論文2000-37SP-5-5

이웃 탐색점에서의 평균 절대치 오차를 이용한 2단계 고속 블록 정합 알고리듬

(A Two-Stage Fast Block Matching Algorithm Using Mean Absolute Error of Neighbor Search Point)

鄭 元 植 *, 李 法 基 *, 權 成 根 *, 韓 纘 豪 ***,
申 容 達 **, 宋 奎 翼 *, 李 健 一 *

(Won-Sik Cheong, Bub-Ki Lee, Seong-Geun Kwon, Chan-Ho Han, Yong-Dal Shin, Kyu-Ik Sohng, and Kuhn-Il Lee)

요 약

본 논문에서는 이웃 탐색점에서의 평균 절대치 오차 (mean absolute error, MAE)를 이용하여 전역 탐색 알고리듬 (full search algorithm, FSA)과 거의 같은 움직임 추정 성능을 얻으면서도 고속으로 움직임을 추정할 수 있는 2단계 고속 블록 정합 알고리듬을 제안하였다. 제안한 방법에서는 현재 탐색점에서 블록 정합을 통하여 얻을 수 있는 MAE의 최소 범위를 이웃 탐색점에서의 MAE를 이용하여 구한 뒤, 이를 이용하여 블록 정합이 필요한 탐색점에 대하여서만 블록 정합을 행하였다. 즉, 제안한 방법에서는 블록 정합이 필요한 탐색점 수를 줄임으로써 고속으로 움직임을 추정하였으며, 움직임 추정을 두 단계로 나누어 수행하였다. 모의 실험을 통하여 제안한 방법이 FSA와 거의 같은 움직임 추정 성능을 유지하면서도 많은 계산량의 감소를 얻을 수 있음을 확인하였다.

Abstract

In this paper, we propose a two-stage fast block matching algorithm using the mean absolute error (MAE) of neighbor search point that can reduce the computational complexity to estimate motion vector while the motion estimation error performance is nearly the same as full search algorithm (FSA). In the proposed method, the lower bound of MAE 6at current search point is calculated using the MAE of neighbor search point. And we reduce the computational complexity by performing the block matching process only at the search point that has to be block matched using the lower bound of MAE. The proposed algorithm is composed of two stages. The experimental results show that the proposed method drastically reduces the computational complexity while the motion compensated error performance is nearly kept same as that of FSA.

* 正會員, 慶北大學校 電子電氣工學部
(School of Electronic and Electrical Engineering, Kyungpook National University)
** 正會員, 永同大學校 電子工學部
(School of Electronic Engineering, Youngdong University)
*** 正會員, 慶雲大學校 멀티미디어情報學部
(Dept. of Software, School of Multimedia and Information, Kyungwoon University)
接受日字 : 1999年8月27日, 수정완료일 : 2000年4月3日

Ⅰ. 서 론

블록 정합 알고리듬 (block matching algorithm, BMA)을 이용한 움직임 추정 기법은 H.261,[1] H.263,[2] 및 MPEG[3,4] 등과 같은 많은 동영상 압축 시스템에서 사용되고 있으며, 시간적인 중복성을 제거하여 높은 압축율을 얻는데 핵심적인 역할을 담당하고 있다.

BMA를 이용한 움직임 추정 기법에서는 입력 영상을 임의의 작은 블록으로 나눈 뒤, 블록내의 모든 화소들은 같은 방향으로 평행 이동한다는 것을 가정하여 이전 프레임의 탐색영역에서 정합 척도가 최적인 블록을 찾는다. 이때, 정합 척도로는 평균 자승 오차 (mean squared error, MSE)와 비슷한 성능을 가지면서도 계산량이 적은 MAE가 많이 사용된다.

BMA를 이용하여 움직임을 추정하는 경우에 가장 좋은 움직임 추정 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 3 단계 탐색 (three step search, TSS), 중첩 3 단계 탐색 (overlapped TSS, OTSS)알고리듬, 및 교차 탐색 (cross search) 알고리듬 등이 있으며,[6-10, 13, 18] 이 방법들 중 간단하면서도 성능이 좋은 TSS 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 빠질 수 있는 단점을 가지며 특히, MPEG에서와 같이 탐색영역이 큰 경우에는 국부 최소에 빠질 확률이 매우 커져서[18] 움직임 추정의 정확성이 크게 떨어지게 된다.

Jong 등[18]은 3 단계 탐색 알고리듬을 큰 탐색영역에 적용할 때 발생하는 성능 저하를 개선하기 위하여 중첩 3 단계 탐색 알고리듬을 제안하였다. 이 알고리듬에서는 탐색영역을 네 개의 영역으로 나눈 뒤, 각 영역에 대하여 3 단계 탐색 알고리듬을 적용하고, 탐색영역의 중심에 대하여 3 단계 탐색 알고리듬을 한번 더 적용하였다. 그러나, 이 알고리듬은 MPEG-1 SM3에서 권고된 3 단계 탐색 알고리듬의 움직임 추정 성능을 어느 정도 개선시켰지만, 이의 움직임 추정 오차는 전역 탐색 알고리듬에 비하여 여전히 크다.

또한, Liu 등[11]은 전역 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.

본 논문에서는 이웃 탐색점에서의 MAE를 이용하여 블록 정합이 필요한 탐색점 수를 줄임으로써FSA와 거의 같은 움직임 추정 성능을 얻으면서도 고속으로 움직임 추정을 행할 수 있는 2단계 고속 블록 정합 알고리듬을 제안하였다. 제안한 방법에서는 현재 탐색점에서 블록 정합을 통하여 얻을 수 있는 MAE의 최소 범위를 이웃 탐색점에서의 MAE를 이용하여 구한 뒤, 이를 이용하여 블록 정합이 필요한 탐색점에 대하여서만 블록 정합을 행하였다. 이때, 현재 탐색점에서의 MAE의 최소 범위를 구하기 위해서는 이웃 탐색점에서의 MAE가 필요하며, 블록 정합의 수행 여부를 결정하기 위해서는 MAE의 최소 범위와 비교할 수 있는 기준 MAE가 필요하다. 그러므로, 제안한 방법에서는 첫 번째 단계에서 탐색 영역을 3×3 크기의 영역으로 겹치지 않게 나눈 뒤, 각 영역의 중심 탐색점에 대하여 블록 정합을 행하여 MAE를 구하고, 이들 중 가장 작은 MAE를 기준 MAE로 정한다. 또한, 두 번째 단계에서는 각 영역의 중심 탐색점에서의 MAE를 이용하여 나머지 탐색점에서의 MAE의 최소 범위를 구한 뒤, 최소 범위가 기준 MAE 보다 작은 탐색점에 대하여서만 블록 정합을 행한다. 즉, 제안한 방법에서는 블록 정합이 필요한 탐색점 수를 줄임으로써 고속으로 움직임을 추정하였다.

제안한 방법의 성능을 평가하기 위한 여러 가지 동영상에 대한 컴퓨터 모의 실험을 통하여, 제안한 방법이 FSA와 거의 같은 성능을 유지하면서도 많은 계산

량의 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 것으로서, 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 움직임 추정 성능을 얻을 수 있는 방법은 FSA로서, 이 방법은 이전 프레임에 설정된 탐색 영역의 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 w인 경우에 탐색영역에서 정합을 행하여야 할 탐색점수는 $(2w+1)^2$이다. 따라서 FSA는 움직임 추정 오차 측면에서 최적인 움직임 벡터를 추정할 수 있지만, 탐색점 수가 너무 많으므로 계산량이 많은 단점이 있다.

