예측 방향성 탐색 알고리즘을 이용한 새로운 블럭 정합 움직임 추정 방식

서 재 수† · 남 재 열†† · 곽 진 석††† · 이 명 호†††

요 약

본 논문은 블럭 정합 움직임 추정을 위한 새로운 방식을 제안하고자 한다. 일반적으로 영상 시퀀스는 시간적으로 많은 상관성을 갖고 있다. 따라서 이전 프레임 블록들로부터 많은 정보를 얻을 수 있다면 현재 블럭의 움직임 추정에 대한 성능을 높일 수 있고 또한 탐색 횟수를 줄임으로써 계산 복잡도를 크게 줄일 수 있다. 이러한 영상의 시간적인 특성과 움직임 벡터의 가운데 중심적인 특성(center-biased property)을 이용한 새로운 블럭 정합 움직임 추정 방식을 제안하고자 한다. 본 논문에서 제안된 방식은 영상의 시간적인 상관성을 이용하여 현재 프레임 블럭과 같은 위치에 있는 이전 두 개의 프레임 블럭들에 대해 본 논문에서 정의한 방향성을 조사하게 된다. 만약 방향이 같다면 이전 프레임 블럭의 방향으로 처음의 탐색 위치를 이동하여 일정한 탐색 패턴에 따라서 탐색을 하게 되고, 만약 방향이 다르다면 ±2인 탐색 영역에 대해 전역 탐색을 하게 된다 실험 결과 제안된 방식은 기존의 대표적인 고속 탐색 방식들에 비해 PSNR(Peak-to-Signal Noise Ratio) 값에 있어서 평균적으로 1.5dB 개선되고 영상에 따라 최고 3.4dB 정도 우수한 결과를 나타낸다. 또한 탐색 횟수에서는 다른 고속 탐색 알고리즘보다 평균 20%를 줄일 수 있었고, 정확한 움직임 벡터를 찾는 비교에 있어서도 월등히 우수한 결과를 나타내고 있다. 제안된 방식은 정량적인 결과뿐만 아니라 부호화후 복호화한 영상의 화질에 있어서도 다른 고속 탐색 알고리즘보다 월등히 우수한 화질을 제공한다.

A New Block Matching Motion Estimation using Predicted Direction Search Algorithm

Jae-Soo Seo† · Jae-Yeal Nam†† · Jin-Suk Kwak††† · Myoung-Ho Lee†††

ABSTRACT

This paper introduces a new technique for block matching motion estimation. Since the temporal correlation of the image sequence, the motion vector of a block is highly related to the motion vector of the same coordinate block in the previous image frame. If we can obtain useful and enough information from the motion vector of the same coordinate block of the previous frame, the total number of search points used to find the motion vector of the current block may be reduced significantly. Using that idea, an efficient predicted direction search algorithm (PDSA) for block matching algorithm is proposed.

Based on the direction of the blocks of the two successive previous frames, if the direction of the two successive blocks is same, the first search point of the proposed PDSA is moved two pixels to the direction of the block. The searching process after moving the first search point is processed according to the fixed search patterns. Otherwise, full search is performed with search area ±2.

Simulation results show that PSNR values are improved up to the 3.4dB as depend on the image sequences and improved about 1.5dB on an average. Search times are reduced about 20% than the other fast search algorithms. Simulation results also show that the performance of the PDSA scheme gives better subjective picture quality than the other fast search algorithms and is closer to that of the FS(Full Search) algorithm.

※ 본 연구는 97 한국과학재단 특정기초연구(97-0100-0201-3)의 지원으로 수행되었습니다.

† 정 회 원 : (주)웹컴정보시스템
†† 정 회 원 : 계명대학교 컴퓨터·전자공학부 교수
††† 정 회 원 : 한국전자통신연구원 무선·방송 기술 연구소

논문접수 : 1999년 12월 29일, 심사완료 : 2000년 2월 12일

1. 서 론

요즈음 폭발적으로 성장하고 있는 인터넷망, 이동 통신망 등에서는 사용자가 대화형으로 쉽게 접근, 편집, 처리 등과 같은 다양한 기능을 제공할 수 있는 새로운 멀티미디어 서비스 및 대화형 영상 서비스에 대한 요구가 날로 증가하고 있다. 이러한 분야에서 MPEG-4 기술은 필수적인 핵심기술로 이용될 수 있을 것이다. 특히 차세대 이동 통신 기술로 각광 받고 있는 IMT-2000용 단말기에 MPEG-4 방식의 핵심 기술들을 이용한다면 그 시장 잠재력과 효용성은 막대할 것이다 그러나 MPEG-4 방식의 기술들을 IMT-2000용으로 이용하기 위해서는 비디오 코덱의 칩 개발이 필요하며 이때 비디오 코덱 압축에 있어서 계산량의 많은 부분을 차지하는 움직임 추정을 위한 효율적이고 간단한 알고리즘이 개발되어야 한다.

일반적으로, 블럭 정합 움직임 추정 기법에서 프레임은 $N \times N$ 화소의 블럭들로 구성되고, 각 블럭은 탐색 과정에서 최대 w 화소의 이동 범위를 갖게 되어 $(N+2w)$의 크기에 포함되는 모든 점들을 조사하게 된다. 또한 블럭 정합 움직임 추정 기법에서 연속된 영상의 움직임 벡터들은 80%정도가 가운데를 중심으로 ±3픽셀 이내에 포함되고 거의 움직임이 없는 영상의 경우 90~99%정도가 ±3픽셀 이내에 포함된다. 따라서 연속된 영상의 움직임 벡터는 가운데 중심적인 특성(center-biased property)을 갖는다[1-3].

