

비디오 코딩을 위한 인접블록 움직임 벡터를 이용한 예측 탐색 알고리즘

곽성근^{1*}

¹인천대학교 디자인학부

A Prediction Search Algorithm in Video Coding by using Neighboring-Block Motion Vectors

Sung-Keun Kwak^{1*}

¹School of Design, University of Incheon

요 약 동영상의 현재 블록의 움직임 벡터와 이전 블록의 움직임 벡터는 시간적 상관성을 갖고 있다. 본 논문에서는 비디오의 시공간적인 특성과 인접 블록 움직임 벡터의 통계적 특성을 이용하는 새로운 예측 탐색 알고리즘을 제안한다. 제안된 ANBA는 이전 프레임과 현재 프레임의 인접 블록의 움직임 벡터들의 평균값으로 구한 후보 벡터와 현재 블록의 이전 시점의 움직임 벡터의 2개의 후보 벡터 중에서 가장 작은 SAD 값을 갖는 점을 정확한 움직임 벡터를 찾기 위한 초기 탐색점 위치로 결정한다.

실험 결과 제안된 방식은 MVFAST와 PMVFAST에 비해 PSNR 값에 있어서 평균적으로 0.01~0.64dB 개선되고 영상에 따라 최고 1.06dB 정도 우수한 결과를 나타내었다.

Abstract There is the temporal correlation of the video sequence between the motion vector of current block and the motion vector of previous block. In this paper, we propose a new prediction search algorithm for block matching using the temporal and spatial correlation of the video sequence and local statistics of neighboring motion vectors.

The proposed ANBA(Adaptive Neighboring-Block Search Algorithm) determines the location of a better starting point for the search of an exact motion vector using the point of the smallest SAD(Sum of Absolute Difference) value by the predicted motion vectors of neighboring blocks around the same block of the previous frame and the current frame and use a previous motion vector.

Simulation results show that PSNR(Peak-to-Signal Noise Ratio) values are improved up to the 1.06dB as depend on the video sequences and improved about 0.01~0.64dB over MVFAST and PMVFAST.

Key Words : Motion Vector, ANBA, Predictor Candidate Point, Block Matching Algorithm

1. 서 론

디지털 비디오 신호 전송에서 프레임간의 시간적 중복성을 줄이기 위한 블록 정합 움직임 추정은 알고리즘의 단순성과 고속 처리를 위한 하드웨어적인 접근과 용이성으로 인해, MPEG 계열이나 H.264/AVC 등의 표준 부호화 방식으로 채택되어 DMB, Mobile Phone 및 초해상도 비디오 전송 같은 다양한 비디오 서비스에 활용되고 있다.

일반적으로 움직임 추정 기법은 영상의 각 단위로 블록 또는 화소 단위로 적용되며 이중, 계산 복잡도 및 하드웨어 구현에 있어서 용이한 블록 단위의 움직임 추정이 널리 사용되고 있다.

블록 단위로 움직임을 추정하는 대표적인 알고리즘이 FS(Full Search)이다. 그러나 전역 탐색 블록 정합 알고리즘은 과정이 간단하고 예측 효율과 추정의 정확도를 고려할 때 전체적으로 좋은 특성을 가지며 하드웨어 구현

본 논문은 인천대학교 2010학년도 자체연구비 지원에 의하여 연구되었음.

*교신저자 : 곽성근(skkwak@incheon.ac.kr)

접수일 11년 06월 14일

수정일 11년 07월 05일

게재확정일 11년 08월 11일

이 용이하고 또한 탐색 영역의 내부 전체를 탐색하면서 가능한 모든 블록들에 대한 정합을 수행하므로 정합 오차가 가장 적은 움직임 벡터를 찾을 수 있지만 많은 계산량이 필요한 단점이 있다.

이러한 FS의 단점을 극복하기 위해 속도가 개선된 TSS(Three Step Search)[1], NTSS(New Tree-Step Search), DS(Diamond Search) 등의 다양한 고속 블록 정합 알고리즘(FBMA: Fast Block Matching Algorithm)이 개발 되었다. 이들 속도 개선 알고리즘은 주로 탐색 영역 내에서 탐색할 위치의 포인터 개수를 감소시켜 계산량의 감소를 유도하는 탐색 패턴에 따라 SAD 계산횟수를 줄임으로써 계산량을 줄였지만, 그에 따라 화질이 열화되는 단점이 있었다.

이러한 고속 블록 정합 알고리즘의 단점을 개선하기 위해 움직임 벡터간의 상관관계를 이용하여 계산량과 화질의 열화를 획기적으로 줄이는 DS 기반의 MVFAST (Motion Vector Field Adaptive Search Technique)와 PMVFAST (Predictive Motion Vector Field Adaptive Search Technique)가 제안되었다[4,5,7]. 그러나 이 방식 또한 참조한 블록의 움직임 벡터들의 상관성이 떨어질 경우 압축 성능이 현저히 떨어지는 단점이 있다.

따라서 본 논문에서는 시공간적으로 인접한 블록들은 비슷한 속도로 거의 같은 방향으로 움직인다는 점[3]을 고려하여 현재 부호화 블록의 인접 블록과 같은 위치에 있는 이전 프레임의 인접 블록들의 움직임 정보를 이용하여 현재 영상의 움직임 후보 벡터를 구한다. 이때 영상의 국부적인 변화에 따른 움직임을 효과적으로 반영하기 위해 현재 블록과 같은 위치에 있는 블록의 움직임 벡터를 또 하나 후보 벡터를 설정한다.