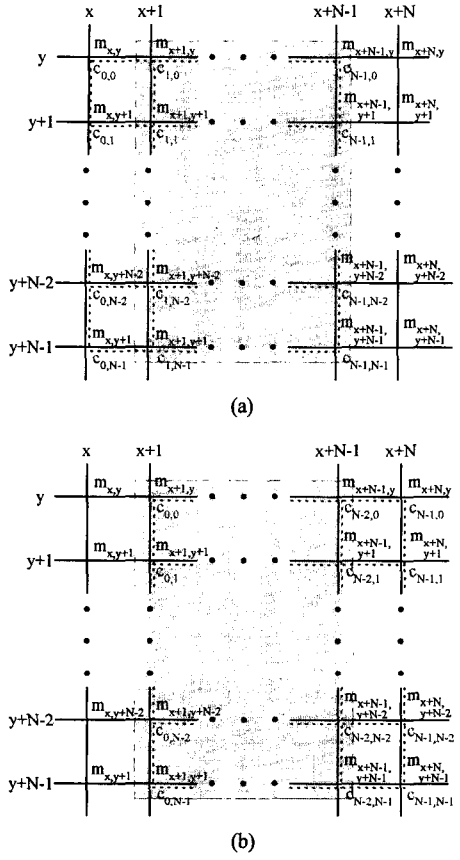
이와 같은 단점을 줄이기 위하여, 탐색점 수를 줄이는 방법과 블록의 화소들을 부표본화한 뒤 블록 정합을 행하는 방법 등이 연구되었다. 탐색영역에서 탐색점 수를 줄이는 방법의 경우에는 움직임 추정 오차는 움직임 방향으로 단조 감소한다는 가정을 이용하여 전역 탐색을 행하지 않고 탐색영역의 일부에 대해서만 탐색을 행하므로, 계산량은 크게 줄일 수 있었지만 국부 최소에 빠질 수 있는 단점을 가진다. 즉, 움직임 추정의 정확성을 어느 정도 희생함으로써 계산량의 감소를 얻기 때문에 움직임 추정 오차가 커지며, 특히 이 방법에서 사용된 가정이 잘 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,[16,17] FSA와 동일한 성능을 유지하면서도 움직임 추정에 필요한 계산량을 줄이기 위한 방법이 연구되고 있다.[14,15]

또한, Liu 등[11]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. 블록 MAE의 최소 및 최대 범위

본 논문에서는 이웃 탐색점에서의 MAE를 이용한 2단계 고속 블록 정합 알고리듬을 제안한다. 제안한 방법에서는 이웃 탐색점에서의 MAE를 이용하여 현재 탐색점에서의 MAE의 최소 범위를 구한 뒤, 이를 이용하여 블록 정합이 필요한 탐색점에 대하여서만 블록 정합을 행함으로써 고속으로 움직임을 추정한다. 본 장에서는 먼저, 현재 탐색점에서의 MAE를 이웃 탐색점에서의 블록 정합 결과로부터 구할 수 있음을 보인 뒤, 여기에 삼각 부등식 (triangular inequality) 개념을 적용하여 현재 탐색점에서의 MAE의 최소 및 최대 범위를 구하는 방법을 제시한다. 그리고, 다음 장에서 MAE

의 최소 범위를 이용한 2단계 고속 블록 정합 알고리듬을 제안한다.



+ : coordinate of search region.
·:· : coordinate of current block.
□ : overlapped search area with neighbor search point.

그림 1. (a) 탐색점 (x,y) 및 (b) 탐색점 $(x+1,y)$에서의 블록 정합

Fig. 1. Block matching at (a) search point (x,y) and (b) search point $(x+1,y)$.

1. 이웃 탐색점에서의 오차 블록을 이용한 현재 탐색점에서의 MAE 계산

그림 1에서와 같이, 수평 및 수직 방향의 크기가 N인 현재 블록 C와 탐색 영역의 (x,y) 위치에서의 후보 정합 블록 (candidate matching block) $\mathbf{M}(x,y)$의 화소들의 휘도 값을 $N\times N$ 행렬로 나타내면 각각

$$\mathbf{C} = [\ \mathbf{C}_0\ \cdots\ \mathbf{C}_{N-1}\ ] \qquad (3)$$
$$\textit{where},\ \mathbf{C}_k = [\ c_{0,k}\ \cdots\ c_{N-1,k}\ ]^T$$

$$\mathbf{M}(x,y) = [\ \mathbf{M}_0(x,y)\ \cdots\ \mathbf{M}_{N-1}(x,y)\ ]$$
$$\textit{where},\ \mathbf{M}_k(x,y) = [\ m_{y,x+k}\ \cdots\ m_{y+N-1,x+k}\ ]^T \qquad (4)$$

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

$$\mathbf{D}(x,y) = [\ \mathbf{M}_0(x,y) - \mathbf{C}_0\ \cdots\ \mathbf{M}_{N-1}(x,y) - \mathbf{C}_{N-1}\ ] \qquad (5)$$

와 같다. 그리고, 탐색점 (x,y)에서의 MAE는

$$\begin{aligned} MAE(x,y) &= \frac{1}{N^2}\|\ \mathbf{M}(x,y) - \mathbf{C}\ \| \\ &= \frac{1}{N^2}\|\ \mathbf{D}(x,y)\ \| \end{aligned} \qquad (6)$$

와 같이 구할 수 있다.

또한, 탐색점 (x,y)에서 블록 정합을 행하였다면, 이 탐색점에서의 오차 블록을 이용하여 이웃 탐색점의 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

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

와 같다. 여기서, 식 (4)에서 $\mathbf{M}_k(x,y)$의 정의로부터 $\mathbf{M}_k(x+1,y) = \mathbf{M}_{k+1}(x,y)$의 관계가 성립함을 알 수 있으며, 그림 1로부터 이를 확인할 수 있다. 또한, 수평 방향의 이웃하는 탐색점인 (x,y)와 $(x+1,y)$에서의 블록 정합 시에 사용되는 탐색 영역에서의 화소들은 그림 1에서 보는바와 같이 하나의 열을 제외한 나머지는 동일함을 알 수 있다. 그러므로, 탐색점 (x,y)에서의 오차 블록 $\mathbf{D}(x,y)$을 이용하여 탐색점 $(x+1,y)$에서의 오차 블록 $\mathbf{D}(x+1,y)$를 구할 수 있다. 이를 위하여 현재 블록 내의 화소들의 이웃 화소간의 수평 방향으로의 화소값의 차를

$$\begin{aligned} \mathbf{C}_{DH+} = [\ &\mathbf{C}_0 - \mathbf{C}_{N-1}\ \mathbf{C}_1 \\ &- \mathbf{C}_0 \cdots\ \mathbf{C}_{N-1} - \mathbf{C}_{N-2}] \end{aligned} \qquad (8)$$

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

$$\mathbf{D}(x,y) + \mathbf{C}_{DH+} = [\ \mathbf{M}_0(x,y) - \mathbf{C}_{N-1}\ \mathbf{M}_1(x,y) - \mathbf{C}_0 \cdots \mathbf{M}_{N-1}(x,y) - \mathbf{C}_{N-2}] \tag{9}$$

와 같이 나타낼 수 있으며, 이 식을 식 (7)과 비교해 보면, 식 (7)의 마지막 열을 제외한 나머지 부분과 이 식의 첫 번째 열을 제외한 나머지 부분이 동일함을 알 수 있다. 이를 그림으로 알아보면, 그림 1에서 표시되어 있는바와 같이 탐색점 (x,y) 및 탐색점 $(x+1,y)$에서의 블록 정합에 회색영역으로 표시된 이전 프레임의 화소들이 공통으로 사용되고 있고, 탐색점 (x,y)에서의 수평 좌표 x에 해당하는 부분과 탐색점 $(x+1,y)$에서의 수평 좌표 $x+N$에 해당하는 부분이 서로 다름을 알 수 있다. 이러한 차이로 인하여 식 (9)와 식 (7)의 차이가 발생된다. 그러므로 탐색점 (x,y)에서의 오차 블록 $\mathbf{D}(x,y)$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 $\hat{\mathbf{D}}t_{H+}(x,y)$를

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

와 같이 구한 뒤, 이를 $\mathbf{C}_{DH+}$와 더하면

$$\hat{\mathbf{D}}_{H+}(x,y) + \mathbf{C}_{DH+} = [\ \mathbf{M}_N(x,y) - \mathbf{C}_{N-1}\ \mathbf{M}_1(x,y) - \mathbf{C}_0 \cdots \mathbf{M}_{N-1}(x,y) - \mathbf{C}_{N-2}] \tag{11}$$

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

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

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 및 최대 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 $\mathbf{C}_{DH+}$를 구하는 과정과 $\widehat{\mathbf{D}_{H+}}(x,y)$의 첫 번째 열을 대치시키는 과정으로 인하여 계산량의 이득을 얻을 수 없다. 그러므로 본 논문에서는 삼각 부등식을 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

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

와 같이 정의된다.[14] 그리고, 삼각 부등식은

$$|\ \|\ \mathbf{A}_1\ \| - \|\ \mathbf{A}_2\ \|\ | \le \|\ \mathbf{A}_1 + \mathbf{A}_2\ \| \le \|\ \mathbf{A}_1\ \| + \|\ \mathbf{A}_2\ \| \tag{14}$$

