論文 97-22-9-21

인접한 블럭의 움직임 벡터를 이용한 수정된 삼단계 움직임 추정 기법

正會員 오 황 석*, 백 윤 주*, 이 흥 규*

Modified Three Step Search Using Adjacent Block's Motion Vectors

Hwang-Seok Oh*, Yun-Ju Baek*, Heung-Kyu Lee* *Regular Members*

요 약

움직임 보상 부호화 기법은 연속한 비디오 프레임간의 시간적 중복성을 제거하기 때문에 비디오 영상 압축에 매우 중요한 역할을 한다. 그러나 많은 계산량으로 인하여 실시간 응용이나 고해상도 응용에 많은 어려움이 있다. 이러한 문제점을 해결하기 위하여 빠른 탐색 기법과 하드웨어 설계 기법이 활발히 연구되어 왔다. 특히 계산량을 크게 줄이고 안정된 성능을 갖는 삼단계 탐색 기법이 널리 이용되고 있으며, 이를 기반으로 한 새로운 탐색 기법들이 제안되었다. 본 논문에서는 인접한 블럭들의 움직임 벡터가 미치는 영향을 고려하여 수정된 삼단계 탐색 기법을 제안하고 이의 성능을 평가한다. 실험에 의하여 제안된 기법이 삼단계 탐색 기법에 비교하여 적은 계산량을 가지며, MAE 측면에서 이득이 있음을 보였다.

ABSTRACT

The motion compensated video coding technology is very important to compress video signal since it reduces the temporal redundancies in successive frames. But the computational complexity of the motion estimation(ME) is too enormous to use in the area of real-time and/or high resolution video processing applications. To reduce the complexity of ME, fast search algorithms and hardware design methods are developed. Especially, the three step search(TSS) is well known method which shows stable performance in various video sequences. And other variations of TSS are developed to get better performance and to reduce the complexity. In this paper we present the modified TSS using neighboring block's motion vectors to determine first step motion vector in TSS. The presented method uses the correlation of the adjacent blocks with same motion field. The simulation results show that it has a good MAE performance and low complexity comparing with original TSS.

*한국과학기술원 전산학과
論文番號:96296-0914
接受日字:1996年 9月 14日

Ⅰ. 서 론

비디오 영상은 연속한 프레임간의 높은 상관성으로 인하여 많은 시간적 중복성을 지니고 있다. 비디오 영상에서 연속한 프레임간의 움직임 정보를 찾고 움직임 정보를 전송한다면 개개의 프레임 내에서 압축뿐만 아니라 시간적 중복성을 제거하여 고압축을 할 수 있다. 일반적으로 많이 알려진 움직임 정보를 찾는 기법으로 움직임 추정(motion estimation)이 있으며, 이는 크게 블럭 단위로 동일한 움직임을 갖는다는 가정하에서 움직임 정보를 찾는 블럭 정합 알고리즘(block matching algorithm)과 경사법을 이용하여 화소 단위로 움직임을 추정하는 화소 재귀적 알고리즘(pel-recursive algorithm)으로 나눌 수 있다. 화상 전화기, 고화질 TV, 화상 회의 그리고 요구형 비디오 등과 같은 응용 분야에서 비디오 코딩을 위해서 움직임 추정 기법이 매우 중요한 요소로 사용되고 있다. 현재 많은 비디오 코딩 알고리즘에서는 데이터 흐름의 규칙성, 계산의 복잡도 등을 고려하여 블럭 정합 기법이 많이 사용되고 있다[4, 5, 6].

블럭 정합 기법에서 탐색 영역 내의 모든 후보 블럭과 차이를 비교하여 가장 유사한 블럭을 찾는 전역 탐색 기법이 많이 이용되고 있지만 많은 계산량으로 인하여 실시간 비디오 코딩 응용 분야 및 소프트웨어 구현에 많은 어려움이 있다. 이러한 계산량을 줄이기 위해서 현재까지 블럭 정합 기법을 이용하는 많은 빠른 움직임 추정 기법들이 개발되어 왔다. 이들은 블럭간의 최소 차이를 보이는 곳에서 멀어질수록 단조 증가한다는 가정을 기반으로 몇 개의 방향성을 지닌 후보 블럭에서 움직임 추정을 하는 탐색 방법들(TSS, OTS, CSA, etc.)이 있다[1, 2, 3, 7, 8, 9, 10, 11, 12, 13].

이들 중 삼단계 탐색 기법은 영상의 특성에 무관하게 안정적인 결과를 보여주고 있으며[1, 10] 전역 탐색에 비하여 계산량이 매우 적다. Renxiang[10]은 움직임 벡터들이 원점을 중심으로 분포되어 있다는 가정하에서 삼단계 탐색 기법을 수정한 새로운 탐색 기법을 제안하였다. 그러나 위 탐색 기법은 움직임이 비교적 작은 즉, 움직임 벡터가 원점을 중심으로 분포되어 있을 경우에 매우 효율적이지만 유사한 움직임을 갖는 블럭간의 상호 상관성을 고려하지 않았다. 본 논문에서는 비디오 영상에서 인접한 블럭간의 움직임 벡터의 상관성을 밝히고, 이를 삼단계 탐색 기법에 적용하였을 경우 계산량 및 성능에 어떠한 영향을 미치는가를 알아본다.