그리고 두 개의 후보 벡터에 대한 정합 오차(BDM: Block Distortion Measure)를 비교하여 더 작은 값을 갖는 후보 벡터에서부터 탐색을 수행하여 움직임 벡터의 상관성을 높임으로써 효과적으로 탐색점 수를 줄이고, 또한 비디오의 움직임 특성이 원점에 근접한 성질을 이용할 수 있는 3X3 정사각형 패턴을 적용함으로써 기존의 탐색 패턴을 이용한 알고리즘의 국부적 최소(Local Minimum)로 인한 시각적인 화질 저하의 문제점과 PMVFAST이 화면내 움직임이 크고 평행이동 특성이 두드러진 영상에서 성능이 떨어지는 단점을 개선하는 탐색 기법인 ANBA (Adaptive Neighboring-Block Search Algorithm)을 제안한다.

본 논문의 구성은 다음과 같다. 제2장에서 기존의 예측 탐색 알고리즘에 대해 설명하고, 제3장에서는 제안한 ANBA를 위한 예측된 움직임 벡터의 설정 방법과 구현에 대한 방법을 논하며, 제4장에서는 실험 결과에 대하여

기존 방법과 같은 기준을 통해서 비교 분석하고, 제5장에서는 결론 및 향후 연구 과제를 제시한다.

2. 기존의 탐색 알고리즘

초창기의 고속 알고리즘들은 탐색 초기점을 (0, 0)으로 결정하는 경우가 많은데, 이는 많은 매크로블록의 움직임 벡터가 (0, 0) 주변이라는 통계를 이용한 것이다. 이에 속하는 알고리즘 중 다이아몬드 탐색은 SDS(Small Diamond Search)와 LDS(Large Diamond Search)이라는 2가지 탐색 패턴을 이용하였다. 다이아몬드 탐색 기법은 배경과 같이 거의 움직임이 없는 부분에서는 좋은 결과를 보여주지만 움직임이 큰 부분에서는 좋은 성능을 보여주지 못한다.

이러한 단점을 보완한 알고리즘이 MVFAST와 PMVFAST이다. MVFAST와 PMVFAST는 현재 부호화 블록의 움직임 벡터가 인접 블록의 움직임과 상관성이 높다는 점을 이용하여 탐색 시작점을 결정한다.

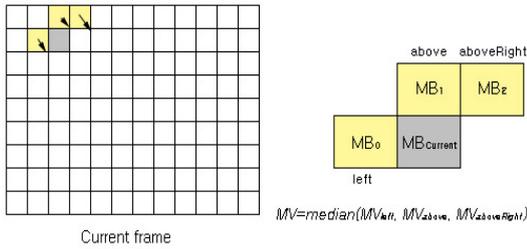
고속 움직임 벡터 탐색 기법인 MVFAST는 다이아몬드 탐색 기법을 개선한 방법으로 다이아몬드 탐색 기법에 비해 부호화 효율과 탐색 속도에 있어서 큰 성능 향상을 이루었다. 이 기법은 초기 탐색점으로 (0, 0)을 검사한다. 만약 (0, 0)에서의 SAD값이 임계치 T보다 작으면, 움직임 탐색을 중지하고 움직임 벡터를 (0, 0)으로 사용한다. 이를 움직임 탐색의 조기종결조건이라고 한다. 움직임 탐색이 단 하나의 탐색을 끝낼 수 있다는 점이 MVFAST 알고리즘의 가장 강력한 장점이다. 만약 움직임 벡터가 (0, 0)으로 정해지지 않는다면, 움직임 활동도 (Motion Activity)를 계산한다. MVFAST는 매크로블록의 움직임의 종류를 구분하기 위하여 움직임 활동도를 높음(high), 중간(medium), 낮음(low)의 세 가지로 정의한다. 움직임 활동도는 초기 탐색 중심점과 탐색 전략을 결정한다. 만약 움직임 활동도가 낮거나 중간이면 탐색 중심점은 원점이 된다. 그렇지 않으면 최소 SAD를 이끌어내는 집합 V에 속하는 벡터가 초기 탐색 중심점으로 선택된다. V는 인접한 3개의 블록의 움직임 벡터를 포함한다. 움직임 활동도가 낮거나 높다면 탐색 전략은 SDS가 된다. 움직임 활동도가 중간이라면 탐색 전략은 LDS로 한다.

그리고 현재 프레임의 블록들간의 시공간적인 상관관계를 반영하여 예측하는 PMVFAST는 그림 1과 같이 수평(left), 수직(above) 그리고 대각(aboveRight) 방향에 있는 MB_{left} , MB_{above} , $MB_{aboveRight}$ 들의 움직임 벡터를 참조하여 예측 움직임 벡터를 구한다[3-4]. 이와 같은 주변 움직임 벡터의 중간값을 사용하여 그 차를 인코딩하는

예측 알고리즘의 현재 부호화 블록의 예측 움직임 벡터 MV_p 는 식 (1)과 같이 이들 인접 블록 각각의 움직임 벡터 MV_{left} , MV_{above} , $MV_{aboveRight}$ 들의 중간값으로 한다.

$$MV_p = median(MV_{left}, MV_{above}, MV_{aboveRight}) \quad (1)$$

$$\begin{cases} d_x = median \sum_i MV_x^i \\ d_y = median \sum_i MV_y^i \end{cases} \quad i \in \{0, 1, 2\}$$



[그림 1] 기존 방식에서 참조하는 인접 매크로 블록
[Fig. 1] The Position Relation of the Current Block and the Reference Blocks.

