평균 PSNR을 표 1에 나타내었다. 이 표로부터 제안한 TFFSA와 FSA의 평균 PSNR이 동일함을 알 수 있다. 이는 제안한 TFFSA는 식 (18)에서와 같이 블록 정합의 수행 여부를 각 탐색점에서의 MAE의 최소값을 이용하여 결정하여, 기준 MAE보다 작은 MAE를 가질 수 없는 탐색점에 대하여는 블록 정합을 행하지 않고, 블록 정합이 필요한 부분에 대하여서만 블록 정합을 행하므로 FSA와 동일한 성능을 얻을 수 있기 때문이다. 또한, 표 1에서 움직임과 영상의 복잡한 부분이 FLOWER GARDEN 및 FOOTBALL 영상보다 상대적으로 적은 MOBILE 영상과 SUSIE 영상의 PSNR이 크게 나타남을 알 수 있다. 또한 영상의 복잡한 부분은 비슷하지만 FLOWER GARDEN 영상보다 빠른 움직임을 가지는 FOOTBALL영상의 PSNR이 가장 작음을 볼 수 있다.

FSA와 제안한 TFFSA의 블록 정합을 행한 탐색점 수의 40 프레임 평균치를 표 2에 나타내었다. 이 표에 나타난 수치는 FSA의 탐색점 수를 100%로 하여 제안한 방법의 탐색점 수를 FSA의 탐색점 수에 대하여 상대적으로 나타내었다. 이 표로부터 제안한 TFFSA의 경우 블록 정합이 필요한 탐색점 수는 FSA의 약 35%~45%로 FSA에 비하여 훨씬 적음을 알 수 있다. 이를 자세히 살펴보기 위하여 각 실험 영상 40 프레임에 대한 제안한 TFFSA의 FSA에 대한 탐색점 감소율을 그림 3에 나타내었다.

표 2. FSA에 대한 제안한 TFFSA의 평균 탐색점 수 비교

Sequences	MOBILE	FLOWER GARDEN	FOOT BALL	SUSIE
FSA	100.0 %	100.0 %	100.0 %	100.0 %
TFFSA	38.3 %	35.4 %	43.0 %	44.8 %
Reduction ratio	61.7 %	64.6 %	57.0 %	55.2 %

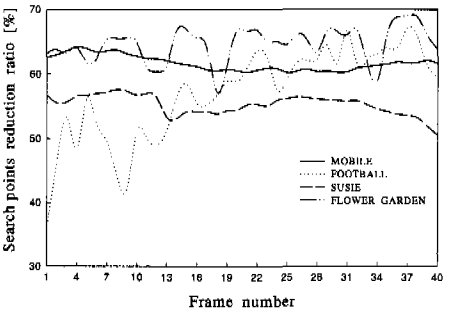


그림 3. 제안한 TFFSA의 탐색점 감소율

이 그림을 살펴보면 SUSIE 영상 및 MOBILE 영상의 경우에는 프레임에 따른 탐색점 감소율의 변화가 거의 없음을 확인할 수 있다. 즉, 식 (18)을 살펴보면 제안한 TFFSA의 탐색점 수는 MAE의 최소 범위가 기준 MAE보다 큰 탐색점의 수에 따라 달라질 수 있음을 알 수 있지만, 이 그림으로부터 탐색점 수의 변화량이 평균 값을 중심으로 큰 변화가 없음을 확인할 수 있다. 또한, FOOTBALL 영상의 경우에는 탐색점 감소율이 프레임에 따라 약 20% 정도의 변동폭을 가짐을 알 수 있다. FOOTBALL 영상의 앞부분에서는 풋볼 선수들의 빠른 움직임으로 인하여 이전 프레임에는 없던 선수들이 나타나기 때문에 이전 프레임에는 없는 새로운 영역이 많이 발생하게된다. 그러므로 이 부분

에 대하여서는 탐색점의 감소가 다소 적음을 알 수 있고, 프레임이 진행되면서 많은 탐색점의 감소를얻으면서 탐색점 감소율이 안정화됨을 이 그림으로부터 확인할 수 있다. 또한, FLOWER GARDEN 영상의 경우에는 프레임에 따라 약간의 탐색점 감소율의 변화가 있음을 알 수있다. 이상의 결과로부터 제안한 방법의 경우에는 움직임 추정 오차 측면에서 최적인 FSA와 동일한 움직임 추정 성능을 유지하면서도 평균 65%~55% 정도의 탐색점 감소를 얻을 수 있음을 확인할 수 있었다.

또한, 제안한 TFFSA의 계산량은 식 (20)에서 볼 수 있는바와 같이 블록 정합을 위한 계산량과 블록 정합 여부를 결정하기 위한 부가적인 계산량으로 이루어진다. 제안한 TFFSA의 FSA에 대한 계산량의 감소율을 그림 4에 나타내었다. 이 그림에서 나타난 제안한 방법의 FSA에 대한 계산량의 감소 정도 R은

$$R = (1 - \frac{C_{proposed}}{C_{FSBMA}}) \times 100 \quad \% \qquad (21)$$

와 같이 표현 될 수 있다. 이 그림을 살펴보면 제안한 방법의 부가적인 계산량은 모든 프레임에 대하여 동일한 양이므로, 그림 3에 나타낸 제안한 방법의 탐색점 감소율과 동일한 모양을 가짐을 알 수 있다. 또한 이 그림으로부터 제안한 TFFSA이 영상에 따라 다소의 차이는 있지만 FSBMA에 비하여 약 60%에서 50% 정도의 계산량 감소를 얻을 수 있음을 알 수 있다.