와 같고, 이를 식 (12)에 적용하면

$$|\ \|\ \hat{\mathbf{D}}_{H+}(x,y)\ \| - \|\ \mathbf{C}_{DH+}\ \|\ | \le \|\ \hat{\mathbf{D}}_{H+}(x,y) + \mathbf{C}_{DH+}\ \| \le \|\ \hat{\mathbf{D}}_{H+}(x,y)\ \| + \|\ \mathbf{C}_{DH+}\ \| \tag{15}$$

와 같다. 또한 식 (12)로부터 $\|\ \hat{\mathbf{D}}_{H+}(x,y) + \mathbf{C}_{DH+}\ \| = \|\ \mathbf{M}(x+1,y) - \mathbf{C}\ \|$이므로 이를 식 (15)에 대입한 뒤, 식의 양변을 N^2으로 나누면 식 (15)는

$$|\ \frac{1}{N^2}\|\ \hat{\mathbf{D}}_{H+}(x,y)\ \| - \frac{1}{N^2}\|\ \mathbf{C}_{DH+}\ \|\ | \le MAE(x+1,y) \le \frac{1}{N^2}\|\ \hat{\mathbf{D}}_{H+}(x,y)\ \| + \frac{1}{N^2}\|\ \mathbf{C}_{DH+}\ \| \tag{16}$$

와 같다. 이 식을 살펴보면 탐색점 $(x+1,y)$에서의 MAE의 최소 및 최대 값을 현재 블록 내의 화소들의 이웃 화소간의 화소값의 차 $\mathbf{C}_{DH+}$와 탐색점 (x,y)에서의 오차 블록 $\mathbf{D}(x,y)$의 첫 번째 열을 대치한 블록

$\widehat{\mathbf{D}_{H+}}(x,y)$ 로부터 구할 수 있음을 알 수 있다. 이때 식 (16)에서 $\| \mathbf{C}_{DH+} \|$는 전체 탐색 영역에 대하여 한 번만 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.

Ⅳ. 제안한 2단계 고속 블록 정합 알고리듬

앞 장에서 살펴본 바와 같이, 현재 탐색점에서의 MAE의 최소 및 최대 범위는 이웃 탐색점에서의 MAE를 이용하여 식 (16)에서와 같이 구할 수 있다. 이를 이용하여 블록 정합이 필요한 탐색점에 대하여서만 블록 정합을 행한다면, 움직임 추정 오차 측면에서 최적인 FSA와 동일한 움직임 추정 성능을 얻으면서도 움직임 추정에 필요한 계산량을 줄일 수 있다.[17] 그러나, 식 (16)의 최소 및 최대 범위를 구하기 위해서는 식 (5)로 표현되는 이웃 탐색점에서의 오차 블록을 사용하지 않고, 식 (10)에서 정의된 첫 번째 열을 대치시킨 새로운 오차 블록을 구하여야하므로 이에 따른 부가 계산량이 필요하며, 이에 따라 움직임 추정 알고리듬이 복잡해 질 수 있다. 그러므로, 본 논문에서는 부가 계산량을 없애고, 알고리듬을 간단히 하기 위하여, 식 (5)에서 정의된 이웃 탐색점에서의 오차 블록 $\mathbf{D}(x,y)$를 사용하여 FSA와 거의 같은 움직임 추정 성능을 가지면서도, 움직임 추정에 필요한 계산량을 줄일 수 있는 고속 블록 정합 알고리듬을 제안한다.

본 장에서는 MAE의 최소 범위를 구할 때, 식 (10)에서 정의된 새로운 오차 블록 $\hat{\mathbf{D}}_{H+}(x,y)$를 사용하지 않고 $\mathbf{D}(x,y)$를 사용 할 경우에 발생할 수 있는 오차에 대한 해석을 한 뒤, 이를 이용한 2단계 고속 블록 정합 알고리듬을 제안한다.

1. 이웃 탐색점에서의 MAE를 이용한 현재 탐색점에서의 MAE의 최소 범위 계산

식 (16)에서와 같이 현재 탐색점에서의 MAE의 최소 범위를 이용하여 FSA와 동일한 움직임 추정 성능을 유지하면서 계산량을 줄이기 위해서는 이웃 탐색점 (x,y)에서의 MAE를 사용하지 않고, 식 (10)에서 정의된 새로운 오차 블록 $\hat{\mathbf{D}}_{H+}(x,y)$를 이용하여야 한다. 또한, 우리가 이웃 탐색점 (x,y)에서 블록 정합을 통하여 구하는 것은 오차 블록 $\mathbf{D}(x,y)$가 아니라, 탐색점 (x,y)에서의 MAE인 $\| \mathbf{D}(x,y) \|$이다. 그러므로, 이로부터 식 (16)의 최소 범위를 구하는데 필요한 $\| \hat{\mathbf{D}}_{H+}(x,y) \|$를 구하기 위해서는 $\mathbf{D}(x,y)$의 첫 번째 열을 대치시키는 과정이 아니라, 미리 계산되어 진 $\| \mathbf{D}(x,y) \|$의 첫 번째 열에 해당하는 양을 뺀 후, 대치시키는 열에 해당하는 양을 더하는 과정이 필요하다. 즉,

$$\begin{aligned} \| \hat{\mathbf{D}}_{H+}(x,y) \| &= \| \mathbf{D}(x,y) \| \\ &\quad - \| \mathbf{M}_0(x,y) - \mathbf{C}_0 \| \\ &\quad + \| \mathbf{M}_N(x,y) - \mathbf{C}_0 \| \end{aligned}$$

$$\textit{where,} \quad \| \mathbf{M}_0(x,y) - \mathbf{C}_0 \| = \sum_{i=0}^{N-1} | m_{y+i,x} - c_{i,0} |$$

$$\| \mathbf{M}_N(x,y) - \mathbf{C}_0 \| = \sum_{i=0}^{N-1} | m_{y+i,x+N} - c_{i,0} | \tag{17}$$

와 같이 $\| \mathbf{D}(x,y) \|$로부터 $\| \hat{\mathbf{D}}_{H+}(x,y) \|$를 구할 수 있다. 이를 위해서는 $\| \mathbf{M}_0(x,y) - \mathbf{C}_0 \|$를 구하기 위한 계산량과 $\| \mathbf{M}_N(x,y) - \mathbf{C}_0 \|$를 구하기 위한 계산량이 부가적으로 필요하다. 그러므로, 본 논문에서는 $\| \hat{\mathbf{D}}_{H+}(x,y) \|$를 사용하지 않고, $\| \mathbf{D}(x,y) \|$를 이용하여 현재 탐색점에서의 MAE의 최소 범위의 근사 값을 구한 뒤, 이를 이용하여 고속으로 움직임을 추정한다.

2. 블록 MAE의 근사 최소 범위의 오차

수평 방향의 이웃 탐색점에서의 오차 블록을 이용하여 구한 현재 탐색점에서의 MAE의 실제 최소 범위는 식 (16)에서 구한 것과 같이

$$LB_{act}(x+1,y) = | \frac{1}{N^2} \| \hat{\mathbf{D}}_{H+}(x,y) \| - \frac{1}{N^2} \| \mathbf{C}_{DH+} \| | \tag{18}$$

와 같이 나타낼 수 있다. 그러나 이를 구하기 위해서는 식 (17)과 같은 과정이 필요하며, 이에 따른 부가 계산량이 필요하다. 즉, 탐색점 (x,y)에 대하여 블록 정합

을 행하여 MAE를 구하였다면, 탐색점 (x,y)의 주위의 8개 탐색점에 대하여 MAE의 최소 범위를 구할 수 있다. 이때, 수평 방향의 이웃 탐색점인 $(x-1,y)$와 $(x+1,y)$에서의 MAE의 최소 범위를 구하는데 사용되는 하나의 열이 대치된 오차 블록을 구하기 위해서는 식 (17)에서와 같이 하나의 열을 대치시켜야 하고, 수직 방향의 이웃 탐색점인 $(x,y-1)$와 $(x,y+1)$에 대하여서는 하나의 행을 대치시키는 과정이 필요하며, 대각 방향의 이웃 탐색점 $(x-1,y-1)$, $(x+1,y-1)$, $(x-1,y+1)$, 및 $(x+1,y+1)$에 대하여서는 하나의 열과 행을 대치시키기 위한 부가 계산량이 필요하다. 그러므로, 본 논문에서는 이러한 부가 계산량을 줄이고, 알고리듬을 간소화 시키기 위하여 MAE의 최소 범위의 근사 값, 즉 MAE의 근사 최소 범위를

$$LB_{app}(x+1,y) = |\frac{1}{N^2}\| \mathbf{D}(x,y)\| - \frac{1}{N^2}\| \mathbf{C}_{DH+}\||| \tag{19}$$

와 같이 구하여 사용한다. 이 식을 살펴보면, 식 (18)에서 사용된 $\| \hat{\mathbf{D}}_{H+}(x,y)\|$ 대신 $\| \mathbf{D}(x,y)\|$를 사용하여 $LB_{app}(x+1,y)$를 구한다. 그러므로, 식 (17)에서와 같이 $\| \hat{\mathbf{D}}_{H+}(x,y)\|$를 구하기 위해서 필요한 부가 계산량이 필요 없다. 그러나, 이와 같이 실제 MAE의 최소 범위인 $LB_{act}(x+1,y)$를 사용하지 않고, 이의 근사 값인 $LB_{app}(x+1,y)$를 사용하기 때문에 이로 인한 오차가 발생하게 된다.