PMVFAST는 기존의 많은 알고리즘들이 (0, 0) 벡터를 중심으로 탐색을 하는데 비해, 인접 블록의 움직임 벡터들의 중간값을 중심으로 탐색을 시작하며, 초기 탐색 종료율을 위한 임계값 결정 알고리즘을 채택하여 탐색 속도를 한층 더 최소화시켰으며 화질 또한 FS에 근접하는 결과를 보인다[7]. 즉, 탐색을 실시하여 빠르면서도 높은 화질을 보여주며, 영상에 따라 FS보다 높은 PSNR을 얻을 수 있는데 이는 움직임 벡터간의 상관성이 높아지면서 움직임 벡터를 부호화하는데 필요한 비트가 줄어들기 때문에 가능한 현상이다.

PMVFAST의 주 장점은 간소함이며, 탐색 패턴의 사용은 일반적으로 정수와 매우 규칙적이므로 하드웨어 구현을 보다 매력적으로 표현할 수 있다. 또한 PMVFAST의 다이아몬드 패턴은 초기 단계에서 보다 적은 점으로 탐색을 가능하게 하므로 속도 증가비를 적절하게 증대시킨다.

그러나 PMVFAST는 보다 많은 지향성으로 다음 단계의 탐색 방향이 정확하지 않게 되어 수행의 정밀도를 저하시키는 알고리즘 내부의 문제인 국부적인 최소에 빠지는 경향이 있다.

3. 제안한 움직임 추정 기법

제안하는 ANBA(Adaptive Neighboring-Block Search Algorithm)는 PMVFAST에 기초하고 있다. 그러나 기존의 PMVFAST는 현재 부호화 블록의 인접 블록의 움직임 벡터들의 중간값을 중심으로 탐색을 시작하는데 비해, 제안한 알고리즘은 이전 프레임과 현재 프레임간의 인접 블록에서 변화한 움직임 정보에 따른 후보 벡터와 부호화 블록의 이전 시점의 움직임 벡터를 적용하여 그 두 점을 비교하여 최소 정합 오차가 작은 점에서부터 탐색을 실시함으로써, 속도와 화질의 뚜렷한 개선이 가능하다. 즉, 탐색 시작점을 더욱 정확하게 예측할 수 있게 되면 속도면에서 뚜렷한 효과를 얻을 수 있을 뿐 아니라, 임계값과 조기 종료 조건을 줄일 수 있어 높은 화질을 이끌어 낼 수 있기 때문이다.

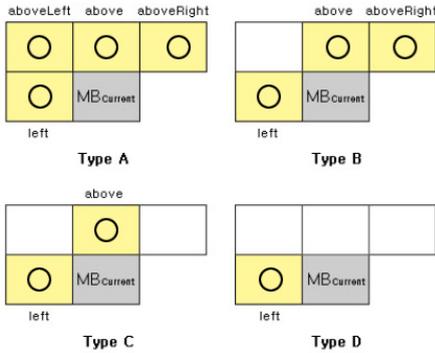
Yao Nie는 현재 부호화 블록의 정확한 움직임 벡터를 예측을 얻기 위해 두 개의 요인을 고려하였다[6].

- 1) 예측 움직임 벡터를 구하기 위해 움직임 벡터 주위의 인접 블록으로 구성된 지원 영역(ROS: Region Of Support)을 선택한다.
- 2) 예측 움직임 벡터를 계산하기 위한 알고리즘.

3.1 참조할 인접블록의 지원 영역

시간적인 관점에서, 현재 부호화 블록과 같은 위치의 참조 프레임의 블록이 시간적인 지원 영역 후보에 똑바로 선택되어야 한다. 더군다나 같은 참조 프레임으로부터 인접 블록들이 예측하기 위한 유망한 후보들을 제공하여야 한다. 그러므로 활용되는 시간적인 상관성은 이전 움직임 벡터 필드에 있어야 함으로 한정된 기억 장소에 사실상의 구현은 바람직하지 않는다. 그러므로 우리는 다만 이 방법에서 공간적인 상관성만 이용한다.

공간적인 관점에서, 비디오 프레임의 범위 내에서 모든 블록들은 래스터스캔 과정에서 움직임 벡터들의 인접 블록들은 현재 부호화 블록에서 바로 이웃되는 left, above, above-left와 above-right가 참조할 대상이 된다. 이외의 가까운 위치의 블록들은 현재 부호화 블록과 상관성이 적거나 예측 신뢰성이 떨어진다. 그 이상의 블록들의 사용은 더 많은 복잡한 계산이 필요하므로 공간적인 지원 영역은 그림 2와 같이 4개의 신뢰성 있는 시나리오로 인접 블록들을 정의한다.