대표적인 블럭 정합 움직임 추정 기법으로는 전역 탐색 알고리즘 (full search algorithm)이 있다. 전역 탐색 알고리즘은 현재 프레임 블럭과 같은 좌표를 갖는 이전 프레임 블럭의 위치를 중심으로 탐색 영역에 포함되는 모든 점들을 조사하여 최적의 움직임 벡터를 찾는다. 탐색 영역에 포함되는 모든 점들을 탐색 할 경우 $(2w+1)^2$개의 탐색 횟수를 갖게 되고 ±15화소의 탐색 영역을 갖는다면 961번의 블럭 정합 탐색을 하게 된다. 따라서, 전역 탐색 알고리즘은 움직임 추정 기법들 중에서 최적의 움직임 벡터를 찾는 장점이 있는 반면에 탐색 영역에 포함되는 모든 점들을 탐색하기 때문에 다른 고속 탐색 알고리즘들에 비해서 계산량이 너무 많다는 단점이 있다. 따라서 이러한 전역 탐색의 과도한 계산량을 줄일 수 있는 고속 탐색 알고리즘들이 많이 연구되었다[4-11].

이러한 고속 탐색 알고리즘들 중에서 대표적인 방식은 3단계 탐색 알고리즘 (three-step search algorithm)이다[4]. 3단계 탐색 알고리즘은 일정한 패턴에 따라 27개 점들에 대해서 탐색을 하기 때문에 계산량을 줄일 수 있는 가장 간단하면서도 효율적인 알고리즘이다. 그러나 3단계 탐색 알고리즘의 경우 계산량을 줄이고 간단하다는 장점은 갖고 있으나 모든 영상들에 대해서 고정된 탐색 패턴을 적용하기 때문에 움직임이 거의 없는 영상의 경우 문제가 없지만 움직임이 많은 영상의 경우 첫 번째 탐색이 잘못 되었을 경우 local optima에 빠질 수 있는 단점을 갖고 있다 또한 이러한 3단계 탐색 알고리즘을 개선하기 위해서 4단계 탐색 알고리즘 (four-step search algorithm)이 제안되었고[5], 이 방식은 3단계 탐색 알고리즘 보다 첫 번째 탐색 범위를 줄임으로써 local optima에 빠지는 단점을 보완하려고 하였다. 그러나 이 방식 또한 모든 영상에 대해서 일정한 탐색 패턴을 적용함으로써 움직임이 많은 영상의 경우 그 효율이 떨어짐을 볼 수 있다 또한 움직임 벡터의 가운데 중심적인 특성을 이용한 다이아몬드 탐색 알고리즘 (unrestricted center-biased diamond search algorithm)이 제안되었다[6]. 이 알고리즘은 일반적인 영상 시퀀스의 움직임 벡터가 ±3 화소 이내에 포함된다는 특성을 이용함으로써 움직임 벡터를 찾는 확률을 높이려는 방식이다. 따라서 이 방식은 움직임 벡터가 가운데 중심으로 분포되어 있는 영상의 경우에 적합하다. 하지만 급격한 움직임이 있는 영상의 경우에는 움직임 벡터가 ±3 화소의 범위를 벗어나는 경우가 많기 때문에 움직임이 많은 영상의 경우에는 적합하지 않다.

최근에는 기존의 고속 탐색 알고리즘의 단점을 개선하기 위해 이전에 탐색된 움직임 벡터들의 정보를 이용하는 예측 탐색 알고리즘이 (Prediction Search Algorithm) 제안되었다[7]. 이 방식은 인접한 이전 블럭들의 움직임 벡터 정보를 이용함으로써 보다 정확한 움직임 벡터를 찾으려고 하였다 그러나 이 방식은 인접한 블럭의 움직임 벡터들의 상관성이 떨어질 경우 압축 성능이 현저히 떨어진다는 단점이 있다[7, 11].

결국 모든 고속 탐색 알고리즘들은 계산량을 줄이기 위해서 탐색 영역에 포함되는 특정한 몇몇 점들만 조사하여 움직임 벡터를 찾기 때문에 국부적인 탐색을 하게 된다. 즉, 현재의 탐색 단계에서 몇몇의 점들이 가장 작은 SAD(Sum Absolute Differences) 값을 갖는 점과 비슷한 SAD 값을 갖게 되면 다음 탐색 단계의 탐색 방향이 정확하지 않게 되어 국부적인 탐색을 하

게 되고, 결국에는 압축된 영상의 성능이 저하되는 단점이 있다.

따라서, 본 논문에서는 계산량과 압축된 영상의 성능 사이의 상호 보완적인 관계, 움직임 벡터의 가운데 중심적인 특성, 연속된 프레임에서 움직임 벡터들의 높은 시간적인 상관성 등을 이용하여 다른 고속 탐색 알고리즘보다 부족한 정보로 인해서 야기되는 국부적인 탐색을 방지하면서 압축 성능을 향상시킬 수 있는 예측 방향성 탐색 알고리즘을 제안하고자 한다.

본 논문의 구성은 다음과 같다. 제 2절에서는 본 논문에서 제안하는 예측 방향성 탐색 알고리즘을 설명하고, 제 3절에서는 PSNR, MSE(Mean Square Error), 탐색 횟수, 전역 탐색 알고리즘의 블럭과 비교하여 같은 움직임 벡터를 갖는 블럭의 개수 등과 같은 기준을 통해서 다른 탐색 알고리즘과 성능을 비교 분석하였다. 제 4절에서는 결론을 내리고 현재 연구가 진행중인 움직임 추정 알고리즘에 대해 간단히 소개한다.

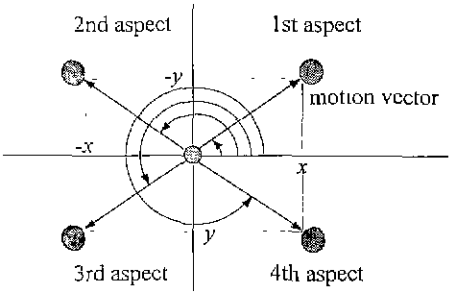
2. 예측 방향성 탐색 알고리즘

본 논문에 제안하는 예측 방향성 탐색 알고리즘은 현재 블럭의 움직임 벡터를 찾기 위해서 현재 프레임 블럭과 같은 위치에 있는 이전 두 개의 연속된 프레임 블럭들의 움직임 벡터를 이용한다. 먼저 이전 프레임에 속한 블럭들의 방향성을 검사하고 그 결과에 따라 처음 탐색 위치를 적절히 보정하여 움직임 벡터를 찾게된다. 그러한 블럭의 방향은 움직임 벡터의 성분을 갖고 계산되는 각도에 따라서 결정된다. 움직임 벡터를 이용하여 각도를 계산하는 과정은 식 (1)에 정의되어 있다.

$$Radian = asin\left(\frac{|y|}{\sqrt{(x^2+y^2)}}\right)$$
$$MVAngle = Radian \times 180 \quad (1)$$

여기서, *asin*은 arcsine 함수이고, *x*, *y*는 움직임 벡터의 *x*, *y* 성분이다. *MVAngle*은 (그림 1)에서의 1사분면 각도이다.