이러한 MAE의 최소 범위의 오차는 움직임 추정에 사용되는 블록의 크기가 $N\times N$라 하면, 수평 및 수직 방향의 이웃 탐색점의 경우에는 MAE의 최소 범위는 $-N$에서 N사이의 오차를 가질 수 있다. 그리고, 대각 방향의 이웃 탐색점의 경우에는 행과 열이 하나의 화소에서 겹치기 때문에 $-(2N-1)$에서 $2N-1$사이의 오차를 가질 수 있다. 이러한 오차의 최대치는 식 (17)에서 알 수 있는바와 같이, $\mathbf{M}_0(x,y) - \mathbf{C}_0$의 모든 원소가 0의 값을 가지고, $\mathbf{M}_N(x,y) - \mathbf{C}_0$의 모든 원소가 최대 값을 가지는 경우에 발생할 수 있다. 그러나, 이러한 경우는 실제 영상에서 거의 발생하지 않는다. 실제 영상에서의 MAE의 최소 범위의 오차를 알아보기 위하여, 720×480 크기의 FOOTBALL, FLOWER GARDEN, MOBILE, 및 SUSIE 영상 각 40 프레임에 대하여 구한 MAE 최소 범위의 오차의 확률 분포를 그림 2에 나타내었다. 여기서, 오차는 $LB_{act}(x+m,y+n)-LB_{app}(x+m,y+n)$을 나타낸다. 이때, m과 n은 -1과 1사이의 값을 가질 수 있으며, $LB_{act}(x\pm1,y)-LB_{app}(x\pm1,y)$는 수평, $LB_{act}(x,y\pm1)-LB_{app}(x,y\pm1)$는 수직, $LB_{act}(x\pm1,y\pm1)-LB_{app}(x\pm1,y\pm1)$는 대각 방향을 나타내고, 복호의 순서는 같다. 이 그림을 살펴보면, MAE의 최소 범위를 구하는데 $\| \mathbf{D}(x,y)\|$를 사용하더라도 식 (17)

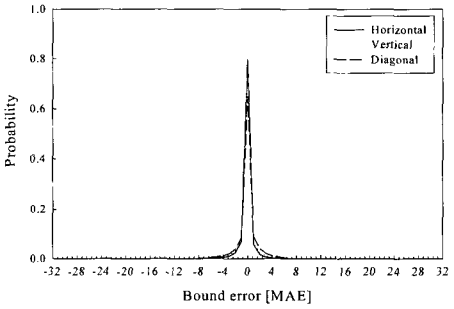


그림 2. MAE의 최소 범위 오차의 확률 분포
Fig. 2. Probability distribution of MAE lower bound error.

표 1. MAE의 최소 범위 오차의 확률 분포
Table 1. Probability distribution for error of lower bound of MAE.

Bound Error		-3	-2	-1	0	1	2	3
Probability distribution	Horizontal	0.0103	0.0241	0.0689	0.7977	0.0549	0.0160	0.0060
	Vertical	0.0157	0.0306	0.0711	0.7632	0.0605	0.0233	0.0112
	Diagonal	0.0211	0.0384	0.0826	0.6463	0.0914	0.0392	0.0205
Cumulative distribution	Horizontal	0.0252	0.0493	0.1182	0.9160	0.9708	0.9868	0.9928
	Vertical	0.0300	0.0606	0.1317	0.8949	0.9554	0.9787	0.9899
	Diagonal	0.0524	0.0908	0.1734	0.8197	0.9111	0.9503	0.9708

과 같이 행 및 열을 대치시키는 과정을 통하여 구한 실제 최소 범위와 같을 확률이 매우 높음을 알 수 있으며, 오차가 커짐에 따라 오차가 발생할 확률이 급격히 떨어짐을 알 수 있다. 이는 오차가 발생하더라도 작은 값을 가짐을 나타낸다.

이를 자세히 살펴보기 위하여 오차의 대부분이 발생하는 -3과 3사이의 확률 분포를 표 I에 나타내었다.

이 표로부터 실제 MAE의 최소 범위와 근사 값이 같을 확률이 수평 및 수직 방향의 이웃 블록의 경우에는 약 80%와 76%로 매우 높음을 알 수 있으며, 대각 방향의 이웃 블록의 경우에는 수평 및 수직 방향의 이웃 블록의 경우보다는 낮지만 약 65%의 높은 확률을 가짐을 알 수 있다. 그리고, -3에서 3사이의 작은 오차를 가질 확률이 약 97%에서 92%로서 오차가 발생하더라도 작은 값을 가짐을 확인할 수 있다. 또한, 최소 범위의 오차가 움직임 추정 성능에 영향을 미치는 경우는 현재 탐색점에서의 MAE값이 탐색 영역 내에서 최소 값을 가짐에도 불구하고, 최소 범위의 오차로 인하여 블록 정합을 행하지 않는 경우이다. 그러나, 식 (15)와 식 (16)을 살펴보면, 식 (16)에서 최소 범위는 $\hat{\mathbf{D}}_{H+}(x,y)$와 $\mathbf{C}_{\mathbf{DH}+}$의 부호가 모두 같은 경우의 $\frac{1}{N^2}\| \hat{\mathbf{D}}_{H+}(x,y) + \mathbf{C}_{\mathbf{DH}+} \|$와 같고, 최대 범위는 $\hat{\mathbf{D}}_{H+}(x,y)$와 $\mathbf{C}_{\mathbf{DH}+}$의 부호가 모두 다른 경우의 $\frac{1}{N^2}\| \hat{\mathbf{D}}_{H+}(x,y) + \mathbf{C}_{\mathbf{DH}+} \|$와 같음을 알 수 있다. 여기서, $\frac{1}{N^2}\| \hat{\mathbf{D}}_{H+}(x,y) + \mathbf{C}_{\mathbf{DH}+} \|$는 식 (15)와 식 (16)으로부터 탐색점 $(x+1,y)$에서의 MAE인 $MAE(x+1,y)$이다. 또한, $\hat{\mathbf{D}}_{H+}(x,y)$와 $\mathbf{C}_{\mathbf{DH}+}$의 부호가 모두 같거나 모두 반대인 경우는 매우 적고, 부호가 혼재하는 경우가 많기 때문에 실제 MAE는 MAE의 최소 범위와 최대 범위 사이에 존재하여, MAE의 최소 범위와 실제 MAE는 어느 정도의 차이가 있음을 유추할 수 있다. 그런데, 앞에서 살펴본 바와 같이 MAE의 최소 범위의 오차는 -3에서 3사이의 작은 값에 집중되어 나타나므로, 최소 범위의 오차로 인하여 현재 탐색점에서의 MAE값이 탐색 영역 내에서 최소 값을 가짐에도 불구하고, 블록 정합을 행하지 않기 때문에 발생하는 움직임 추정 성능의 저하는 거의 없을 것으로 예상할 수 있다. 그러므로 본 논문에서는, 실제 최소 범위 $LB_{act}(x+1,y)$를 사용하지 않고, 최소 범위의 근사 값 $LB_{apb}(x+1,y)$를 사용하여 고속으로 움직임을 추정한다.

3. 2단계 고속 블록 정합 알고리듬

본 논문에서는 현재 탐색점에서의 MAE의 근사 최소 범위를 이웃 탐색점에서의 MAE를 이용하여 구한 뒤, 이를 이용하여 블록 정합이 필요한 탐색점의 수를 줄임으로써 고속으로 움직임을 추정할 수 있는 방법을 제안한다. 제안한 방법에서는 현재 탐색점에서의 MAE의 근사 최소 범위를 구하기 위하여 이웃 탐색점에서의 MAE가 필요하고, 블록 정합의 수행 여부를 결정하기 위하여 MAE의 최소 범위와 비교하기 위한 기준 MAE가 필요하다. 그러므로 제안한 방법에서는 움직임 추정 과정을 2단계로 나누어 수행한다.