이상의 모의 실험 결과로부터 제안한 TFFSA는

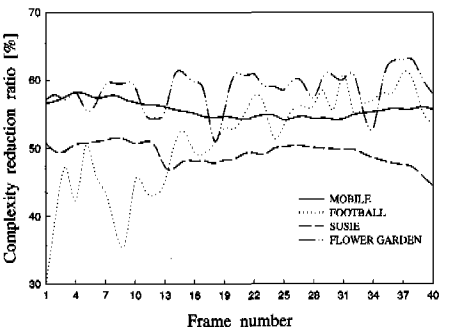


그림 4. 제안한 TFFSA의 계산량 감소율

FSA와 동일한 움직임 추정 오차를 유지하면서도 블록 정합이 필요한 탐색점 수는 65%~55%, 전체적인 계산량은 60%~50% 정도 줄일 수 있음을 확인할 수 있었다.

V. 결 론

본 논문에서는 전역 탐색 알고리듬과 동일한 성능을 나타내면서도 고속으로 움직임을 추정할 수 있는 블록 움직임 추정을 위한 2단계 고속 전역 탐색 알고리듬을 제안하였다. 제안한 방법에서는 첫 번째 단계에서 9:1로 부표본화된 탐색점에 대하여 블록 정합을 행하여, 여기서 얻어지는 최소 MAE를 기준 MAE (reference MAE)로 설정한다. 두 번째 단계에서는 첫 번째 단계에서 블록 정합을 행하지 않은 탐색점에 대하여 각 탐색점에서 가질 수 있는 MAE의 최소 범위를 구한 뒤, 이를 기준 MAE와 비교하여 기준 MAE보다 작은 값을 갖는 탐색점에 대하여서만 블록 정합을 행하였다. 이때, MAE의 최소 범위는 첫 번째 단계에서 블록 정합을 통하여 얻은 MAE들과 현재 블록 내의 화소들의 이웃 화소간의 화소 값의 차를 이용하여 구하였다. 즉, 제안한 방법에서는 MAE의 최소 범위를 이용하여, 블록 정합이 필요한 탐색점의 수를 줄임으로써 FSA와 동일한 움직임 추정 성능을 유지하면서도 움직임 벡터의 추정을 위한 계산량을 줄일 수 있었다

제안한 방법의 성능을 평가하기 위한 여러 가지 동영상에 대한 컴퓨터 모의 실험을 통하여, 제안한 방법은 FSBMA와 동일한 움직임 추정 오차를 유지하면서도 블록 정합이 필요한 탐색점 수는 65%~55%, 전체적인 계산량은 60%~50% 정도 줄일 수 있음을 확인할 수 있었다.

참 고 문 헌

[1] *ITU-T Recommendation H.261*, "Video codec for audiovisual services at p×64 kbits/s."

[2] *Draft ITU-T Recommendation H.263*, "Video coding for low bit rate communication."

[3] *ISO/IEC 11172-2*, "Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5Mbits/s: Video."

[4] *ISO/IEC 13818-2*, "Information technology - Generic Coding of Moving Pictures and Associated Audio Information: Video."

[5] J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Trans. Commun.*, vol.

29, no. 12, pp. 1799-1801, Dec., 1981.

[6] K. M. Yang, M. T. Sun, and L. Wu, "A family of VLSI design for the motion compensation block-matching algorithm," *IEEE Trans. Circuit and Systems*, vol. 36, no. 10, pp. 1309-1316, 1989.

[7] 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.

[8] R. Strinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," *IEEE Trans. Commun.*, vol. COM-33, pp. 1011-1014, Sept. 1985.

[9] M. Ganbari, "The cross search algorithm for motion estimation," *IEEE Trans. Commun.*, vol. COM-38, no. 7, pp. 950-953, July 1990.

[10] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. Circuit and Systems for Video Technology*, vol. 4, no. 4, pp. 438-442, Aug. 1994.

[11] J. Lu and M. L. Liou, "A simple and efficient search algorithm for block-matching motion estimation," *IEEE Trans. Circuit and Systems for Video Technology*, vol. 7, no. 2, pp. 429-433, Apr. 1997.

[12] B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," *IEEE Trans. Circuit and Systems for Video Technology*, vol. 3, no. 2, pp. 148-157, Apr. 1993.

[13] H. Fujiwara, M. L. Liou, M.-T. Sun, K.-M. Yang, M. Maruyama, K. Shomura, and K. Ohyama, "An all ASIC implementation of a low bit-rate video codec," *IEEE Trans. Circuit and Systems for Video Technology*, vol. 2, no. 2, pp. 123-134, June 1992.

[14] P. A. Ruetz, P. Tong, D. A. Luthi, and P. H. Ang, "A high-performance full-motion video compression chip set," *IEEE Trans. Circuit and Systems for Video Technology*, vol. 2, no. 2, pp. 111-122, June, 1992.

[15] W. Li and E. Salari, "Successive elimination algorithm for motion estimation," *IEEE Trans. Image Processing*, vol. 4, no. 1. pp. 105-107, Jan. 1995.

[16] 이법기, 이경환, 정원식, 최정현, 이건일, 김덕규, "인접 블록의 중첩된 탐색 영역을 고려한 고속 블록 정합 알고리즘," *대한전자공학회논문지*, 제 36권, S편, 제 5호, pp. 48-55, 1999년 5월.

[17] Y. L. Chan and W. C. Siu, "New adaptive pixel decimation for block motion vector estimation," IEEE Trans. Circuit and Systems for Video Technology, vol. 6, no. 1, pp. 113-118, Feb. 1996.

정 원 식(Won-Sik Cheong) 정회원
통신학회논문지 제24권 제1B호 참조

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

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

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

김 경 규(Kyoung-Kyoo Kim) 정회원
통신학회논문지 제24권 제1B호 참조

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

이 건 일(Kuhn-Il Lee) 정회원
통신학회논문지 제24권 제1B호 참조