본 논문의 구성은 다음과 같다. 제 2장에서 블럭 정합 알고리즘 및 삼단계 탐색 기법과 이의 수정된 기법에 대하여 간단히 살펴보고 제 3장에서는 인접한 블럭의 움직임 벡터를 이용한 삼단계 탐색 기법에 대하여 기술한다. 그리고 제 4장에서 제안한 기법의 성능과 복잡도를 비교 분석하며, 제 5장에서 결론을 맺는다.

Ⅱ. 블럭 정합 알고리즘

블럭 정합 알고리즘은 블럭 단위로 탐색 영역 내에서 블럭 정합 기준에 의하여 차이가 가장 작은 블럭을 찾아 이에 해당되는 가로, 세로의 변위 즉 움직임 벡터를 추정하는 알고리즘이다[11, 12]. 블럭 정합 알고리즘은 영상의 한 프레임을 여러개의 동일한 크기의 블럭으로 겹치지 않게 나누고 이들의 각 블럭에 대하여 참조 프레임(reference frame)의 탐색 영역(search area) 내에서 정합 오차가 가장 작은 블럭을 찾는다. 이 때 현재 프레임의 한 블럭과 이전 프레임 내에서 가장 정합이 잘 되는 블럭간의 위치 차이를 움직임 벡터라고 한다. 그림 1은 블럭 정합 알고리즘을 이용하여 움직임 벡터를 찾는 것을 보여준다.

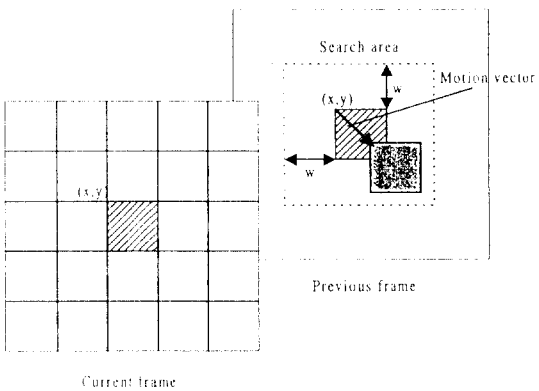


그림 1. 블럭 정합 알고리즘

현재 프레임의 한 블럭 크기를 $N \times N$, 탐색 영역의 최대 변위를 p라고 하면 탐색 영역의 크기는 $(2p+N)$

$\times(2p+N)$이 되며, $(2p+1)^2$개의 후보 블럭들과 정합이 이루어진다. 블럭 정합 알고리즘을 이용한 움직임 추정에서 $R(i, j)$를 현재 프레임의 한 블럭에 대한 왼쪽 위의 좌표에 해당하는 화소의 밝기, 그리고 $S(i+m, j+n)$을 참조 프레임에서 m 화소와 n 라인의 변위를 가지는 후보 블럭의 왼쪽 위의 좌표에 해당하는 화소의 밝기라고 하면 움직임 추정은 다음과 같이 기술된다.

$$\vec{s(D)}=\sum_{i=1}^{N}\sum_{j=1}^{N} Dist(R(z, t), S(z-\vec{D}, t-1))$$
$$\text{where } z=(i, j),\ \vec{D}=(m, n),\ -p\le m, n\le p \qquad (1)$$
$$(u, v)=\{(m, n)\,|\,\min_{m, n}\vec{s(D)}\}$$

여기서 $\vec{s(D)}$는 정합에 사용된 블럭 정합 기준에 의하여 참조 블럭과 후보 블럭들 사이의 차이며, z는 프레임 내에서 x, y의 좌표를 갖는 화소 위치이다. (u, v)는 현재 블럭과 탐색 영역 내의 모든 블럭에 대하여 가장 적은 블럭간의 오차를 가지는 블럭에 대한 변위 차이이며, 이것을 움직임 벡터라고 한다[4, 5, 6].

블럭 정합 알고리즘에서 계산량과 움직임의 정확성에 영향을 미치는 요소로 블럭간의 정합 기준과 움직임 벡터 탐색 방법이 있다. 블럭 정합 알고리즘에서 탐색 시 모든 후보 블럭에 대하여 정합 기준에 의하여 가장 유사한 블럭을 찾는 기법을 전역 탐색이라고 한다. 그러나 많은 계산량으로 인하여 실시간 응용에 어려움이 있어 빠른 탐색 기법들[1, 2, 3, 7, 8, 9, 10, 11, 12, 13]이 연구되어왔다. 그들 중 대표적인 것이 삼단계 탐색 기법[1]으로 다음 알고리즘과 같이 움직임 추정을 수행한다.

알고리즘 1 : 삼단계 탐색 알고리즘(Three Step Search : TSS)

초기화 :

- 원점을 탐색의 중심으로, 탐색 변위를 최대 변위의 반으로 설정한다.