[그림 2] 지원 영역(ROS)의 4가지 유형
[Fig. 2] Four Types of Region of Support

그림 2에서 Type A는 4개의 인접 블록 모두를 사용하는 것이며, Type B는 움직임 벡터들의 차등 부호화를 위한 H.263의 국제 표준으로 채용된 예측되는 지원 영역이다. Type C는 2개의 직접적으로 인접하는 블록으로 구성되며, Type D는 현재 부호화 블록의 바로 왼쪽 이웃에 위치하는 1개의 블록만을 참조한다.

표 1은 3개의 실험 비디오에 대해 그림 2와 같은 지원 영역의 4가지 유형별로 각 블록에 대한 평균 PSNR과 FS를 1로 기준하여 상대적인 속도 향상에 대한 비교 결과를 나타내었다.

[표 1] 4가지 지원영역의 PSNR과 속도향상 비교 결과
[Table 1] Comparison of ROS of Simulation Video Sequences

Sequence	Format		Type A	Type B	Type C	Type D
News	CIF	PSNR	38.109	38.114	38.109	38.034
		Speed Up	39.2	39.3	39.3	38.4
Akiyo	CIF	PSNR	42.739	42.738	42.738	42.719
		Speed Up	41.7	41.4	41.5	41.5
Miss America	CIF	PSNR	38.981	38.988	38.992	38.663
		Speed Up	28.7	27.6	27.7	29.2

따라서 본 논문에서는 지역영역 중 가장 신뢰성이 있는 Type C와 같이 현재 부호화 블록의 이전 시점의 수직 블록과 수평 블록을 참조한다.

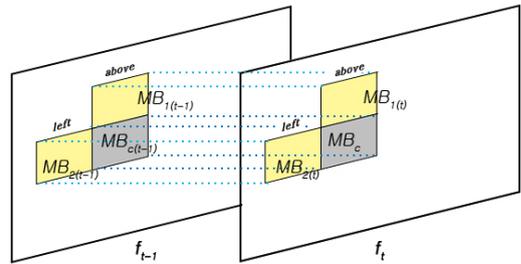
3.2 움직임 벡터 예측 후보의 설정

본 논문에서 제안하는 ANBA에서는 2개의 움직임 벡터 예측 후보를 구하여 이들의 SAD를 비교하여 가장 작은 값을 갖는 점으로 탐색 원점을 이동시켜 블록 정합을

수행한다.

영상의 움직임은 급격히 변화하기보다는 완만하게 변화한다는 연속성 제약 사항(motion continuity constraint)[2]과 공간적으로 인접한 매크로 블록들은 비슷한 속도로서의 같은 방향으로 움직인다는 점[3]에 근거하여 그림 3과 같이 이전 프레임의 같은 위치에 있는 블록의 수평, 수직의 인접 블록을 포함하여 3개의 평균 움직임 벡터를 참조한다.

첫 번째 움직임 벡터 예측 후보(1st Predictor) MV_{p1} 는 블록들은 수평, 수직으로만 움직인다는 점에 근거하여 그림 3에서 현재 부호화 블록의 인접 매크로블록인 $MB_{1(t)}$, $MB_{2(t)}$ 와 이전 프레임의 인접 매크로블록 $MB_{1(t-1)}$, $MB_{2(t-1)}$ 에서 결정한다.



[그림 3] 참조하는 이전 프레임의 같은 위치의 매크로 블록
[Fig. 3] MV of the Current Block might have more Relationship with the MVs of the Blocks around the Collocated Block in the Previous Frame

인접 매크로블록 MB_i (MB_i for $i \in \{1, 2\}$)의 움직임 벡터를 $MV_i = [\Delta x_i, \Delta y_i]$ 라고 할 때, 수직(above) 인접 블록 움직임 벡터는 현재 프레임과 이전 프레임에서 각각 $MV_{1(t)} = [\Delta x_{1(t)}, \Delta y_{1(t)}]$, $MV_{1(t-1)} = [\Delta x_{1(t-1)}, \Delta y_{1(t-1)}]$ 으로 움직임 벡터의 변화는 $[\Delta x_{1(t)} - \Delta y_{1(t-1)}, \Delta x_{1(t)} - \Delta y_{1(t-1)}]$ 가 된다.

이를 수평(left) 인접 블록에 적용하면 움직임 벡터의 변화는 $[\Delta x_{2(t)} - \Delta x_{2(t-1)}, \Delta y_{2(t)} - \Delta y_{2(t-1)}]$ 이다.

이들 두 블록의 평균값을 이전 프레임에서 현재 프레임으로 변화할 때의 움직임 Δx_p , Δy_p 는 식(2)와 같다.

$$\begin{cases} x_p = \frac{(\Delta x_{1(t)} - \Delta x_{1(t-1)}) + (\Delta x_{2(t)} - \Delta x_{2(t-1)})}{2} \\ y_p = \frac{(\Delta y_{1(t)} - \Delta y_{1(t-1)}) + (\Delta y_{2(t)} - \Delta y_{2(t-1)})}{2} \end{cases} \quad (2)$$

따라서 현재 부호화 블록의 좌표가 (m, n) 이면, 첫 번째 후보 벡터 MV_{p1} 의 좌표는 $(m + \Delta x_p, n + \Delta y_p)$ 이다.