식 (1)에서 계산된 각도는 90 °보다 작은 값만을 갖는다. 따라서 움직임 벡터의 위치에 따른 최종 각도를 결정하게 되는데 그러한 관계를 (그림 1)에 나타내었다. 즉, 움직임 벡터의 위치에 따른 최종 각도를 계산하는 방식을 표시하는 것이다.



〈그림 1〉 움직임 벡터의 위치와 최종 각도사이의 관계

따라서 움직임 벡터의 최종적인 각도를 계산하기 위해서는 움직임 벡터의 위치 즉, 몇 사분면인가를 고려하여야 한다. 이러한 움직임 벡터의 위치에 따라서 최종적인 각도를 계산하는 과정은 <표 1>에 정의되어 있다.

〈표 1〉 움직임 벡터의 위치를 고려한 최종적인 각도 계산

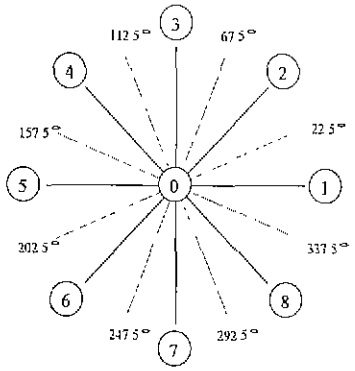
움직임 벡터의 위치	움직임 벡터의 최종 각도
1사 분면	*FMVAngle* = *MVAngle*
2사 분면	*FMVAngle* = 180 - *MVAgle*
3사 분면	*FMVAngle* = 180 - *MVAgle*
4사 분면	*FMVAngle* = 360 - *MVAgle*

<표 1>에서 계산된 움직임 벡터의 최종적인 각도를 바탕으로 결정되는 이전 프레임 블럭들의 방향은 <표 2>와 (그림 2)에 정의되어 있다.

<표 2>와 (그림 2)에서 같이 본 논문에서 9개의 방향만을 고려한 이유는 일반적인 영상 시퀀스에 있어서 움직임이 수평, 수직 또는 대각선 방향을 고려하더라도 충분히 영상 시퀀스에 대한 움직임을 추정 할 수 있기 때문이다. 9개의 방향보다 작은 수의 방향을 고려한다면 영상에 대한 정확한 움직임을 추정 할 수 없어서 정확한 방향 구분이 힘들게 된다. 또한 너무 많은 방향을 고려한다면 영상의 방향이 너무 세분화되어서 계산량이 많아지는 단점이 있고 또한 움직임 방향이 너무 세분화되었을 경우 영상의 객체에 대한 방향성이 너무 세분화되기 때문에 두 개의 연속된 프레임을 이용하는 특성을 이용하지 못하고 오히려 방향을 9개 고려했을 때 보다 성능을 저하시키는 결과를 가져오기 때문에 본 논문에서는 9개의 방향만을 고려하게 되었다.

〈표 2〉 움직임 벡터의 각도와 블럭의 방향과의 관계

FMVAangle 의 각도	방 향
FMVAngle = 0 °	⓪ 방향
0 ° < *FMVAngle* < 22.5 ° and 337.5 ° ≤ *FMVAngle* < 360 °	① 방향
22.5 ° ≤ *FMVAngle* < 67.5 °	② 방향
67.5 ° ≤ *FMVAngle* < 112.5 °	③ 방향
112.5 ° ≤ *FMVAngle* < 157.5 °	④ 방향
157.5 ° ≤ *FMVAngle* < 202.5 °	⑤ 방향
202.5 ° ≤ *FMVAngle* < 247.5 °	⑥ 방향
247.5 ° ≤ *FMVAngle* < 292.5 °	⑦ 방향
292.5 ° ≤ *FMVAngle* < 337.5 °	⑧ 방향



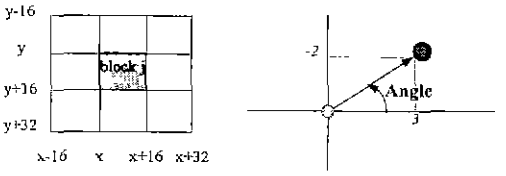
(그림 2) 〈표 2〉의 그래프적인 표현

따라서 본 논문에서 제안된 예측 방향성 탐색 알고리즘은 이전 두 개의 연속된 프레임 블럭들의 방향을 조사하여 보다 정확한 초기 탐색 위치를 얻기 위해서 현재 프레임 블럭과 같은 위치에 있는 두 개의 연속된 프레임 블럭들의 방향이 같은 경우와 방향이 다른 경우를 고려하게 된다. 이렇게 두 개의 연속된 프레임 블럭을 이용하는 이유는 대부분의 동영상의 경우 두 개의 연속된 프레임 블럭의 방향은 많은 연관성을 갖고 있기 때문이다. <표 3>은 대표적인 영상 시퀀스에 대해 연속된 두 프레임 간의 같은 좌표를 갖는 두 개 블럭의 방향성을 조사한 결과를 보여준다.

따라서, 제안된 예측 방향성 탐색 알고리즘은 이러한 이웃한 블럭들의 움직임 벡터의 높은 상관성을 이용하여 현재 블럭의 움직임 벡터를 찾기 위해서 같은 좌표를 갖는 2개의 이전 프레임 블럭들의 방향을 고려하게 된다. 즉, (그림 3), (그림 4)와 같이 현재 i_{th} 프레임의 j_{th} 블럭의 움직임 벡터를 탐색하기 위해서 $(i-2)_{th}$ 프레임과 $(i-1)_{th}$ 프레임의 j_{th} 블럭들의 방향을 이용한다.