단계 1 :

- 탐색 중심과 가로, 세로 탐색 변위에 있는 8개의 후보 블럭과 정합을 수행한다.

단계 2 :

- 9개의 후보 블럭과 정합 오차가 가장 작은 블럭을 찾는다.

단계 3 :

- 만약 현재 탐색 변위가 1 이면 단계 2에서 찾은 블럭의 변위를 움직임 벡터로 하고 탐색을 마친다.
- 탐색 변위가 1이 아니면 단계 2에서 찾은 블럭의 위치를 탐색의 중심으로 하고 탐색 변위를 반으로 줄여 단계 1부터 3까지 반복한다.

그림 2는 최대 변위가 가로/세로 ±7인 경우 삼단계 탐색 과정을 보여준다. 먼저 변위가 4인 8개 및 원점의 후보 블럭(원으로 표시됨)과 현재 블럭과 정합을 하여 가장 유사한 블럭을 찾고(그림에서(4, 4) 블럭) 이를 중심으로 변위가 처음 탐색의 반인 2로 하여 주위의 8개 블럭(사각형으로 표시됨)에 대하여 탐색한다. 이와 같은 과정을 변위가 1일 때까지 수행하여 가장 작은 정합 오차를 갖는 블럭을 찾으며, 이때의 현재 블럭과의 변위 차이를 움직임 벡터(그림에서 실선 화살표)라고 한다.

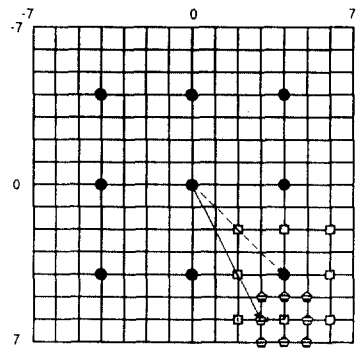


그림 2. 삼단계 탐색 기법에서 움직임 벡터를 찾아가는 과정

삼단계 탐색 기법은 움직임 벡터의 크기가 모든 위치에 동일한 확률로 분포하며, 정합 왜곡이 가장 적은 블럭에서 멀어질수록 단조 증가한다는 가정을 기반으로 몇 개의 방향성을 지닌 후보 블럭에서 움직임 추정한다. 그러나 블럭 단위로 움직임을 추정할 경우 영상에 포함된 배경은 대부분 움직임이 적으며, 원점 부근에 많은 움직임 벡터가 존재한다. 이러한 사실을 기반으로 Renxiang[10]은 새로운 삼단계 탐색 기법을 제안하였다. 그림 3은 새로운 삼단계 탐색 기법과 탐색 시 정합되는 후보 블럭들의 위치를 보여준다.

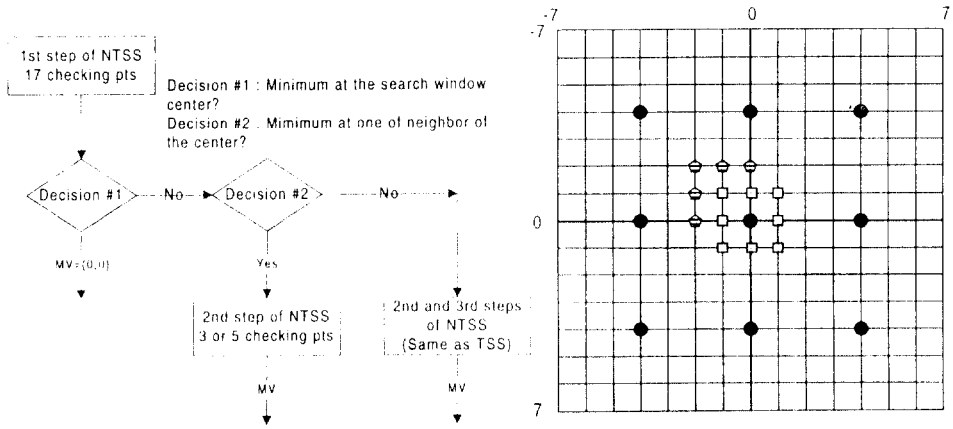


그림 3. 새로운 삼단계 탐색 기법의 알고리즘과 탐색 점 표시

알고리즘 2:새로운 삼단계 탐색 기법(New Three Step Search : NTSS)

초기화:

- 원점을 탐색의 중심으로, 탐색 변위를 최대 변위의 반으로 설정한다.

단계 1:

- 탐색 중심과 원점을 중심으로 변위가 가로/세로 1인 8개 후보 블럭 및 가로/세로 탐색 변위에 있는 8개의 후보 블럭과 정합을 수행한다(그림에서 원과 사각형으로 표시됨).

단계 2:

- 17개의 후보 블럭과 정합 오차가 가장 작은 블럭을 찾는다.