두 번째 후보 벡터(2nd Predictor) MV_{p2} 는 시간적 관점에서 $MV_{p2} = MV_{c(t-1)}$ 으로 현재 매크로 블록의 이전 프레임의 같은 위치에 있는 움직임 벡터이므로 좌표는 $(m + \Delta x_{c(t-1)}, n + \Delta y_{c(t-1)})$ 이다.

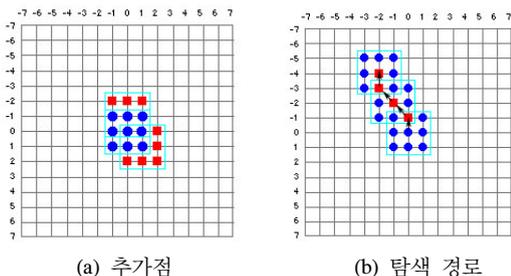
생성된 2개의 후보점 $(m + \Delta x_p, n + \Delta y_p)$ 와 $(m + \Delta x_{c(t-1)}, n + \Delta y_{c(t-1)})$ 의 SAD를 비교하여 가장 작은 값을 갖는 후보점을 탐색 초기점으로 설정한 후 블록 정합을 수행한다.

$$MV_p = \min SAD(MV_{p1}, MV_{p2}) \quad (3)$$

이전 프레임 블록의 움직임 벡터의 이동 거리를 고려한 이 방식은 연속된 프레임간 움직임 벡터의 변화가 커질 경우에 MV_p 가 탐색창의 경계 범위를 벗어날 수 있으므로 이러한 문제를 전처리할 필요가 있다.

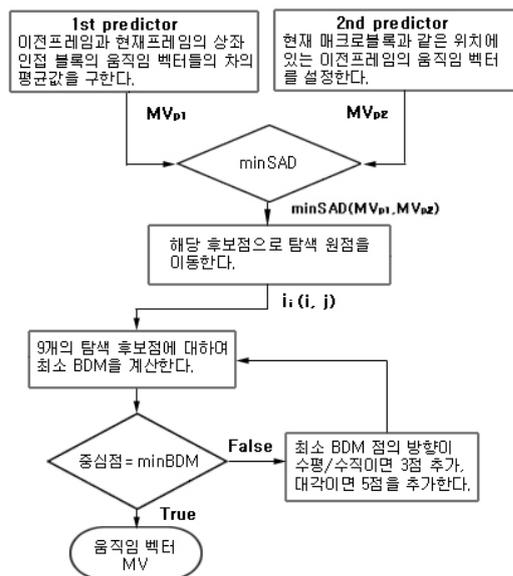
3.3 제안한 움직임 추정 알고리즘

이동된 탐색 원점에서 탐색 영역의 가장자리로 점진적인 탐색을 해 나갈 때 탐색 영역의 중심에 있는 일부 탐색점들을 블록 정합 대상에서 제외되어 화질의 저하를 초래하는 탐색 후보점 사이의 간격이 있는 패턴을 사용하지 않고, 그림 4와 같이 움직임 벡터의 대부분이 탐색 영역의 중심에 분포하는 특성을 고려하여, 중심점과 수평, 수직 방향의 거리가 1인 4점과 중심점으로부터 거리가 $\sqrt{2}$ 인 4개의 점으로 이루어진 3×3 화소를 가지는 정사각형 블록 패턴을 이용하여 모든 탐색점들이 블록 정합의 대상이 되도록 함으로써 움직임 추정의 성능을 높인다.



[그림 4] 3×3 정사각형 블록 패턴의 추가점 및 탐색 경로
[Fig. 4] Search Path of 3×3 Block Pattern

그림 5는 제안한 ANBA의 순서를 간단히 나타낸 흐름도로, ANBA에서는 두 개의 후보점에서 SAD를 비교하여 더 작은 값을 갖는 MV_p 만큼 블록의 중앙점에서부터 이동시킨 탐색점을 중심으로 추가된 8개를 포함한 9개의 탐색 후보점에 대하여 다음과 같은 알고리즘을 수행하여 움직임 벡터를 추정한다.



[그림 5] ANBA 움직임벡터 추정방식의 흐름도
[Fig. 5] Flowchart of Motion Vector Estimation of ANBA

1단계 : 현재 부호화 블록과 같은 위치에 있는 이전 프레임의 인접 블록들의 움직임 벡터 $MV_{i(t-1)}$ 와 현재 부호화 블록의 인접 블록들의 움직임 벡터 $MV_{i(t)}$ 로부터 현재 부호화 블록의 예측된 움직임 벡터 MV_{p1} 를 얻는다.

2단계 : 현재 부호화 블록의 이전 시점의 움직임 벡터 $MV_{c(t-1)}$ 인 MV_{p2} 와 1단계에서 구한 MV_{p1} 와 같은 벡터를 갖는 2개의 점을 서로 비교하여 최소 정합 오차를 갖는 점으로 탐색 원점을 이동한다.

3단계 : 이동된 탐색 원점을 중심으로 3×3 화소를 가지는 정사각형 블록 탐색 패턴 내에 존재하는 9개의 탐색점에 대하여 최소 정합 오차를 계산한다. 이때 중심점이 최소 정합 오차면 5단계로 가고, 아니면 최소 정합 오차를 가지는 점을 중심으로 하여 4단계로 간다.