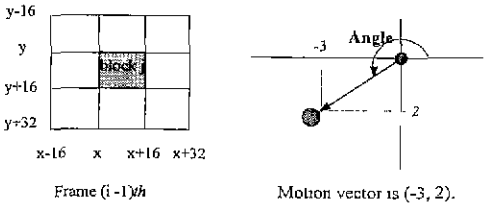
〈표 3〉 연속된 두 프레임의 같은 좌표를 갖는 두 개의 블럭이 같은 방향을 갖는 개수(총 396개)

Image Sequences	같은 방향을 갖는 블럭의 개수	Percentage
Coastguard	354.0	89.39%
Foreman	306.9	77.50%
Mother	381.6	96.36%
News	344.4	86.97%
Stefan	352.9	89.12%
Table	366.5	92.55%
평 균	351.1	88.65%



블럭의 각도는 37.43° 이고, 방향은 ②이다

(그림 3) 움직임 벡터의 각도와 블럭의 방향을 계산하는 예 I



블럭의 각도는 217.43° 이고, 방향은 ⑥이다.

(그림 4) 움직임 벡터의 각도와 블럭의 방향을 계산하는 예 II

만약 현재 프레임 블럭과 같은 좌표를 갖는 이전 두 개의 연속된 프레임 블럭들의 방향이 같다면 제안된 PDSA 방식의 첫 번째 탐색 위치를 블럭의 방향으로 2화소 이동하고, 2화소 이동한 탐색 위치를 기준으로 (그림 7)의 두 가지 탐색 패턴을 갖고 (그림 5)와 같은 탐색을 하게 된다 블럭의 방향이 다르다면 (그림 6)과 같이 ±2의 탐색 영역을 갖고 전역 탐색을 하게 된다. (그림 7)에서 Face pattern은 9개의 점들 중에서 가장 작은 SAD를 갖는 점이 상하좌우에 있을 경우 3개의

점을 추가하여 다음 단계의 탐색을 하는 탐색 패턴이고, Vertex pattern은 현재의 탐색 단계에서 가장 작은 SAD를 갖는 점이 대각 방향일 경우 5개점을 추가하여 다음 단계의 탐색을 하는 탐색 패턴이다 (그림 5)와 (그림 6)에서 첫 번째 탐색 위치(*first search point*)는 현재 블럭의 좌측 상단의 점을 의미한다 제안된 방식은 탐색 영역의 범위를 벗어나는 모든 점들은 무시하고 중간에 탐색을 멈추는 방법을 제시하였는데 중간에 멈출 수 있는 조건은 현재 탐색 단계에서 가장 작은 SAD를 갖는 점의 위치가 이전 탐색 단계에서 가장 작은 SAD를 갖는 점의 위치와 같다면 중간에 탐색을 멈추고 최종적인 움직임 벡터로 결정한다.

새로 제안된 PDSA 알고리즘을 요약하면 다음과 같다

단계 1 : 식 (1)과 <표 1>을 이용하여 움직임 벡터의 각도를 계산한다. 계산된 각도를 바탕으로 <표 2>를 이용하여 블럭의 방향을 결정한다.

단계 2 : 만약 이전 두 프레임의 같은 좌표를 갖는 2개 블럭들의 방향이 같다면 첫 번째 탐색 위치를 그 방향으로 2화소 이동하여 단계 3으로 이동하고, 만약 방향이 같지 않다면 단계 4로 이동한다

단계 3 : 첫 번째 탐색 위치를 3×3탐색 영역의 중심점으로 하는 9개의 탐색 점들을 갖고 탐색을 진행한다. 만약 9개의 점들 가운데 가장 작은 SAD를 갖는 점이 이전 단계의 탐색에서와 같은 점을 갖는다면, 즉 가장 작은 SAD를 갖는 점이 탐색 영역의 가운데 점인 (α, α)라면 단계 5로 이동, 그렇지 않으면 다음 과정을 반복한다.

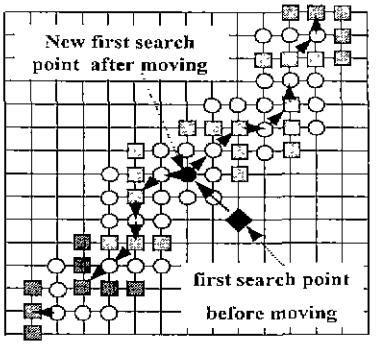
a) 만약 이전의 탐색 과정에서 가장 작은 SAD 갖는 점이 수평 또는 수직 방향이라면 [i.e., (α-1, α-1), (α+1, α-1), (α-1, α+1), or (α+1, α+1)], (그림 7)의 (a)와 같은 탐색 패턴을 갖고 새로운 3개의 탐색 점을 추가하여 가장 작은 SAD를 갖는 점을 가운데 점으로 하여 움직임 벡터를 추정하게 된다.

b) 만약 이전의 탐색 과정에서 가장 작은 SAD를 갖는 점이 대각 방향이라면 [i.e., (α-1, α), (α+1, α), (α, α-1), or (α, α+1)], (그림 7)의 (b)와 같은 탐색 패턴을 갖고 새로운 5개의 탐색 점을 추가하여 가장 작은 SAD를 갖는 점을 가운데 점으로 하여 움직임 벡터를 추정하게 된다.

단, 탐색 영역(예, ±7)을 벗어나는 모든 점들은 무시한다. 매 단계마다 가장 작은 SAD를 갖는 점은 재정의 되고, 가장 작은 SAD를 갖는 점이 이전 탐색 단계에서 가장 작은 SAD를 갖는 점과 같다면 단계 5로 이동, 그렇지 않으면 단계 3을 반복 수행한다.

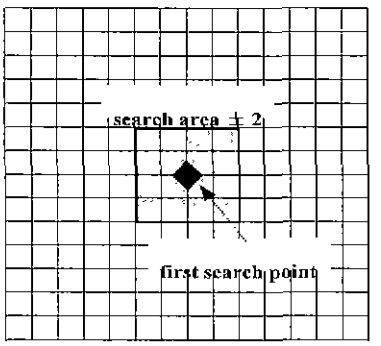
단계 4 : ±2의 탐색 영역을 갖고 전역 탐색을 수행하고 단계 5로 이동한다.