단계 3:

- 만약 정합 오차가 가장 작은 블럭이 탐색 영역의 원점(0, 0)에 위치하면 움직임 벡터를 (0, 0)으로 하고 탐색을 마친다.
- 만약 정합 오차가 가장 작은 블럭이 탐색 영역 원점에 이웃한 8개이 블럭 중 하나이면 이를 중심으로 변위가 1인 8개의 후보 블럭에 대하여 정합을 하고 가장 작은 왜곡을 갖는 블럭의 변위를 움직임 벡터로 하고 탐색을 마친다.
- 정합 오차가 가장 작은 블럭이 탐색 변위 만큼 떨어진 위치에 있다면 삼단계 탐색 기법과 같은 방법으로 탐색을 수행한다.

비디오 영상에는 일반적으로 움직임 벡터가 탐색 영역의 원점을 중심으로 분포하지만 동일한 움직임 객체이거나 카메라의 움직임에 의하여 인접한 블럭간의 큰 상관성이 존재한다[9, 10]. 본 논문에서는 이러한 사실을 기반으로 삼단계 탐색과 새로운 삼단계 탐색 기법에 적용하였을 경우 성능 향상과 복잡도 증가를 제 3장과 4장에 기술한다.

Ⅲ. 인접한 블럭의 움직임 벡터를 이용한 삼단계 움직임 추정

비디오 프레임에서 인접한 블럭간의 움직임 구조는 많은 상관 관계가 있다. 그림 4는 Foreman 영상을 대상으로 100 프레임까지 전역 탐색을 한 경우 인접한 블럭간의 움직임에 대한 상호 상관 관계로서 인접한 블럭간의 각 방향별로 움직임 벡터의 차를 도시하였다. 이 그림으로부터 인접한 블럭 사이에 큰 상관 관계 즉, 움직임 벡터의 차가 0 부근에 많을수록 인접한 블럭간 큰 상관 관계가 있음을 알 수 있다. 프레임 내의 배경 부분은 대부분 움직임이 원점을 중심으로 있지만 객체의 움직임에 대하여서 하나의 객체가 여러 블럭으로 나누어 지는 경우 각 블럭은 유사한 움직임을 가지게 된다. 또한 객체를 중심으로 카메라가 움직이는 경우 모든 블럭이 카메라의 움직임에 대한 상대적인 움직임을 가지게 되며 동일한 움직임 구조를 지닌다. 이러한 사실을 바탕으로 본 논문에서는 인접한 블럭들의 상호 상관성을 이용하여 삼단계 탐색 기법에 적용하였을 경우 성능 향상을 분석한다.

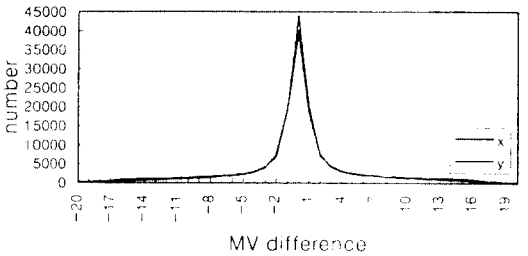


그림 4. 인접한 블럭간의 움직임 벡터 상관관계

3.1 인접한 블럭의 움직임 벡터를 삼단계 탐색 기법에 적용

삼단계 탐색 기법에 인접한 블럭의 움직임 벡터를 이용하여 움직임 추정하는 방법은 다음과 같다. 삼단계 탐색 시 단계 1에서 원점, 최대 변위의 반 만큼 떨어진 8개의 후보 블럭과 그림 5와 같이 인과 관계에 있는 블럭의 움직임 벡터들(그림에서 mv1, mv2, mv3, mv4)를 초기 벡터로 하여 최소의 블럭 오차를 갖는 블럭을 찾아 이를 중심으로 삼단계 탐색을 계속한다. 탐색 알고리즘은 다음과 같다.

알고리즘 3:인접한 블럭의 움직임 벡터를 이용한 삼단계 탐색 기법(ATSS)

초기화:

- 원점을 탐색의 중심으로, 탐색 변위를 최대 변위의 반으로 설정한다.

단계 1:

- 인과 관계에 있는 인접한 블럭의 움직임 벡터 만큼 떨어진 블럭과 현재 블럭의 정합을 수행한다.

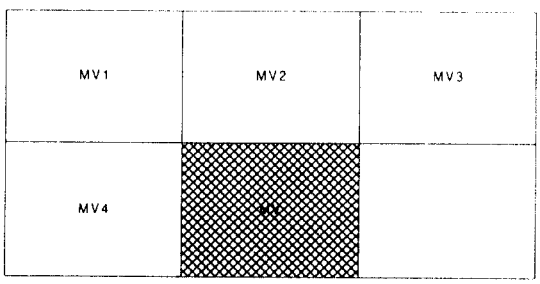


그림 5. 삼단계 탐색 시 인과 관계에 의하여 찾아진 움직임 벡터 이용
(빗금 친 부분이 현재 블럭이며, 인접한 블럭의 움직임 벡터 mv1, mv2, mv3, mv4를 이용한다.)

단계 2:

- 탐색 중심과 가로, 세로 탐색 변위에 있는 8개의 후보 블럭과 현재 블럭의 정합을 수행한다.

단계 3:

- 후보 블럭과 정합 오차가 가장 작은 블럭을 찾는다.