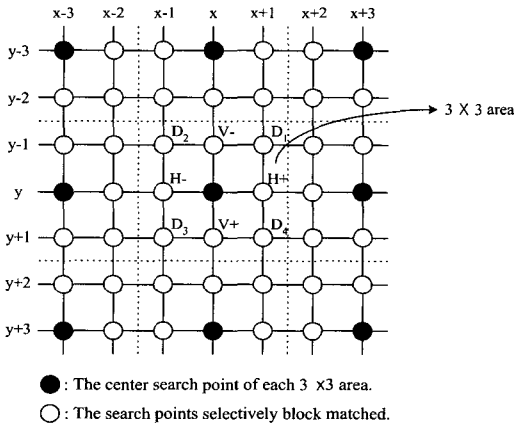


그림 3. 제안한 방법에서 사용되는 탐색점의 구조
Fig. 3. The search point structure for proposed method.

1) 1단계 - 각 영역의 중심 탐색점에서의 MAE 및 기준 MAE 계산

제안한 방법에서는 이웃 탐색점에서의 MAE를 이용하여 현재 탐색점에서의 MAE의 근사 최소 범위를 구한다. 이때, 하나의 탐색점에 대하여 블록 정합을 행하여 MAE를 구하였다면, 주위 8개 탐색점에서의 MAE의 근사 최소 범위를 구할 수 있다. 그러므로, 제안한 방법의 첫 번째 단계에서는 이전 프레임에 설정된 탐색 영역을 그림 3에서와 같이 3×3 크기의 영역으로 겹치지 않게 나눈 뒤, 그림에서 검은 점으로 표시된 각 영역의 중심 탐색점에 대하여 블록 정합을 행하여 MAE를 구한다. 이렇게 구한 각 영역의 중심 탐색점에서의 MAE는 2단계에서 주위 8개 탐색점에 대한 근사

최소 범위를 구할 때 사용된다.

또한, 각 탐색점에서의 MAE의 근사 최소 범위를 이용하여 블록 정합의 수행 여부를 결정하기 위해서는 MAE의 근사 최소 범위와 비교할 수 있는 기준 MAE가 필요하다. 그러므로, 두 번째 단계에서 블록 정합을 행할 것인지를 결정하기 위하여 사용되는 기준 평균 절대치 오차 MAE_{ref}를

$$MAE_{ref} = \min MAE(x, y) \tag{20}$$

where, (x, y): The center search point of each 3×3 area.

와 같이 각 3×3 영역의 중심 탐색점에서 구한 MAE들 중 최소 MAE로 정한다.

2) 2단계 - 근사 MAE의 최소 범위를 이용한 탐색점 줄임

제안한 방법의 두 번째 단계에서는 첫 번째 단계에서 각 3×3 영역의 중심 탐색점에서 블록 정합을 통하여 구한 $MAE(x, y)$, 즉 $\frac{1}{N^2}\| \mathbf{D}(x, y) \|$를 구한 뒤, (x, y) 주위의 8개 이웃 탐색점 $(x\pm1, y)$, $(x, y\pm1)$ 및 $(x\pm1, y\pm1)$에서의 MAE의 근사 최소 범위 $|\frac{1}{N^2}\| \mathbf{D}(x, y) \| - \frac{1}{N^2}\| \mathbf{C}_{Dk} \| |$를 구한다. 여기서, k는 그림 3에 나타낸 것과 같이 주위 8개의 탐색점들 각각의 위치를 나타낸다. 제안한 방법에서는 이렇게 구한 MAE의 근사 최소 범위를 이용하여

$$IF\ (\ MAE_{ref} < |\frac{1}{N^2}\| \mathbf{D}(x, y) \| - \frac{1}{N^2}\| \mathbf{C}_{Dk} \| |\ )$$
$$No\ block\ matching \tag{21}$$

$ELSE$ {
$Block\ matching$
$IF(MAE(i, j) < MAE_{ref.})$
$MAE_{ref} = MAE(i, j)$
}

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

와 같이 MAE의 근사 최소 범위가 기준 MAE인 MAE_{ref}보다 큰 탐색점에 대하여서만 블록 정합을 행한다. 즉, 제안한 방법에서는 첫 번째 단계에서 블록 정합을 행하였던 각 3×3 영역의 중심 탐색점을 제외한 나머지 탐색점에 대하여 그 탐색점에서 가질 수 있는 MAE의 근사 최소 범위를 이웃 탐색점에서의 MAE를 이용하여 구한 뒤, 이 값이 기준 평균 절대치 오차 MAE_{ref}보다 큰 경우에는 블록 정합을 수행하지 않는다. 또한, MAE의 근사 최소 범위가 MAE_{ref}보다 작은 탐색점의 경우에는 이 탐색점에서의 MAE가 MAE_{ref}보다 작을 가능성이 있으므로, 이 탐색점에 대하여서는 블록 정합을 행하여 MAE를 구한 뒤 MAE_{ref}와 비교한다. 이때, 이 탐색점에서의 MAE인 $MAE(i, j)$가 기준 MAE인 MAE_{ref}보다 작으면 MAE_{ref}는 $MAE(i, j)$로 갱신되고, 이것의 최종값을 지닌 탐색점이 최종 움직임 벡터로 결정된다. 이와 같이 제안한 방법에서는 현재 블록의 움직임 탐색 영역의 모든 탐색점에 대하여 탐색을 행하지 않고, 적은 탐색점에 대하여서만 블록 정합을 행하면서도, 움직임 추정 오차 측면에서 최적인 FSA 기법과 거의 같은 성능을 얻을 수 있었다.

4. 제안한 기법의 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

$$C_{FSA} = 2T(2w+1)^2N^2 \tag{22}$$

와 같다.

반면, 제안한 방법의 계산량은 식 (20)의 계산량과 식 (21)의 계산량의 합으로 볼 수 있다. 먼저 식 (20)의 계산량을 살펴보면, 이 식을 이용하여 전체 탐색점의 1/9에 해당하는 탐색점에 대하여 블록 정합을 행하여야 하므로 FSA의 계산량의 1/9에 해당하는 $2T(2w+1)^2N^2/9$의 계산량이 필요하다. 식 (21)의 계산에서는 전체 탐색 영역의 8/9에 해당하는 탐색점에 대하여 MAE의 근사 최소 범위를 구하기 위한 계산량과 MAE의 근사 최소 범위가 MAE_{ref}보다 작은 탐색점에서의 블록 정합을 위한 계산량이 필요하다. 여기서, MAE의 최소 범위를 구하기 위해서는 수평, 수직 및 대각선 방향에 대응되는 $\| \mathbf{C}_{Dk} \|$를 구하여야 한다. $\| \mathbf{C}_{Dk} \|$의 계산을

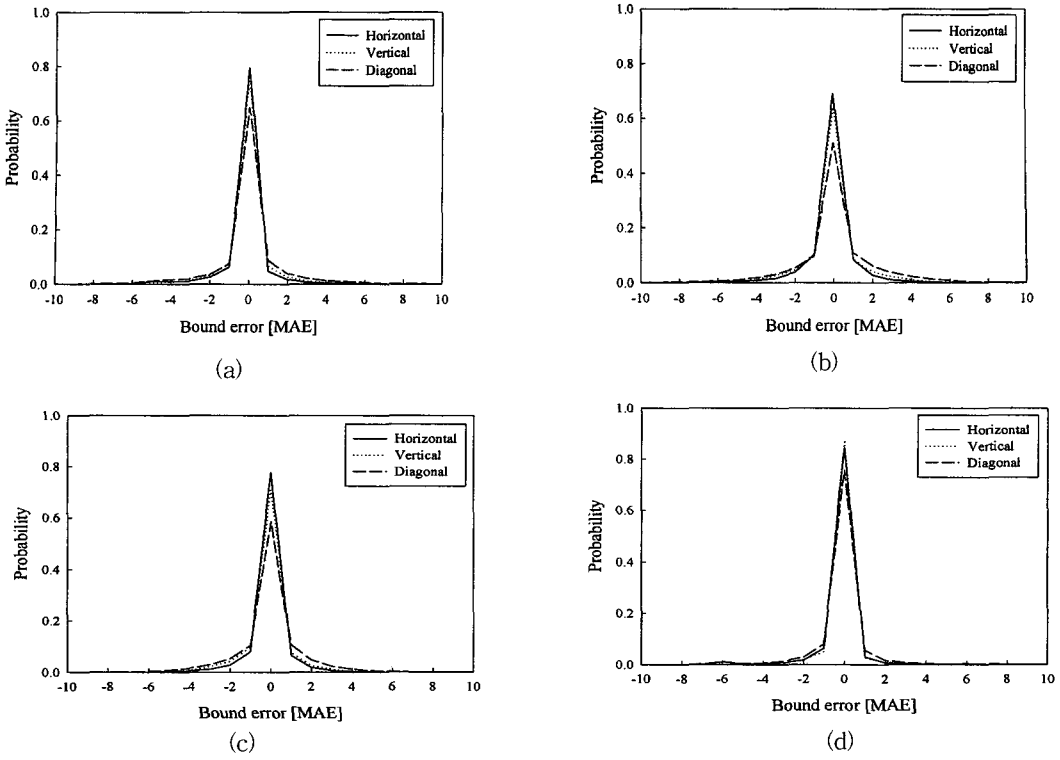


그림 4. (a) FOOTBALL, (b) FLOWER GARDEN, (c) MOBILE, 및 (d) SUSIE 영상에 대한 MAE의 근사최소 범위 오차의 확률 분포

Fig. 4. The probability distribution of MAE lower bound error for (a) FOOTBALL, (b) FLOWER GARDEN, (c) MOBILE, and (d) SUSIE sequences.