단계 5 : 탐색을 멈추고 최종적으로 원하는 움직임 벡터를 찾게된다



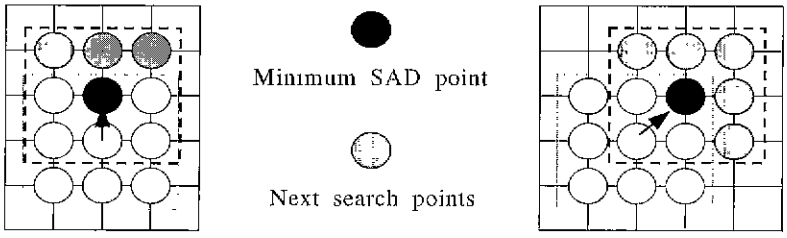
(a)

(그림 5) 블럭이 ④의 방향을 갖는 경우의 탐색 과정



(b)

(그림 6) 블럭이 ⓞ의 방향을 갖는 경우 탐색영역 ±2인 전역탐색을 수행



(a) Face 점에서 가장 작은 SAD값을 갖는 경우의 탐색 패턴
(b) Vertex 점에서 가장 작은 SAD값을 갖는 경우의 탐색 패턴

(그림 7) 제안된 PDSA 방식의 두 가지 탐색 패턴

3. 실험 결과 및 비교

본 논문에서 제안된 PDSA 알고리즘의 성능을 평가하기 위해서 <표 4>에서와 같은 실험 환경에서 실험하였다.

<표 4> 실험 환경

Development software	Visual C++ 6.0
Coding scheme	MPEG-4 Video Coding Algorithm
Frame size	CIF (352 × 288)
Block size	16 × 16 pels
Coding bit rate	384kbps
Frame rate	10 frames/sec.
Number of frames	100frames, 300frames
Search area	±7

또한 새로운 알고리즘의 성능을 정량적으로 비교하기 위해서 다음과 같은 4개의 평가기준을 사용하였다.

1) 평균 PSNR 값. 여기서, PSNR은 식 (2)와 같이 정의하였다

$$PSNR = 20 \log_{10}\left(\frac{255}{RMSE}\right) \quad dB$$
$$RMSE = \sqrt{\frac{1}{XY}\sum_{i=0}^{X-1}\sum_{i=0}^{Y-1}[f(i,j)-f'(i,j)]^2} \qquad (2)$$

여기서, f(i, j)와 f'(i, j)는 원 영상과 복호화후 재구성한 영상들을 각각 나타낸 것이다

2) 평균 MSE (Mean-Square-Error) 값

3) 각 블럭에 대한 탐색 횟수

4) 전역 탐색과 비교하여 각 블럭이 같은 움직임 백터를 갖는 개수

새로운 알고리즘을 평가하기 위해서 2가지 방법으로 실험을 하였다. 첫 번째는 총 100프레임 영상을 매 프레임 마다 부호화를 하여 나온 결과이고, 두 번째는 총 300프레임 영상을 매 3프레임 마다 1프레임을 선택하여 총 100프레임을 부호화한 결과이다.

<표 5>에서 <표 8>까지는 각각의 영상시퀀스의 처음 100프레임씩을 부호화한 결과이고, <표 9>에서 <표 12>는 총 300프레임중 100프레임을 선택하여 부호화한 결과이다

다른 고속 탐색 알고리즘과 새로 제안된 PDSA 알고리즘의 평균 PSNR 값을 비교하는 결과는 <표 5>와 <표 9>에 나타나 있다. <표 6>과 <표 10>은 각 블럭에 대한 평균 MSE값이다. 각 블럭에 대한 평균 탐색 횟수는 <표 7>과 <표 11>에 나타나 있다 그리고 전역 탐색과 비교하여 각 블럭마다 같은 움직임 벡터를 갖는 개수를 나타내는 결과는 <표 8>과 <표 12>에 나타내었다.

이러한 결과를 본다면 새로 제시한 방식은 기존의 고속 탐색 알고리즘보다 탐색 횟수에서 다이아몬드 탐색을 제외한 모든 면에서 우수함을 볼 수 있다. 특히 영상의 압축에서 중요한 요소인 PSNR값에서는 평균적으로 1.5dB 정도 높게 나타나고, 움직임이 많은 영상의 경우 최고 3.4dB 정도의 우수함을 나타내고 있다 또한 정확한 움직임 백터를 찾는 비교에 있어서도 영상의 종류에 상관이 없이 다른 고속 탐색 알고리즘보다

월등히 우수함을 볼 수 있었다.

결국 움직임이 있는 동영상의 경우 짧은 시간에 영상의 움직임이 있더라도 그 시간에는 많은 움직임이 있을 수 없음으로 인해서 움직임 벡터는 가운데 중심적인 특성을 갖게 된다. 또한 이전 영상의 블럭들의 움직임 벡터들을 조사함으로써 현재 블럭의 움직임 벡터를 예측 할 수 있는 것이다. 따라서 본 논문에서 제안한 방식은 이전 프레임 블럭의 움직임을 조사함으로써 현재 영상의 움직임이 어느 방향으로 진행할 것인가를 예측하고 그 방향으로 첫 번째 탐색 위치를 보정함으로써 움직임 추정의 성능을 높이려는 방식이다. 결국 새로 제안한 알고리즘의 경우 수치적으로 모든 면에서 우수함을 나타낼 뿐만 아니라 영상의 부호화후 복호화한 영상의 화질에 있어서도 (그림 8)에서 (그림 10)에 나타낸 것과 같이 우수한 화질을 제공한다. 부호화후 복호화한 영상에 대해서 연속적인 프레임을 본다면 알고리즘들의 비교에 대한 확실한 결과를 볼 수 있지만 지면 관계상 생략하고 각각의 영상에 대해서 확연한 차이가 나는 특정 프레임에 대해서 (그림 8)에서 (그림 10)에 나타내었고, 그림에서 보는 것과 같이 다른 고속 탐색 알고리즘들은 첫 번째 탐색이 잘못 되어 국부적인 탐색을 하게 되어서 영상의 화질이 많이 떨어짐을 볼 수 있고 본 논문에서 제안한 PDSA방식은 화질이 우수한 전역 탐색과 비슷한 결과를 나타낸다. 결론적으로 새로 제안한 알고리즘은 영상의 압축 성능에 있어서 우수한 성능을 나타내고 부가적인 영상의 성능을 평가하는 모든 면에서 다른 고속 탐색 알고리즘 보다 우수함을 볼 수 있었다.