단계 4:

- 만약 탐색 변위가 1 이면 단계 3에서 찾은 블럭의 변위를 움직임 벡터로 하고 탐색을 마친다.
- 탐색 변위가 1이 아니면 단계 3에서 찾은 블럭의 위치를 탐색의 중심으로 하고 탐색 변위를 반으로 줄여 단계 2부터 4까지 반복한다.

3.2 인접한 블럭의 움직임 벡터를 새로운 삼단계 탐색 기법에 적용

Renxiang[10]의 수정된 삼단계 탐색 기법은 움직임이 원점 중심에 많이 분포할 경우 매우 큰 효과를 얻을 수 있다. 그러나 움직임이 큰 경우와 인접한 블럭간의 움직임 구조가 비슷할 경우 인접한 블럭의 움직임 벡터를 이용하여 초기 방향을 설정하는 것은 늘어난 계산량에 비하여 큰 성능 향상을 가져올 수 있다. 탐색 알고리즘은 다음과 같다.

알고리즘 4:인접한 블럭의 움직임 벡터를 이용한 수정된 삼단계 탐색 기법(ANTSS)

초기화:

- 원점을 탐색의 중심으로, 탐색 변위를 최대 변위의 반으로 설정한다.

단계 1:

- 탐색 중심과 원점을 중심으로 변위가 가로세로 1인 8개 후보 블럭 및 가로/세로 탐색 변위에 있는 8개의 후보 블럭, 인과 관계에 있는 인접한 블럭의 움직임 벡터 만큼 떨어진 블럭과 정합을 수행한다.

단계 2:

- 후보 블럭과 정합 오차가 가장 작은 블럭을 찾는다.

단계 3:

- 만약 정합 오차가 가장 작은 블럭이 탐색 영역의 원점(0, 0)에 위치하면 움직임 벡터를 (0, 0)으로 하고 탐색을 마친다.
- 만약 정합 오차가 가장 작은 블럭이 탐색 영역 원점에 이웃한 8개이 블럭 중 하나이면 이를 중심으로 변위가 1인 8개의 후보 블럭에 대하여 정합을

하고 가장 작은 왜곡을 갖는 블럭의 변위를 움직임 벡터로 하고 탐색을 마친다.

- 정합 오차가 가장 작은 블럭이 탐색 변위 만큼 떨어진 위치에 있다면 삼단계 탐색 기법과 같은 방법으로 탐색을 수행한다.

Ⅳ. 실험 및 결과

제안된 인접 블럭의 움직임 벡터를 이용한 수정된 삼단계 탐색 기법의 성능을 평가하기 위하여 실험 영상으로 CIF(common intermediate format : 352x288)의 Miss America, Foreman, Carphone, 그리고 Claire영상과 704x480 크기의 Fashion model 영상을 사용하였다. Miss America와 Claire 영상은 움직임이 많지 않으며, Foreman과 Carphone 영상은 다소 움직임이 큰 영상으로 배경의 변화가 있는 영상이다. 그리고, Fashion model 영상은 다른 영상에 비하여 움직임이 매우 크며, 빠르게 변화는 영상이다. 움직임 추정 기법의 성능을 평가하기 위해서 움직임 추정 시 각 블럭에 대한 MAE(mean absolute error)와 각 움직임 벡터를 얻기까지 수행된 정합 블럭 수(number of matching block : NOMB)를 사용하였으며, 제안한 두 가지 방법(인접한 블럭의 움직임 벡터를 이용한 삼단계 탐색 기법 : ATSS, 인접한 블럭의 움직임 벡터를 이용한 수정된 삼단계 탐색 기법 : ANTSS)과 전역 탐색(full search : FS), 삼단계 탐색(three step search : TSS), 새로운 삼단계 탐색(new three step search : NTSS), 그리고 최근에 발표된 4단계 탐색 기법[13]과 비교하였다.

표 1과 2는 각 실험 영상에 대하여 최대 변위가 16, 8인 경우 움직임 추정 후 각 블럭의 MAE 값과 각 블럭의 움직임 벡터를 찾기까지 정합이 이루어진 블럭의 수를 보여준다. 움직임이 작은 Miss America나 Claire 영상에서 움직임 최대 변위가 16인 경우 한 블럭의 움직임 추정을 위하여 ATSS, ANTSS는 각각 TSS, NTSS에 비하여 1~3개의 블럭 정도 많은 정합이 되었으며, MAE도 크게 감소하지는 않았다. 그러

표 1. 실험 영상에 대한 블럭 단위 MAE와 탐색시 정합 블럭 수(NOMN), (블럭의 크기 : 16x16, 탐색 영역의 크기 : 16)