위해서는 전체 탐색영역에 대하여 각 방향에 $2N^2$번의 덧셈과 뺄셈의 계산이 필요하지만, 탐색점의 위치가 각 3×3 영역의 중심점에 대하여 180° 의 차이를 갖는 경우에 대하여서는 동일한 값을 갖는다. 즉, $\| C_{DH+} \|$와 $\| C_{DH-} \|$는 동일한 값이다. 그러므로 $\| C_{Dk} \|$를 구하기 위한 계산량은 한 블록 전체에 대하여 $8N^2$번의 덧셈과 뺄셈의 계산이 필요하다. 또한, MAE의 최소 범위가 MAE_{ref}보다 작은 탐색점에 대하여 블록 정합을 행하기 위한 계산량은 탐색점당 $2N^2$이다. 그러므로 제안한 방법의 계산량은

$$C_{proposed} = \frac{2}{9} TN^2(2w+1)^2 + 2PN^2 + 8TN^2 \qquad (23)$$

와 같다. 여기서 P는 식 (21)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.

V. 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.[12] 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 움직임

이 거의 없는 배경과 물체의 움직임이 빠른 영역이 혼재 하는 영상이며, SUSIE 영상은 비교적 단순한 영상으로 큰 움직임이 없는 영상이다. 그리고 정합 척도로는 MAE를 사용하였다. 또한 본 실험에서는 제안한 MAE의 근사 최소 범위의 오차에 대한 분석 결과를 제시하고, FSA와 제안한 2단계 고속 블록 정합 알고리듬에 대한 결과를 계산량 및 움직임 보상된 영상과 원 영상간의 PSNR을 이용하여 평가하였다.

1. MAE의 근사 최소 범위로 인한 오차 분석

제안한 방법에서는 현재 탐색점의 MAE의 근사 최소 범위를 이용하여 블록 정합이 필요한 탐색점 수를 줄임으로써 고속으로 움직임을 추정한다. 이때, 현재 탐색점의 MAE의 최소 범위를 사용하지 않고, 근사 값을 사용하기 때문에 이에 의한 오차가 발생할 수 있다. 그러므로 본 절에서는 MAE의 근사 최소 범위의 사용으로 인한 오차와 이 오차가 움직임 추정 성능에 미치는 영향에 대한 실험 결과를 제시한다.

1) 최소 범위의 오차

본 실험에서 사용된 실험 영상에 대한 MAE의 근사 최소 범위의 오차를 그림 4에 나타내었다. 이 그림에서는 MAE의 근사 최소 범위의 오차 $LB_{act}(x+m, y+n) - LB_{app}(x+m, y+n)$의 대부분이 0 근처에 집중되어 나타나므로, -10에서 10 사이에 대하여서만 나타내었다. 이 그림을 살펴보면, 실험에 사용된 영상에 관계없이 그림 2에서 제시한 확률 분포와 마찬가지로 오차가 발생하지 않을 확률이 매우 높고, 오차가 발생하더라도 작은 값을 가질 확률이 높음을 알 수 있다. 또한, MAE의 최소 범위의 오차는 움직임 추정 성능에 영향을 미칠 수는 있으나, 오차의 발생이 모두 움직임 추정 성능에 영향을 미치는 것은 아니다. 즉, 최소 범위의 오차와 기준 MAE의 관계에 따라 최소 범위의 오차가 움직임 추정 성능에 영향을 미칠 수도 있고, 영향이 없을 수도 있다. 이러한 MAE의 최소 범위의 오차가 움직임 추정 성능에 미치는 영향에 대하여 살펴보기 위하여 오차의 종류를 MAE의 근사 최소 범위 LB_{app}가 실제 최소 범위 LB_{act}에 비하여 작아지는 경우와 LB_{app}가 LB_{act}에 비하여 커지는 경우로 나누어 생각한다. 첫 번째로, 최소 범위의 오차로 인하여 LB_{app}가 LB_{act}보다 작아지는 경우는 그림 5의 (a), (b) 및 (c)에서와 같은 세 가지 경우가 있을 수 있다. 이들 중 움직임 추정 성능에 영향을 줄 수 있는 경우는 그림 5의 (c)에서와 같이, 실제 MAE의 최소 범위 LB_{act}는 기준 MAE MAE_{ref} 보

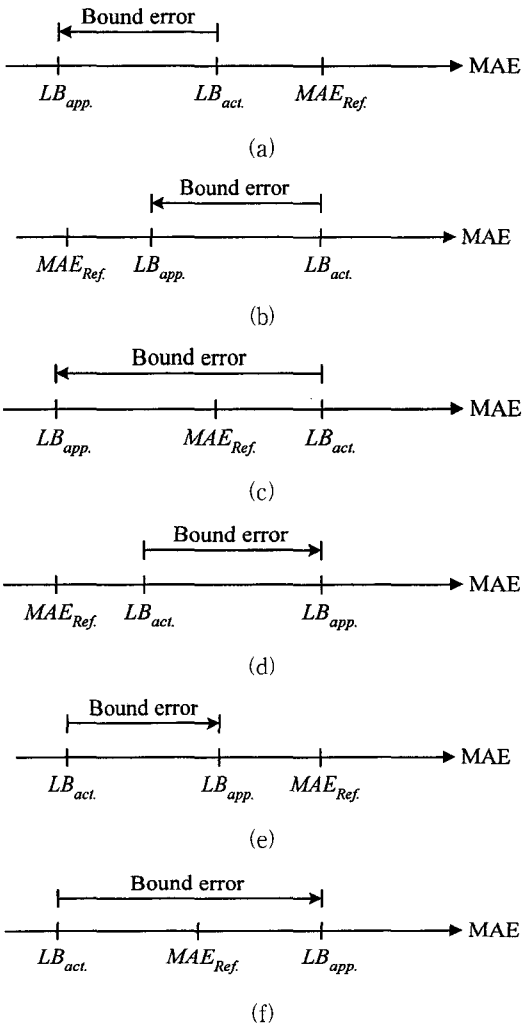


그림 5. LB_{app}가 LB_{act}보다 작아졌을 때, (a), (b) 최소 범위의 오차가 움직임 추정 성능에 영향을 미치지 않는 경우와 (c) 영향을 주는 경우 및 LB_{app}가 LB_{act}보다 커졌을 때 (d), (e) 최소 범위의 오차가 움직임 추정 성능에 영향을 미치지 않는 경우와 (f) 영향을 미치는 경우

Fig. 5. (a), (b) The case that lower bound error would not affet the motion estimation performance and (c) affect the motion estmation performance when the LB_{app} become smaller than LB_{act} and (d), (e) the case that lower bound error would not after the motion estimation performance and (f) affect the motion estimation performance when the LB_{app} become larger than LB_{act}.

표 2. MAE의 최소 범위의 오차에 의한 블록 정합이 필요한 탐색점의 증가
Table 2. The increase of search point which needs block matching process due to lower bound error.

	Horizontal	Vertical	Diagonal	Total
Increase of search points	0.091%	0.141%	0.629%	0.861%
Decrease of search points	0.044%	0.094%	0.547%	0.685%
Total increase of search points	0.047%	0.047%	0.082%	0.176%

표 3. 최소 범위 오차로 인하여 움직임 벡터를 잘못 찾을 확률
Table 3. The probability of finding wrong motion vector due to lower bound error.