〈표 5〉 평균 PSNR 값 (연속된 100프레임을 부호화 결과)

Image / ME Method	Sequence Image							
	Coastguard	Foreman	Mother	News	Stefan	Table	Average	Percentage
FSA (±15)	31.279	36.801	41.623	41.471	29.676	34.107	35.826	100%
TSS	29.200	32.917	41.547	41.346	25.636	30.703	33.558	93.67%
FSS	29.039	32.915	41.547	41.371	25.697	30.842	33.568	93.70%
UCBDS	28.620	32.695	41.558	41.414	25.543	30.797	33.438	93.33%
PSA	29.197	33.017	41.561	41.430	25.750	31.190	33.691	94.04%
PDSA	30.932	36.423	41.604	41.433	27.342	33.303	35.173	98.18%

〈표 6〉 영상의 평균 MSR 값 (연속된 100프레임을 부호화 결과)

Image / ME Method	Sequence Image							
	Coastguard	Foreman	Mother	News	Stefan	Table	Average	Percentage
FSA (±15)	130638	681.45	273.97	189.54	256998	95885	830.03	100%
TSS	2361.07	1291.00	339.92	262.01	4022.15	2241.41	1752.08	211.19%
FSS	2296.60	1285.09	339.76	261.76	4038.71	2136.71	1726.44	208.00%
UCBDS	2261.28	1452.47	338.93	254.00	4047.73	2212.41	1761.19	212.18%
PSA	2397.02	1321.51	340.56	266.86	4104.91	2278.41	1784.88	215.04%
PDSA	1541.66	794.29	295.40	207.22	2602.57	1331.18	1127.60	135.85%

〈표 7〉 각 블럭에 대한 평균 탐색 횟수 (연속된 100프레임을 부호화 결과)

Image / ME Method	Sequence Image							
	Coastguard	Foreman	Mother	News	Stefan	Table	Average	Percentage
FSA (±15)	961	961	961	961	961	961	961	100%
TSS	27	27	27	27	27	27	27	2.81%
FSS	27.21	27.23	27.01	27.00	27.45	27.55	27.24	2.83%
UCBDS	14.08	14.62	13.17	13.15	14.70	16.55	14.38	1.50%
PSA	27.53	27.48	27.05	27.04	27.54	27.37	27.34	2.74%
PDSA	16.71	22.35	23.59	23.79	21.11	20.90	21.43	2.23%

〈표 8〉 전역 탐색과 비교해서 같은 움직임 벡터를 갖는 블럭의 개수 (연속된 100프레임을 부호화 결과)

Image / ME Method	Sequence Image							
	Coastguard	Foreman	Mother	News	Stefan	Table	Average	Percentage
FSA (±15)	396	396	396	396	396	396	396	100%
TSS	39.15	139 17	353 52	364.63	153 35	102.13	191 99	48 48%
FSS	39 44	139.41	353 58	364 66	153.55	108 46	193 19	48 79%
UCBDS	39.54	135 84	352.33	364.28	153.37	106 48	192 01	48.49%
PSA	35.49	141.92	353 64	361.80	153.21	106 40	192 58	48 63%
PDSA	341 14	326 53	383.26	388.90	259.65	200.84	316 72	79 98%

〈표 9〉 영상의 평균 PSNR 값 (총 300 프레임 중 100 프레임을 선택하여 부호화한 결과)

Image / ME Method	Sequence Image							
	Coastguard	Foreman	Mother	News	Stefan	Table	Average	Percentage
FSA (±15)	30 251	34.859	41 608	40.449	26 943	36.488	34 933	100%
TSS	28.238	31.217	40 259	40.300	25.161	35 015	33 165	95 78%
FSS	28 495	31.826	40 413	39 809	25 263	35.913	33 620	96 24%
UCBDS	28.528	31 796	40 911	40 404	25.534	35 457	33.772	96 68%
PSA	28 360	32.094	40 869	10 380	25.627	35 820	33.858	96 92%
PDSA	28 642	33 525	41 584	40 443	25.677	36.432	34 384	98 43%

〈표 10〉 영상의 평균 MSR 값 (총 300 프레임 중 100 프레임을 선택하여 부호화한 결과)

Image / ME Method	Sequence Image							
	Coastguard	Foreman	Mother	News	Stefan	Table	Average	Percentage
FSA (±15)	1885.95	1377 28	414.56	419.82	3227.60	950 17	1379 23	100%
TSS	3560.25	3071 02	645.18	729 76	5817.80	1607 14	2571 86	186 47%
FSS	3580.86	3171 91	653.70	736 42	5848.08	1713 27	2615 7	189 65%
UCBDS	3586 26	3158.83	647 56	718 50	5897 48	1716 03	2620 78	190 02%
PSA	3807.63	3527 78	666.78	772.90	6189 36	1892.42	2809 48	203.70%
PDSA	2048.16	1966.69	435.55	453 63	4257 56	1157 73	1719 89	124 70%

〈표 11〉 각 블럭에 대한 평균 탐색 횟수 (총 300 프레임 중 100 프레임을 선택하여 부호화한 결과)

Image / ME Method	Sequence Image							
	Coastguard	Foreman	Mother	News	Stefan	Table	Average	Percentage
FSA (±15)	961	961	961	961	961	961	961	100%
TSS	27	27	27	27	27	27	27	2 81%
FSS	27.50	28 33	27 17	27.15	28 19	27 76	27 68	2.88%
UCBDS	14 85	17 72	14.02	13 73	17.41	15 79	15.59	1 62%
PSA	27.41	29 18	27.21	27 30	28 39	28 08	27.93	2 90%
PDSA	22.32	22.78	23.04	23.62	22 23	23 52	22 93	2.39%