실험 영상 및	Miss America		Foreman		Carphone		Claire		Model	
탐색 방법	MAE	NOMB	MAE	NOMB	MAE	NOMB	MAE	NOMB	MAE	NOMB
FS	685.95	984.91	1565.85	984.91	1365.38	984.91	573.39	984.91	1850.47	984.91
TSS	748.60	31.23	1838.05	30.95	1560.16	30.83	601.69	30.92	1861.48	31.67
NTSS	701.16	22.02	1785.03	21.81	1485.62	21.87	589.26	19.15	1879.04	20.17
ATSS	708.16	34.4	1721.27	34.15	1469.09	33.70	588.20	33.96	1835.84	33.48
ANTSS	698.39	23.98	1775.97	26.79	1459.44	25.58	585.17	20.43	1852.95	21.69
FFS	736.95	19.61	1799.84	19.51	1556.56	9.28	595.77	17.42	1928.89	18.49

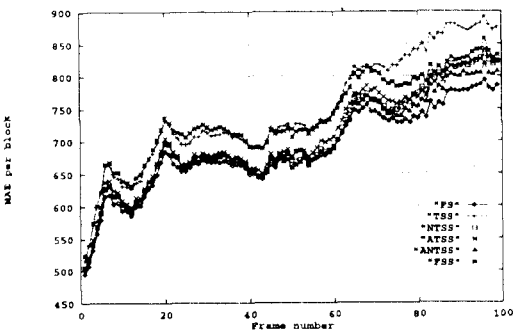
표 2. 실험 영상에 대한 블럭 단위 MAE와 탐색시 정합 블럭 수(NOMN), (블럭의 크기 : 16x16, 탐색 영역의 크기 : 8)

실험 영상 및	Miss America		Foreman		Carphone		Claire		Model	
탐색 방법	MAE	NOMB	MAE	NOMB	MAE	NOMB	MAE	NOMB	MAE	NOMB
FS	690.29	275.42	1658.43	275.42	1405.25	275.42	574.35	275.42	1848.16	275.42
TSS	743.04	23.64	1853.45	23.42	1562.09	23.36	600.94	23.44	1934.62	24.00
NTSS	703.40	21.53	1797.01	21.68	1496.35	21.65	591.10	19.32	1939.35	19.93
ATSS	707.50	26.33	1737.04	26.35	1473.06	26.12	596.33	25.45	1923.30	25.75
ANTSS	701.35	22.93	1736.52	23.54	1463.47	23.86	589.21	20.19	1905.59	21.29
FFS	736.95	19.60	1799.84	19.51	1556.56	19.28	595.77	17.41	1928.89	18.49

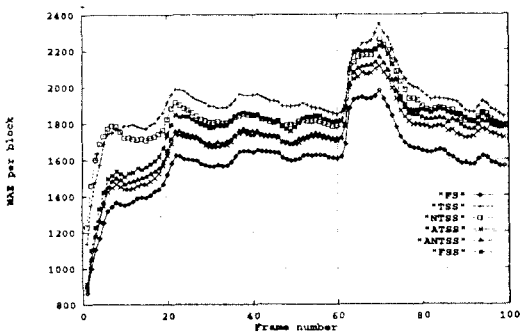
나 Foreman과 Carphone 영상의 경우는 정합 되는 블럭 수도 증가하였으며, MAE도 크게 감소하였다.

특히 변위가 8인 경우는 움직임이 변위를 벗어나는 경우가 많으며, 이로 인하여 인접한 블럭의 움직임 벡터를 초기 벡터로 하여 탐색을 할 경우 많은 이득이 있음을 알 수 있다. 실제 움직임이 최대 변위를 넘어가는 경우 인접한 블럭의 움직임 벡터를 초기로 사용하여 삼단계 탐색을 수행할 경우 상관 관계를 이용하여 최대 변위 밖의 움직임을 찾을 수 있다. 이로 인하여 Foreman과 Carphone 영상에서 정합 블럭 수가 각각 2개 정도 늘어난 반면, MAE 이득은 60, 30으로 나타났다.

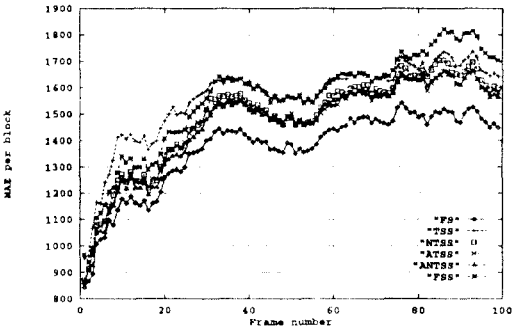
그림 6과 7은 블럭의 크기 16x16, 탐색 영역의 최대 변위가 가로,세로로 16일 경우와 블럭의 크기가 16x16, 탐색 영역의 최대 변위가 8인 경우 각 실험 영상에 대한 프레임 변화에 따른 블럭 당 MAE 변화를 보여준다. 움직임이 적은 실험 영상에서는 블럭 당 평균 MAE의 이득은 크지 않지만 Foreman과 Carphone 영상에서는 제안한 기법의 영향이 큼을 알 수 있다. 이러한 현상은 그림 7에서 더욱 뚜렷하게 나타난다.