4단계 : 이전 단계에서 발견된 최소 정합 오차를 가진 점을 정사각형 블록의 중심점으로 지정하여 새로 계산된 최소 정합 오차가 중심점에 위치하면 5단계로 가고, 아니

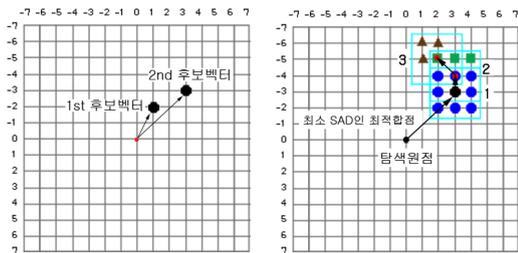
면 4단계를 반복한다.

- a) 만약 이전의 탐색 과정에서 최소 정합 오차를 갖는 점이 수평 또는 수직 방향이라면 새로운 3개의 탐색점을 추가하여 최소 정합 오차를 갖는 점을 중심점으로 하여 움직임 벡터를 추정하게 한다.
- b) 만약 이전의 탐색 과정에서 최소 정합 오차를 갖는 점이 대각 방향이라면 새로운 5개의 탐색점을 추가하여 최소 정합 오차를 갖는 점을 중심점으로 하여 움직임 벡터를 추정하게 한다.

단, 탐색 영역($w=\pm 7$)을 벗어나는 모든 점들은 무시한다. 매 단계마다 최소 정합 오차는 재정의되고, 최소 정합 오차를 갖는 점이 이전 탐색 단계에서 최소 정합 오차 점과 같다면 5단계로 이동하고 그렇지 않으면 4단계를 반복한다.

5단계 : 탐색을 중단하고 이전 단계에서 구한 최소 정합 오차를 가진 점을 최종적으로 움직임 벡터로 결정한다.

제안된 ANBA는 인접 블록간의 움직임 정보로부터 블록 정합을 위한 탐색 원점의 후보를 구함으로써 탐색 대상 영역이 줄어들게 되어 움직임 추정에 대한 계산량이 상대적으로 많이 줄어들게 된다. 즉, 후보점을 중심으로 상대적으로 축소된 탐색 영역 내에서 미소 변위 추정을 행하게 된다. 또한 이전 프레임에서 추정된 움직임 정보를 이용함으로써 효율적으로 더 신뢰할 수 있는 움직임을 추정할 수 있다는 장점이 있다.



[그림 6] 블록 정합 수행 과정
[Fig. 6] Implementation of Proposed Algorithm

그림 6의 (a)는 2개의 후보 벡터의 예이며, (b)는 ANBA의 탐색 수행의 예로서, 2개의 후보 벡터중에서 최소 SAD인 최적합점으로 탐색 원점을 이동하여 블록 정합을 수행하는 과정을 각각 보인 것이다. ANBA는 탐색 영역이 벗어나는 모든 점을 무시하고 중간에 탐색을 멈추는 방법을 제시하였는데 중간에 멈출 수 있는 조건은

현재 탐색 단계에서 최소 정합 오차를 갖는 점의 위치가 이전 탐색 단계의 최소 정합 오차를 갖는 점의 위치와 같다면 중간에 탐색을 멈추고 최종적인 움직임 벡터로 결정한다.

4. 실험 결과

실험은 MPEG-4 VM 인코더를 이용하여 이루어졌으며, 실험에 사용한 영상은 CIF(352× 288)이며, 16×16 화소를 사용하였다. 탐색 영역의 범위는 각각 ± 7 로 설정하고, Bitrate 1024k에 대해 실험하였다.

제안된 ANBA의 성능을 평가하기 위하여 움직임이 아주 적은 News, 움직임이 조금 있는 Miss America, 그보다 움직임이 있는 Foreman, 움직임이 큰 Stefan의 4개의 실험 영상에 대해 각각 80프레임씩을 대상으로 실험하였다.



[그림 7] 실험 영상의 첫 번째 프레임
[Fig. 7] 1st Frames of Experimental Video Sequences

비교 탐색 기법으로는 FS, DS, BBGDS와 예측 탐색 알고리즘인 MVFAST, PMVFAST, 그리고 제안한 ANBA를 사용하였다.

블록 정합의 정도를 평가하기 위해 대표적인 정합 기준인 평가 함수(cost function)로 영상 화질의 품질을 평가하기 위한 식 (4)와 같은 평균 제곱 오차(MSE: Mean Squared Error), 식 (5)의 평균 절대값 오차(MAD: Mean Absolute Difference)와 정합 오차 측정 함수로는 식 (6)의 절대값 오차의 합(SAD: Sum of Absolute Difference)을 이용하였다. 또한 제안하는 ANBA의 성능 향상을 측정하기 위해 블록 당 탐색점의 개수를 기존 방법들과 비교하

였다.

$$MSE(i, j) = \left(\frac{1}{N^2}\right) \sum_{k=1}^N \sum_{l=1}^N [I_t(k, l) - I_{t-1}(k+i, l+j)]^2 \quad (4)$$

$$MAD(i, j) = \left(\frac{1}{N^2}\right) \sum_{k=1}^N \sum_{l=1}^N [I_t(k, l) - I_{t-1}(k+i, l+j)] \quad (5)$$

$$SAD(i, j) = \sum_{k=1}^N \sum_{l=1}^N [I_t(k, l) - I_{t-1}(k+i, l+j)] \quad (6)$$

여기서 N은 영상의 가로와 세로의 각각의 크기이며, $I_t(k, l)$ 은 원영상의 화면을 나타내고, $I_{t-1}(k+i, l+j)$ 은 움직임 추정 화면을 나타내며, 이들 정합 기준들은 최소 값을 가지는 위치를 움직임 벡터로 결정한다.