〈표 12〉 전역 탐색과 비교하여 각 블럭이 같은 움직임 벡터를 갖는 개수
(총 300 프레임 중 100 프레임을 선택하여 부호화한 결과)

Image / ME Method	Sequence Image							
	Coastguard	Foreman	Mother	News	Stefan	Table	Average	Percentage
FSA (±15)	396	396	396	396	396	396	396	100%
TSS	28 90	49.17	278.94	326.06	67 67	247.47	166 37	42 01%
FSS	29 03	49.72	279 13	326 09	67.76	247 31	166.51	42 05%
UCBDS	29.33	50.35	277 09	326 31	67.64	246.97	166.23	42.00%
PSA	24.92	49 82	280.43	326.83	68.12	246.77	166 15	41 96%
PDSA	93.09	182 26	355.85	354 12	118.45	269 00	228.80	57 78%

(a) Full search

(b) Three-step search

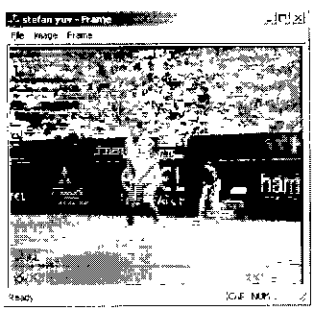
(c) Four-step search

(d) UCBDS

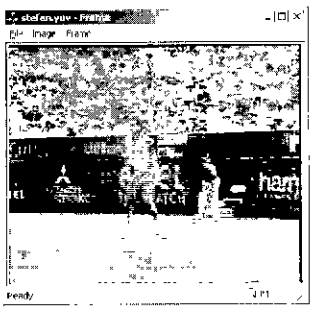
(e) PSA

(f) 제안된 PDSA

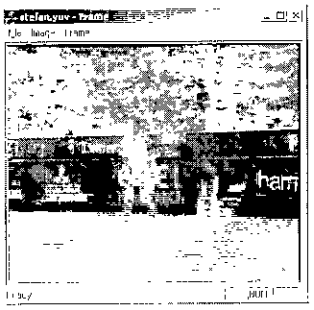
(그림 8) 재생성한 Forman 39번째 프레임 이미지



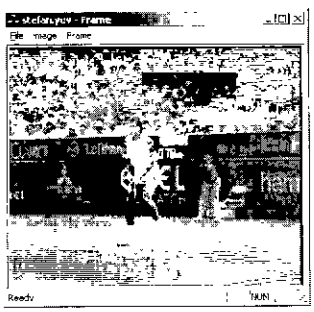
(a) Full search



(b) Three-step search

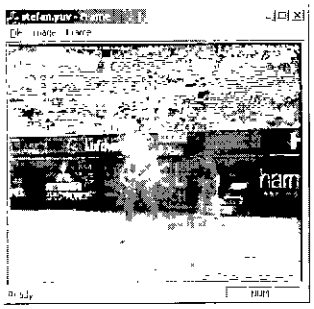


(c) Four-step search



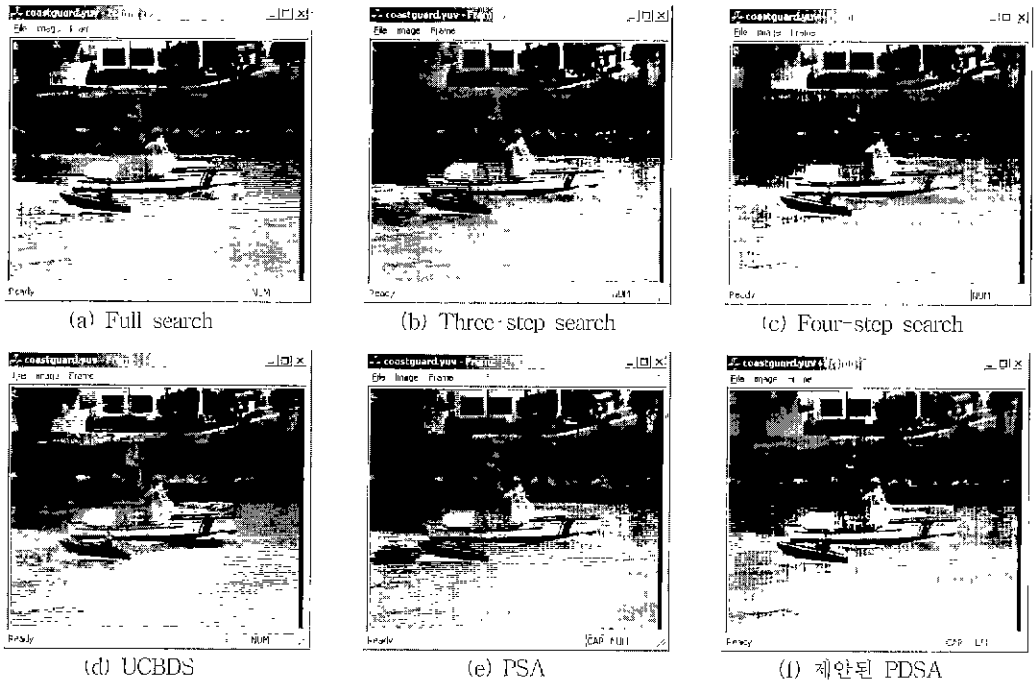
(d) UCBDS

(e) PSA



(f) 제안된 PDSA

(그림 9) 재생성한 Stefan 69번째 프레임 이미지



(a) Full search (b) Three-step search (c) Four-step search

(d) UCBDS (e) PSA (f) 제안된 PDSA

(그림 10) 재생성한 Coastugard 81번째 프레임 이미지

4. 결 론

본 논문에서 제안한 PDSA방식은 현재 프레임과 이전 프레임 사이의 상관성, 즉 연속된 영상의 움직임 벡터의 방향성은 높은 상관도를 갖는다는 특성을 이용하였다. 제안된 방식을 검증하기 위한 컴퓨터 시뮬레이션 결과를 보면 PSNR, MSE 및 FS 알고리즘과 비교하여 움직임 벡터가 같은 좌표를 갖는 블럭의 개수에서 다른 고속 탐색 알고리즘들보다 우수한 탐색 결과를 나타내고 있다. 즉, 위의 결과들에 나타난 것과 같이 PSNR값에 있어서 평균 15dB의 성능의 향상을 가지왔고, 평균 탐색 횟수에 있어서도 약 20% 정도를 줄었으며 전역 탐색과 비교하여 같은 움직임 벡터를 갖는 블럭의 개수에서는 움직임이 많은 영상의 경우 월등한 성능의 향상을 가져왔다. 이러한 결과를 종합해 본다면 본 논문에서 제안한 알고리즘은 영상의 시간적인 특성을 이용함으로써 첫 번째 탐색 위치를 보다 정확하게 예측 보정함으로써 다른 고속 탐색 알고리즘보다 압축 성능과 그 이외의 부가적인 면에서 월등히 좋은 결과를 나타냄을 볼 수 있었다. 그러나 현재 제시한 예측 방향성 탐색 알고리즘은 움직임 벡터의 방향성 정보를 이용하고는 있지만 이동 거리에 대한 정보를 이용하지 못한다. 즉, 현재의 예측 방향성 탐색 알고리즘은 첫 번째 탐색 위치를 정할 때 블럭의 방향으로 2화소씩 일정하게 이동한다. 그러나 일정하게 2화소씩 이동하는 방식보다는 이전 프레임 블럭의 움직임 벡터의 이동 거리를 조사하여 그 이동 거리를 적응적으로 이용한다면 보다 좋은 압축 성능을 얻을 수 있을 것이다. 움직임 벡터의 이동 거리를 조사하여 적응적으로 첫 번째 탐색 위치를 정하는 방식은 현재 연구 진행중에 있다.

참 고 문 헌

[1] Viet L. Do and Kenneth Y. Yun, "A Low-power VLSI architecture for full-search block-matching motion estimation," *IEEE Trans. on Communicaitons*, Vol.8, No.4, pp.393-398, Aug. 1998

[2] Chun-Hung Lin and Ja-Ling Wu, "A lightweight genetic block-matching algorithm for video coding," *IEEE Trans. on CSVT*, Vol.8, No.4, pp.386-392,

Aug. 1998