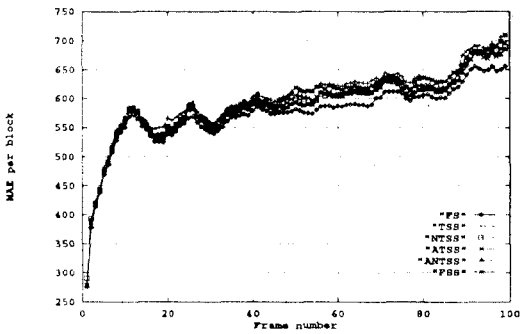
(a) Miss America



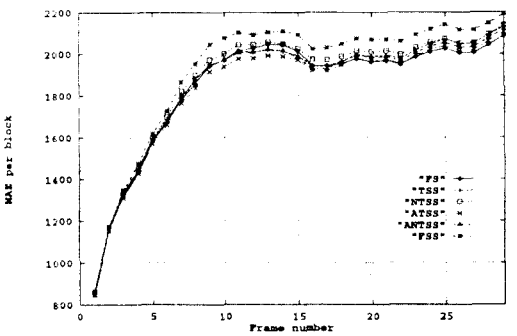
(b) Foreman



(c) Carphone



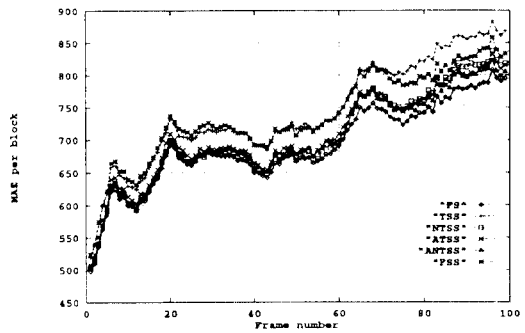
(d) Claire



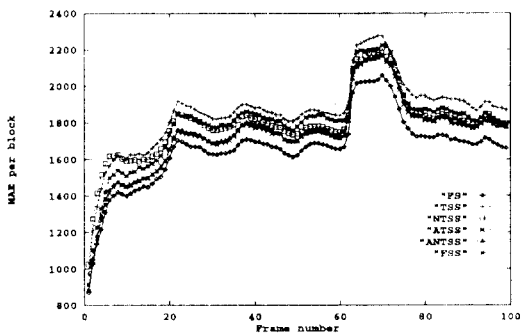
(f) Fashion model

그림 6. 실험 영상에 대한 움직임 추정 후 프레임에 따른 블럭 예측 오차 변화(블럭 크기 16x16, 탐색 변위 +/−16)

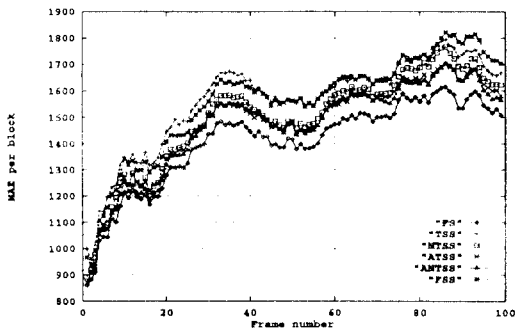
실험을 통하여 인접한 블럭의 움직임 벡터를 이용하여 삼단계 탐색을 수행할 때 움직임이 큰 경우나 비슷한 움직임 구조를 갖는 경우 인접한 블럭과의 상관 관계에 의하여 탐색 시 초기 벡터가 설정되므로 탐색 변위가 TSS나 NTSS보다 커지기 때문에 제안한 기법이 매우 큰 효과를 지님을 알 수 있다.



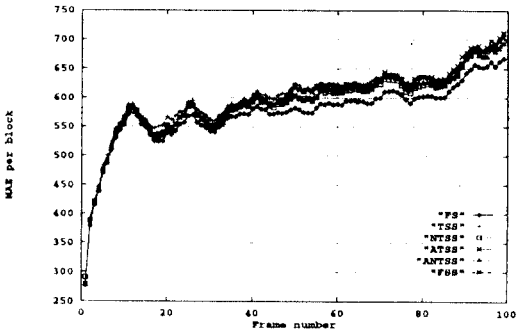
(a) Miss America



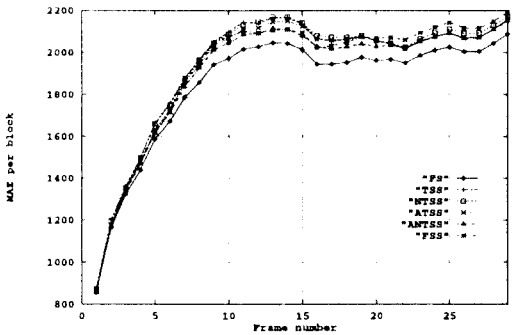
(b) Foreman



(c) Carphone



(d) Claire



(f) Fashion model

그림 7. 실험 영상에 대한 움직임 추정 후 프레임에 따른 블럭 예측 오차 변화(블럭 크기 16x16, 탐색 변위 +/-8)

V. 결 론

본 논문에서는 블럭 정합 시 계산량을 줄이기 위해서 개발된 삼단계 탐색 기법 및 새로운 삼단계 탐색 기법에 인접한 블럭간의 상호 상관성을 이용한 움직임 탐색 기법을 제안하고 실험을 통하여 증가된 계산량 및 성능을 비교하였다. 움직임이 매우 적은 즉 블럭간의 상호 상관성이 미약한 비디오 영상에 대해서는 큰 효과를 얻지 못하였지만 움직임이 많은 영상에서는 증가한 계산량에 비하여 예측 오차를 크게 줄일 수 있었다. 제안한 기법을 인접한 블럭간에 상관성이 매우 크고 움직임 구조가 유사한 객체를 많이 갖는 고해상도 비디오 코딩 응용에 적용하였을 경우 큰 성능 향상을 얻을 수 있다.