	FOOTBALL	FLOWER GARDEN	MOBILE	SUSIE	Average
Probability of finding wrong motion vector	0.44%	1.48%	1.21%	3.32%	1.61%

표 4. 움직임 벡터를 잘못 찾은 경우의 최적의 움직임 벡터와의 MAE 차의 분포
Table 4. The distribution of difference of MAE between the wrong and optimal motion vector.

Difference of MAE	FOOTBALL	FLOWER GARDEN	MOBILE	SUSIE	Average
1	83.97%	61.20%	76.34%	84.06%	77.36%
2	12.24%	23.03%	18.78%	13.21%	16.44%
3	2.95%	9.64%	4.27%	2.29%	4.39%
4	0.84%	3.88%	0.61%	0.45%	1.29%
5	0.00%	1.50%	0.00%	0.00%	0.34%
6	0.00%	0.63%	0.00%	0.00%	0.14%
7	0.00%	0.13%	0.00%	0.00%	0.03%
8	0.00%	0.00%	0.00%	0.00%	0.00%

다 컸지만, 최소 범위의 오차로 인하여 근사 최소 범위 LB_{app}가 MAE_{ref}보다 작아지는 경우이다. 이 경우에는 LB_{act}를 이용하여 블록 정합 여부를 결정한다면 블록 정합을 행하지 않지만, LB_{app}를 이용하기 때문에 블록 정합을 행하게 되므로 블록 정합을 행하는 탐색점의 수가 증가하여 계산량의 증가가 발생한다. 두 번째로, 최소 범위의 오차로 인하여 LB_{app}가 LB_{act}보다 커지는 경우는 그림 5의 (d), (e) 및 (f)에서와 같은 세 가지 경우가 있을 수 있다. 이들 중 움직임 추정 성능에 영향을 줄 수 있는 경우는 그림 5의 (f)에서와 같이 LB_{act}는 MAE_{ref} 보다 작았지만, LB_{app}가 MAE_{ref} 보다 커지는 경우이다. 이 경우에는 LB_{act}를 이용하면 블록 정합을 행하여야 하지만, LB_{app}를 이용하기 때문에 블록 정합을 행하지 않는 경우로서, 이로 인하여 블록 정합을 행하는 탐색점의 수는 감소하지만 움직임 추정 오차가 커질 수 있다.

이러한 MAE의 최소 범위의 오차로 인한 블록 정합이 필요한 탐색점의 증가량과 감소량 및 두 가지를 모두 고려한 전체적인 증가량을 표 2에 나타내었다. 이 표는 실험 영상 전체에 대하여 구한 것으로서, 탐색점의 증가는 그림 5의 (c)의 경우에 의하여 발생하는 것이고, 탐색점의 감소는 그림 5의 (f)에 의한 것이며, 증가 및 감소량은 전체 탐색점에 대한 비율로 나타내었다. 이 표를 살펴보면 수평 및 수직 방향에 비하여 최소 범위의 오차가 큰 대각 방향에서의 탐색점의 증가

및 감소량이 상대적으로 많은 것을 알 수 있다. 또한, 전체적으로 블록 정합이 필요한 탐색점이 약 0.18% 정도로 매우 적게 증가함을 확인할 수 있는데, 이는 최소 범위의 오차로 인한 계산량의 증가가 매우 적음을 나타낸다.

1) 최소 범위의 오차가 움직임 추정 오차에 미치는 영향

앞에서 설명한바와 같이 최소 범위의 오차로 인한 블록 정합이 필요한 탐색점의 감소는 움직임 추정 오차에 영향을 미칠 수 있다. 그러나, 움직임 추정 오차가 커지는 경우는 탐색 영역 내에서 최소 MAE를 가지는 탐색점에서 최소 범위의 오차로 인하여 블록 정합을 행하지 않는 경우이다. 그러므로 표 2에 나타낸 탐색점의 감소가 움직임 추정 오차의 증가를 의미하지는 않는다. 그러므로 본 논문에서는 제안한 방법으로 구한 움직임 벡터가 FSA로 구한 최적의 움직임 벡터와 다른 경우의 확률 즉, 움직임 벡터를 잘못 찾을 확률을 구하여 표 3에 나타내었다. 또한, 움직임 벡터를 잘못 찾은 경우에 FSA를 이용하여 최적의 움직임 벡터를 찾은 경우와의 MAE 차의 분포를 표 4에 나타내었다. 이 표에서 MAE의 차를 8까지만 나타낸 것은 본 실험에서 MAE의 차가 8이상이 되는 경우는 없었기 때문이다. 표 3을 살펴보면, 영상에 따라 약간의 차이는 있지만 제안한 방법에서 찾은 움직임 벡터가 FSA로 찾은 움직임 벡터와 다를 확률이 약 1.61%로 매우 낮음을 알 수 있고, 표 4로부터 제안한 방법이 최적의 움직임 벡터를 찾지 못하더라도, 이로 인하여 발생하는 오차의 증가는 매우 적음을 알 수 있다. 즉, 제안한 방법이 움직임 벡터를 잘못 찾은 경우에, 최적의 움직임 벡터에 의한 MAE와 제안한 방법으로 찾은 움직임 벡터에 의한 MAE의 차는 1인 경우가 많고, 차가 3이하인 경우가 대부분을 차지함을 표 4로부터 확인할 수 있다. 특히, FOOTBALL 영상의 경우는 움직임 벡터를 잘못 찾을 확률이 0.44%로 매우 낮음을 알 수 있다. 이를 프레임 당 개수로 환산하면 약 6개의 블록이 된다. 즉, 720×480 크기의 영상에 대하여 움직임을 추정할 때 16×16 크기의 블록을 사용하였다면, 한 프레임의 총 블록 개수는 1350개가되는데, 이 1350개의 블록 중 약 6개의 블록의 움직임 벡터가 FSA를 이용하여 구한 최적의 움직임 벡터와 다름을 나타낸다. 그리고, 표 4를 살펴보면, 잘못 찾은 움직임 벡터의 경우의 대부분인 약 77%의 경우에 최적의 움직임 벡터에 의한 MAE와의 차가 1임을 알 수 있다. 이로부터 최소 범위의 오차에 의한 움직임 추정 오차의 증가가 거의 없음을 알 수 있다. 그리고, 표 3에서 평균 확률의 두 배인 3.32%의 확률을 나타내고 있는 SUSIE 영상의 경우는 1350개의 블록 중 약 45개의 적은 숫자의 블록에서 오차가 발생하고 있음을 나타내고 있으며, 표 4로부터 이들 중 84%인 약 38개의 블록이 최적의 움직임 벡터에 의한 MAE와의 차가 1로서 매우 작음을 알 수 있고, 98% 이상이 3 이하의 매우 작은 차를 가짐을 알 수 있다. 또한, 표 3과 표 4를 전체적으로 살펴볼 때, 제안한 방법이 최적의 움직임 벡터를 찾지 못할 확률은 매우 낮으며, 최적의 움직임 벡터를 찾지 못하는 경우에도 최적의 움직임 벡터와 MAE 차가 매우 작음을 확인할 수 있다.

이상의 결과로부터 제안한 방법에서 MAE의 실제 최소 범위를 사용하지 않고, MAE의 근사 최소 범위를 사용함으로서 발생하는 최소 범위의 오차는 움직임 추정 성능에 거의 영향을 미치지 않음을 확인할 수 있었다.

2. 제안한 2단계 고속 블록 정합 알고리듬의 움직임 추정 성능

본 절에서는 제안한 방법의 움직임 추정 성능을 움직임 보상된 영상과 원 영상간의 PSNR과 움직임 추정에 필요한 계산량을 이용하여 움직임 추정 오차 측면에서 최적의 성능을 가지는 FSA 알고리듬, 기존의 TSS 알고리듬, 및 기존 OTSS 알고리듬과 비교, 평가하였다.

본 실험에 사용된 FOOTBALL, FLOWER GARDEN, SUSIE, 및 MOBILE 영상에 대한 40 프레임 평균 PSNR을 기존의 FSA, TSS 및 OTSS 알고리듬과 비교하여 표 5에 나타내었다. 이 표로부터 FOOTBALL 영상의 경우에는 제안한 알고리듬의 PSNR이 움직임 추정 오차 측면에서 최적의 성능을 나타내는 FSA와 거의 동일함을 확인할 수 있었으며, FLOWER GARDEN, MOBILE, 및 SUSIE 영상의 경우에는 약 0.03 dB에서 0.17 dB 정도의 적은 PSNR 감소를 나타냄을 확인할 수 있으며, 이로부터 제안한 알고리듬은 FSA에 근접하는 성능을 나타냄을 확인할 수 있었다. 그리고, 기존의 TSS 알고리듬과 제안한 알고리듬의 PSNR을 비교해 보면, 제안한 알고리듬의 PSNR이 영상에 따라 약 3.2 dB에서 1.0 dB 정도로 크게 높음을 확인할 수 있다. 또한, 제안한 알고리듬의

표 5. 실험 영상들에 대한 PSNR 비교
Table 5. The comparison of PSNR of test sequences.