그리고 화질의 평가를 위한 PSNR은 식 (7)과 같다.

$$PSNR = 10 \log_{10} \left(\frac{255^2}{MSE} \right) \quad (7)$$

실험 영상에 대한 실험 결과는 표 2와 표 3, 그림 5에 각각 나타내었다.

이러한 결과를 본다면, 제안한 ANBA가 FS에 비해 평균 탐색점 수가 약 95% 정도 감소하였고, 탐색 속도면에서는 약 18~20배 정도의 성능 향상을 나타내었으며, 기존의 탐색 기법에 비해 탐색점 수가 감소하여 탐색 속도가 향상되었다. 또한 화질면에서 FS를 제외한 기존의 방식보다 우수함을 볼 수 있다. 즉, 평균 PSNR 값의 4개의 실험 영상에 대한 평균이 제안한 방식이 33.197[dB]로 나타나 33.367[dB]인 FS 다음으로 우수하였다.

그리고 실험에 사용한 영상별로 볼 때, 움직임이 아주 작은 News 영상은 FS와 비교하였을 때 탐색점 수를 95% 감소시키면서도 PSNR 값이 0.057[dB] 차이로 거의 화질의 저하가 없었으며, FS를 제외한 다른 기법에 비해 PSNR 값이 가장 우수하였다. 그보다 움직임이 조금 있는 Miss America 영상의 PSNR 값에서는 제안된 기법이 FS보다 0.069[dB] 정도 떨어지나 탐색점 수에서 약 18배 정도의 성능 향상을 나타내었다. 그보다 움직임이 있는 Foreman 영상에서는 FS가 제안된 기법보다 0.105[dB] 정도 우수하나 탐색점 수를 18배 이상 사용되었고, 움직임이 큰 Stefan 영상에서는 FS가 0.437[dB]정도가 좋으나 약 25배 정도의 많은 탐색점 수를 사용하여 속도면에서는 제안한 ANBA가 우수함을 보였다.

[표 2] 각 실험 영상에 대한 성능 비교 평가 함수의 결과 비교 값
[Table 2] Comparison of MAD and PSNR of the Search Algorithm

Sequence	CIF							
	News		MissAmerica		Foreman		Stefan	
Method	MAD	PSNR	MAD	PSNR	MAD	PSNR	MAD	PSNR
FS	1.042	36.225	1.930	39.095	2.817	32.456	6.715	25.693
DS	1.045	36.156	2.018	38.762	2.904	32.175	7.717	24.655
BBGDS	1.045	36.147	1.959	39.025	2.850	32.282	8.202	24.193
MVFAST	1.045	36.156	2.018	38.762	2.904	32.175	7.717	24.657
PMVFAST	1.045	36.153	1.959	39.025	2.832	32.302	7.202	25.215
ANBA	1.044	36.168	1.959	39.026	2.834	32.351	7.030	25.256

[표 3] 각 실험영상에 대한 제안된 기법과 기존기법의 탐색 속도 측면
[Table 3] Comparison of Search Point Number and Speedup of the Search Algorithm

Sequence	CIF							
	News		MissAmerica		Foreman		Stefan	
Method	탐색수	SpUp	탐색수	SpUp	탐색수	SpUp	탐색수	SpUp
FS	204.28	1.000	204.28	1.000	204.28	1.000	204.28	1.000
DS	12.52	16.316	16.40	12.456	15.90	12.848	16.78	12.174
BBGDS	9.82	20.802	11.37	17.967	13.67	14.944	14.52	14.069
MVFAST	10.12	20.186	12.18	16.772	12.91	15.823	12.77	15.997
PMVFAST	9.84	20.706	11.39	17.935	12.67	16.123	11.52	17.232
ANBA	9.82	20.802	11.33	18.030	12.14	16.827	11.46	17.854

이상의 실험 결과에 의하면, PSNR 측면에서 FS를 제외한 모든 기법에 비해 제안된 ANBA가 가장 우수한 것으로 나타났다. 속도면에서는 움직임이 크면 클수록 제안된 방식이 우수하였으며, 기존의 낮은 계산 복잡도를 가진 예측 탐색 알고리즘인 MVFAST와 PMVFAST를 비교하였을 경우에도 제안된 ANBA가 약 0.01~0.64 [dB] 정도의 화질 향상과 낮은 계산 복잡도로 인하여 약 1.02~1.07배의 속도 향상을 보였다.

5. 결론

본 논문에서는 기존의 알고리즘보다 정확하게 탐색 시작점을 예측함으로써 효율적인 부호화를 가능하게 하는 움직임 추정 기법인 ANBA를 제안하였다.

정량적인 측면에서 MVFAST의 경우 추가적인 메모리가 필요하지 않지만 전역 탐색 기법에 비해 낮은 PSNR을 보이고, PMVFAST는 이전 프레임에서의 움직임 벡터를 저장하기 위한 메모리가 필요하지만 전역 탐색 기법과 거의 유사한 PSNR과 높은 속도 개선 효과를 보인다.