참 고 문 헌

1. T. Koga, K. linuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," Proc. NTC81 , pp. G5. 3. 1-G5. 3. 5, Dec. 1981.
2. Jaswant R. Jain, and Anil K. Jain, "Displacement Measurement and Its Application in Interframe Image Coding," IEEE Trans. On Communications, vol. Com-29, no. 12, pp. 1799-1808, Dec. 1981.
3. R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," IEEE Trans. On Communications, Vol. 33, pp. 888-896, Aug. 1985.
4. K.M. Yang, M.T. Sun, and L. Wu, "A family of VLSI designs for the motion compensation block-matching algorithm," IEEE Trans. On Circuits Systems., Vol.36, pp. 1317-1325, Oct. 1989.
5. T. Komarek and P. Pirsch, "Array architecture for block matching algorithms," IEEE Trans. On Circuits Systems, Vol.36, pp. 1310-1308, Oct. 1989.
6. L. D. Vos and M. Stegherr, "Parameterizable VLSI architectures for the full-search block-matching algorithm," IEEE Trans. On Circuits Systems., Vol.36, pp. 1309-1316, Oct. 1989.
7. Ghanbari, "The Corss-Search Algorithm for Motion Estimation," IEEE Trans. On Communications, vol. 38, no. 7, Jul. 1990.
8. Liang-Gee Chen, Wai-Ting Chen, Yeu-Shen Jehng, and Tzi-Dar Chiueh, "An Efficient Parallel Motion Estimation Algorithm for Digital Image Processing," IEEE Trans. On Circuits and Systems for Video Technology, vol. 1, no. 4, Dec. 1991.
9. Bede Liu and Andre Zaccarin, "New Fast Algorithms for the Estimation of Block Motion Vectors," IEEE Trans. On Circuits and Systems for Video Technology, vol. 3, no. 2, Apr. 1993.
10. Renxiang Li, Bing Zeng, and Ming L. Liou, "A New Three-Step Search Algorithm for Block Motion Estimation," IEEE Tralns. On Circuits and Systems for Video Technology, vol. 4, no. 4, Aug. 1994.
11. M.-J. Chen, L.-G. Chen, T.-D. Chiueh, and Y.-P. Lee, "A new block-matching criterion for motion estimation and its implementation," IEEE Trans. Circuits Systetms for Video Technollogy, Vol.5, pp. 231-236, Jun. 1995.
12. Feng, K.-T. Lo, H. Mehrpour and A.E. Karbowiak, "Adaptive block matching motion estimation algorithm for video coding," Electronics letters, vol. 32, no. 18, pp. 1542-1543, Aug. 1995.
13. Lai-Min Po and Wing-Chung Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation," IEEE Trans. On Circuit and Systems for Video Technology, vol. 6, no. 3, pp. 313-317, Jun. 1996.

오 황 석(Hwang-Seok Oh) 정회원

1969년 11월 17일생
1992년 2월 : 경북대학교 자연과학대학 전자계산학과 졸업(학사)
1994년 2월 : 한국과학기술원 전산학과 졸업(공학석사)
1994년 3월~현재 : 한국과학기술원 전산학과 박사과정
※주관심분야 : Video coding, Parallel processing for video coding, Multimedia service system

백 윤 주(Yunju Baek) 정회원

1967년 7월 7일생
1990년 2월 : 과학기술대학 전산학과 졸업(학사)
1992년 2월 : 한국과학기술원 전산학과 졸업(공학석사)
1997년 2월 : 한국과학기술원 전산학과 졸업(공학박사)
1997년 2월~현재 : 핸디소프트 근무
※주관심분야 : Video coding, VLSI architecture, Multimedia systems

이 흥 규(Heung-Kyu Lee) 정회원

1955년 10월 3일생
1978년 2월 : 서울대학교 전자공학과 졸업(학사)
1981년 2월 : 한국과학원 전산학과 졸업(이학석사)
1984년 8월 : 한국과학기술원 전산학과 졸업(공학박사)
1985년 3월~1986년 8월 : U. of Michigan, U.S.A(Research Scientist)
1986년 9월~현재 : 한국과학기술원 전산학과 교수(부교수)
1991년~현재 : 공업진흥청 JTC1/SC29(전문위원)
1994년~현재 : ITU-T SG9(VOD 연구그룹 의장)
※주관심분야 : Real-time processing, Fault-tolerant system, Multimedia service system, Real-time operating system