[3] Tien-ying Kno and C.-C. Kuo, "Fast overlapped block motion compensation with checkboard block partitioning," *IEEE Trans. on CSVT*, Vol.8, No.6, pp.705-712, Oct. 1998.

[4] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. on CSVT*, Vol.4, No.4, pp.438-442, Aug. 1994.

[5] Lai-Man Po and Wing-Chung Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Trans. on CSVT*, Vol.6, No.3, pp.313-317, June 1996.

[6] Jo Yew Tham, Surendra Ranganath, Maitreya Ranganath, and Ashraf Ali Kassim, "A novel unrestricted Center-biased diamond search algorithm for block motion estimation," *IEEE Trans. on CSVT*, Vol.8, No.4, pp.369-377, Aug. 1998.

[7] Lijun Luo, Carong Zou, Xiqi Gao, Member, IEEE, Zhenya He, Fellow, IEEE, "A new prediction search algorithm for block motion estimation in video coding," *IEEE Trans. on CSVT*, Vol.43, No.1, pp.56-61, Feb. 1997.

[8] Liang-Wei, Jhing-Fa Wang, Jau-Yien Lee, and Jung-Dar Shie, "Dynamic search-window adjustment and interlaced search for block-matching algorithm," *IEEE Trans. on CSVT*, Vol.3, No.1, pp.85-87, Feb. 1993.

[9] Alexis M. Tourapis, Oscar C. Au, Ming L. Liou, "Fast motion estimation using circular zonal search," *proc of SPIE VCIP*, Vol.3653, pp.1496-1504, Jan. 1999.

[10] M. Ghanbari, "The cross-search algorithm for motion estimation," *IEEE Trans. on CSVT*, Vol.38, No.7, pp.950-953, July 1990.

[11] Jer Min Jou, Pei-Yin Chen, and Jian-Ming Sun, "The gray prediction search algorithm for block motion estimation," *IEEE Trans. on CSVT*, Vol.9, No.6, pp.843-848, Sep. 1999.

서 재 수

e-mail : sjs@keobuksun.kemyung.ac.kr

1998년 계명대학교 컴퓨터·전자공학부 졸업(공학사)

2000년 계명대학교 대학원 컴퓨터공학과 졸업(공학석사)

2000년~현재 (주)웰컴정보시스템

관심분야 : 영상처리, 컴퓨터 그래픽스

남 재 열

e-mail : jynam@kmucc.kemyung.ac.kr

1983년 경북대학교 전자공학과 졸업 (공학사)

1985년 경북대학교 대학원 전자공학과 졸업 (공학석사)

1991년 University of Texas at Arlington 전기공학과 졸업 (공학박사)

1985년~1987년 한국전자통신연구소 연구원

1991년~1895년 한국전자통신연구소 선임연구원

1995년~현재 계명대학교 컴퓨터·전자공학부 조교수

관심분야 : 영상신호처리, 영상통신

곽 진 석

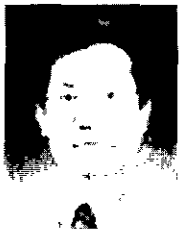
e-mail : jskwak@video.etri.re.kr

1992년 홍익대학교 공과대학 전자공학과 졸업 (공학사)

1994년 홍익대학교 대학원 전자공학과 졸업 (공학석사)

1994년 ~현재 한국전자통신연구원 선임연구원

관심분야 : 영상부호화, VLSI 알고리즘 및 아키텍처



이 명 호

e-mail : mhlee@video.etri.re.kr

1983년 숭실대학교 공과대학 전자공학과 졸업 (공학사)

1985년 숭실대학교 대학원 전자공학과 졸업 (공학석사)

1996년 일본 오사카대학 통신공학과 졸업 (공학박사)

관심분야 : 영상부호화, 멀티미디어 통신, 컴퓨터 그래픽스, 디지털 방송