Sequence	FOOTBALL	FLOWER GARDEN	SUSIE	MOBILE
FSA	24.74 dB	27.42 dB	36.60 dB	32.22 dB
TSS	23.78 dB	24.19 dB	34.74 dB	30.0 dB
OTSS	24.10 dB	26.60 dB	36.08 dB	31.33 dB
Proposed method	24.74 dB	27.39 dB	36.43 dB	32.11 dB

표 6. 실험 영상들에 대한 계산량 비교
Table 6. The comparison of computational complexity for test sequences.

Sequence	FOOTBALL	FLOWER GARDEN	SUSIE	MOBILE
FSA	100.0%	100.0%	100.0%	100.0%
Proposed method	35.4%	28.8%	32.2%	33.9%
Reduction ratio	64.2%	71.2%	67.8%	66.1%

PSNR은 기존의 OTSS 알고리듬의 PSNR에 비하여 영상에 따라 약 0.8 dB에서 0.4 dB 정도 우수함을 이 표로부터 확인할 수 있었으며, 이로부터 기존의 OTSS 알고리듬은 TSS 알고리듬의 움직임 추정 성능은 어느 정도 개선하였지만, 제안한 알고리듬이나 FSA에 비하여서는 여전히 큰 움직임 추정 오차를 가짐을 확인할 수 있었다.

실험 영상들에 대한 40 프레임 평균 계산량을 표 VI에 나타내었다. 여기서, 제안 방법의 계산량은 FSA의 계산량에 대하여 상대적으로 나타내었으며, 기존의 TSS 알고리듬 및 OTSS 알고리듬의 계산량은 각각 FSA의 3.7% 및 13.9%로 일정하기 때문에 이 표에는 나타내지 않았다. 이 표를 살펴보면, 제안한 방법의 계산량은 FSA의 계산량의 약 35%에서 29%임을 알 수 있다. 즉, 제안한 방법으로 움직임 추정을 행하는 경우에는 FSA를 이용하는 경우에 비하여 약 65%에서 71% 정도의 많은 계산량 감소를 얻을 수 있음을 이 표로부터 확인할 수 있다.

또한, 제안한 방법의 계산량은 식 (23)에서 볼 수 있는바와 같이, MAE의 최소 범위가 기준 MAE 보다 작은 탐색점의 개수에 따라 달라진다. 그러므로, 프레임에 따른 계산량의 변화를 알아보기 위하여 각 실험 영상의 프레임별 계산량 감소율을 그림 6에 나타내었다. 이 그림으로부터 FLOWER GARDEN, SUSIE, 및 MOBILE 영상의 경우에는 계산량 감소율이 프레임에 따라 평균값을 중심으로 큰 변화가 없음을 확인할 수 있다. 또한, FOOTBALL 영상의 경우에는 계산량 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 부분에 대하여서는 계산량의 감소가 다소 적음을 알 수 있고, 프레임이 진행되면서 많은 계산량의 감소를 얻으면서 계산량 감소율이 안정화됨을 이 그림으로부터 확인할 수 있다. 또한 SUSIE 영상 및 MOBILE 영상의 경우에는

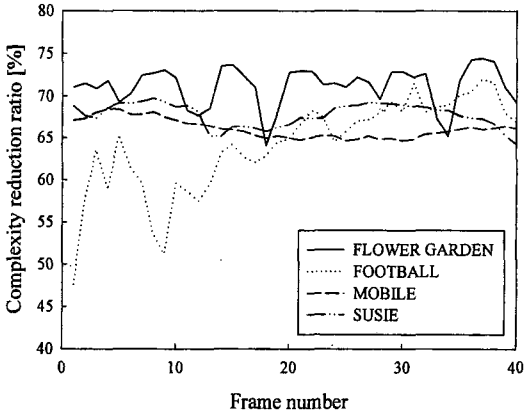


그림 6. 제안한 방법의 계산량 감소율
Fig. 6. The computational complexity reduction ratio of proposed method.

프레임에 따른 계산량의 변화가 약 4% 에서 5% 정도로 매우 안정되어 있음을 확인할 수 있었다.

이상의 결과로부터 제안한 알고리듬이 기존의 TSS 및 OTSS 알고리듬에 비하여 계산량은 조금 많지만 움직임 추정 오차 측면에서 월등한 성능을 나타냄을 확인할 수 있었다. 또한, 제안한 방법은 움직임 추정 오차 측면에서 최적인 FSA와 거의 같은 움직임 추정 성능을 나타내면서도, 많은 계산량의 감소를 얻을 수 있음을 확인할 수 있었다.

VI. 결 론

본 논문에서는 이웃 탐색점에서의 MAE를 이용하여 FSA와 거의 같은 움직임 추정 성능을 얻으면서도 고속으로 움직임을 추정할 수 있는 2단계 고속 블록 정합 알고리듬을 제안하였다. 제안한 방법에서는 현재 탐색점에서 블록 정합을 통하여 얻을 수 있는 MAE의 최소 범위를 이웃 탐색점에서의 MAE를 이용하여 구한 뒤, 이를 이용하여 블록 정합이 필요한 탐색점에 대하여서만 블록 정합을 행함으로써 고속으로 움직임을 추정하였다. 이때, 움직임 추정은 두 단계로 나누어서 수행하였다. 또한, 제안한 방법에서는 현재 탐색점에서의 MAE의 최소 범위를 구하기 위해 필요한 부가 계산량을 줄이기 위하여, MAE의 최소 범위를 사용하지 않고 근사 최소 범위를 구하여 사용하였다.

모의 실험을 통하여 제안한 방법에서 MAE의 근사 최소 범위를 사용함으로서 오차가 발생할 수 있으나, 움직임 추정 성능에는 거의 영향을 미치지 않음을 확인할 수 있었으며, 제안한 방법이 FSA와 거의 같은 움직임 추정 성능을 유지하면서도 많은 계산량의 감소를 얻을 수 있음을 확인하였다.

참 고 문 헌

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

[2] *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] 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.

[6] 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.

[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-aG5.3.5, 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] 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.

[12] 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.

[13] 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.

[14] W. Li and E. Salari, "Successive elimination

algorithm for motion estimation," *IEEE Trans. Image Processing*, vol. 4, no. 1. pp. 105-107, Jan. 1995.

[15] 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.

[16] 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.

[17] 정원식, 이법기, 이경환, 최정현, 김경규, 김덕규, 이건일, "블록 움직임 추정을 위한 2단계 고속 전역 탐색 알고리듬," 한국통신학회논문지, vol. 24, no. 9A, pp. 1392-1400, 1999년 9월

[18] H.-M. Jong, L.-G. Chen, and T.-D. Chiueh, "Accuracy Improvement and Cost Reduction of 3-Step Search Block Matching Algorithm for Video Coding," *IEEE Trans. Circuit and Systems for Video Technology*, vol. 4, no. 1, pp. 88-90, Feb. 1994.

저 자 소 개

鄭 元 植(正會員) 第 34卷 S編 第 9號 參照

權 成 根(正會員)

1996년 2월 경북대학교 전자공학과 졸업(공학사). 1998년 2월 경북대학교 대학원 전자공학과 졸업(공학석사). 1998년 3월~현재 경북대학교 대학원 전자공학과 박사과정 재학중 주관심분야는 영상신호처리 및 시스템 설계

申 容 達(正會員) 第 36卷 S編 第 4號 參照

李 健 一(正會員) 第 34卷 S編 第 9號 參照

李 法 基(正會員) 第 35卷 S編 第 5號 參照

韓 纘 豪(正會員)

1990년 2월 경북대학교 전자공학과 졸업(공학사). 1992년 2월 경북대학교 대학원 전자공학과 졸업(공학석사). 1992년~1997년 8월 현대전자 미디어 연구소 연구원 1997년 8월~현재 경북대학교 대학원 전자공학과 박사과정 재학중 2000년~현재 경운대학교 멀티미디어정보학부 소프트웨어 전공 전임강사 주관심분야는 영상신호처리, 비디오 공학, 음향공학

宋 奎 翼(正會員) 第 36卷 S編 第 12號 參照