본 논문에서는 기존의 예측 탐색 알고리즘이 예측된 움직임 벡터로 탐색 원점을 이동시켜 탐색을 수행함으로써 블록 내에 공존하는 최소 정합 오차를 가지는 탐색점이 제외되어 화질 저하를 초래하는 문제점을 개선하기 위해 이전 움직임 정보에 의해 예측된 움직임 벡터와 탐색 영역의 중심에 가깝게 정의한 탐색 구간별 후보 벡터 중에서 가장 작은 SAD 값을 갖는 점을 탐색 원점으로 하여 탐색을 수행하는 알고리즘을 제안하였다. 실험의 결과를 보면 제안된 ANBA가 화질면에서는 전역 탐색에 근접한 우수한 성능을 보였으며, 기존의 예측 탐색 기법과 거의 같은 탐색점 수를 사용하면서도 화질면에서는 나은 결과를 보였다.

움직임 추정시 본 논문에서 제안한 예측 탐색 기법인 ANBA를 사용하여 움직임 추정을 한다면 보다 빠르게 움직임 벡터를 찾을 수 있을 것이며, 고속 블록 정합 방법에 적합한 탐색 패턴을 도출하고, 탐색 패턴을 적응적으로 변화시키면 다른 고속 블록 정합 방법들보다 탐색점 수를 감소시키고, 우수한 보상 결과를 얻을 수 있을 것으로 기대된다. 또한 일정한 시간 간격을 두고 연속적으로 추출되는 움직임 벡터의 크기를 일정하게 조절할 수 있는 시간적인 동기화에 대한 연구도 보강될 필요가 있다.

움직임 벡터의 예측 과정에서 발생할 수 있는 예외적인 요소를 모두 고려하여 가장 적응적인 현재 프레임의 매크로 블록 내에서 움직임 벡터를 구간별로 탐색하므로

본 연구에서 제안한 탐색 기법을 사용하여 움직임 예측을 한다면 보다 빠르게 움직임 벡터를 찾을 수 있을 것이며, 또한 동영상 압축의 성능을 향상시키고 우수한 보상 결과를 얻을 수 있을 것으로 기대된다.

References

- [1] T. Koga, K. Inuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated Interframe Coding for Video Conferencing", in Proc. National Telecommunications Conf., New Orleans, LA, pp.G5.3.1-G5.3.5, Nov. 1981.
- [2] Vincent S., S. Hwang, "Tracking Feature Points in Time-Varying Image using an Opportunistic Selection Approach", Pattern Recognition, Vol. 22, pp.247-256, 1989.
- [3] Oscar T.-C. Chen, "Motion Estimation Using a One-Dimensional Gradient Descent Search", IEEE Transactions on Circuits & System for Video Tech., Vol. 10, No. 4, pp.608-616, June 2000.
- [4] P. I, Hosur, K. K. Ma, "Motion Vector Field Adaptive Fast Motion Estimation", in Proc, the Second Int. Conf. Inf., Commun., Signal Process, Dec. 2003.
- [5] A. M. Tourapis, O. C. Au, M. L. Liou, "Highly efficient predictive zonal algorithms for fast block-matching motion estimation", IEEE Transactions on Circuits & System for Video Tech., Vol. 12, No. 10, pp.934-947, Oct. 2002.
- [6] Y. Nie, K-K. Ma, "Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation", IEEE Transactions on Image Processing, Vol. 11, No. 12, pp.1442-1449, Dec. 2002.
- [7] F. Moschetti, M. Kunt, E. Debes, "A Statistical Block-Matching Motion Estimation", IEEE Transactions on Circuits & System for Video Tech., Vol. 13, No. 4, pp.417-431, Apr., 2003.
- [8] W. A. C. Fernando, "Sudden Scene Change Detection in Compressed Video using Interpolated Macroblocks in B-frames", Multimedia Tools and Applications, Vol. 28, No.3, pp.301-320, May, 2006.
- [9] Goela, N., Wilson, K., Feng Niu, "An SVM Framework for Genre-Independent Scene Change Detection", Multimedia and Expo, 2007 IEEE International Conference on, pp.532-535, Jul., 2007
- [10] X Yi, N Ling, "Fast Pixel-Based Video Scene

Change Detection", Circuits and Systems, IEEE International Symposium on, Vol.4, pp.3443-3446, May, 2005.

- [11] Shih-Hao Wang, Shih-Hsin Tai, Tihao Chiang, "A Low-Power and Bandwidth-Efficient Motion Estimation IP Core Design Using Binary Search", Circuits and Systems for Video Technology, IEEE Transactions on, pp.760-765, May 2009.
-

곽 성 근(Sung-Keun Kwak)

[정회원]



- 1980년 2월 : 연세대학교대학원 전자공학과 (공학석사)
- 2004년 8월 : 아주대학교대학원 컴퓨터공학과 (공학박사)
- 1980년 3월 ~ 2010년 2월 : 인천전문대학 컴퓨터애니메이션과 교수
- 2010년 3월 ~ 현재 : 인천대학교 디자인학부 교수

<관심분야>

컴퓨터그래픽스, 동영상압축, 컴퓨터애니메이션