영역에 따른 탐색 패턴을 이용한 움직임 벡터 예측 방법

이경희, 석진욱, 서재원
충북대학교, 한국전자통신연구원, 충북대학교

Motion Vector Estimation using Search Pattern according to Region

Kyung-Hee Lee*, Jin-Wuk Seok**, Jae-Won Suh*
*Chungbuk National University, **Electronics and Telecommunications Research Institute

Abstract - 움직임을 예측하는 방법은 동영상 압축에서 중요한 부분 중 하나이다. 동영상은 움직임이 빠르거나 혹은 느린 것으로 나누어 볼 수 있고 이에 따라 움직임을 예측하는 방법을 달리 할 수 있다. 이 논문에서는 동영상의 움직임 벡터의 분포를 예상해보고 그에 따른 영역 대하여 다른 탐색 패턴을 적용하여 움직임 벡터를 예측하는 방법을 제안하였다.

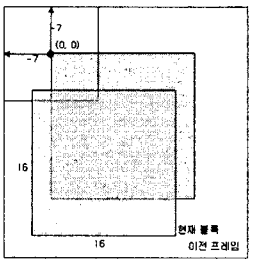
1. 서 론

이동통신에서 멀티미디어 서비스의 발달로 인하여 동영상을 빠르고 효율적으로 압축하는 기술이 중요하게 여겨지고 있다. 동영상 압축을 위해서는 동영상이 가지고 있는 중복성을 줄이는 것이 핵심이다. 동영상은 공간적, 시간적, 통계적 중복성을 가지고 있으며 DCT(Discrete Cosine Transform), ME/MC(Motion Estimation/Motion Compensation), VLC(Variable Length Coding) 등을 이용하여 각각의 중복성을 제거한다. 이중 시간적 중복성을 제거하는 것이 동영상 압축에 가장 큰 영향을 주는데 많은 수행 시간을 요구하는 단점이 있다. 시간적 중복성을 줄이는 방법은 움직임 예측(ME)을 통하여 시간적 상관관계가 있는 동영상의 프레임간의 움직임 벡터(MV)를 찾아 이전 프레임과의 차이를 구하여 그 차이 값을 압축함으로써 그 중복성을 줄인다. 움직임 예측에는 블록 정합 방법(Block Matching Algorithm - BMA)이 사용된다. BMA는 현재 프레임의 기준 블록을 이전 프레임에서 그 블록과 가장 유사한 블록을 찾아 그 변위량인 MV를 구하는 것이다. 보통 이전 프레임에서 현재 프레임과 같은 위치에 ±변위의 탐색 범위를 만들고, 그 안에서 SAD(Sum of Absolute Difference)나 MSE(Mean Squared Error) 값을 찾아 그 값이 가장 작은 곳의 MV를 찾아낸다. 이런 BMA 방법에는 전역 탐색 방법(Full Search BMA - FS)이 있다. FS는 탐색 범위 안에 있는 모든 블록과 기준 블록을 서로 조사하여 SAD 값이 가장 작은 것을 찾아내는 것으로, 가장 유사한 블록을 찾아 낼 수는 있지만 탐색 범위를 모두 조사하므로 그 시간과 복잡도가 커지게 된다. 그렇기 때문에 시간과 복잡도를 줄이기 위해 고속 블록 정합 방법(Fast BMA)이 제안되었고, 그 방법에는 3단계 탐색 방법(TSS)[1], 다이아몬드패턴 탐색 방법(Diamond Search - DS)[2], 육각패턴 탐색 방법(Hexagon-based Search - HEXBS)[3] 등이 있다. 이 방법들은 FS의 단점인 시간과 복잡도를 줄이는 것에 있으며 탐색 범위 안에서 탐색 점의 수를 줄이는 방법 혹은 탐색 패턴을 이용하여 SAD가 최소 점인 곳을 찾아내게 된다.

2. 본 론

2.1 기존의 탐색 방법

기존의 탐색 방법으로 대표적으로 FS와 고속으로 탐색하는 TSS, DS, HEXBS 등이 있다.

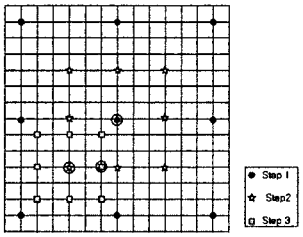


<그림 1> 전역 탐색 방법(Full Search Algorithm)

<그림 1>처럼 ±7의 탐색 범위를 갖는 FS라면 16x16의 현재 블록을 이전 프레임에서 225번 식 (1)과 같이 비교하는 계산을 해야 한다. 이에 따라 고속 탐색 방법들은 비교하는 계산량을 줄이기 위해 탐색점 혹은 탐색 패턴을 이용한다. 식 (1)에서 $I_t(k,l)$은 현재 블록의 화소값, $I_{t-1}(k+i,l+j)$는 새로운 좌표에서 이전 프레임의 블록 화소 값이다.

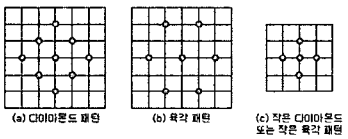
$$SAD = \sum_{k=1}^{16}\sum_{l=1}^{16} |I_t(k,l) - I_{t-1}(k+i,l+j)| \quad (1)$$

TSS는 <그림 2>와 같이 첫 단계에 탐색 변위(D)의 D/2의 거리를 갖는 8점과 중심점으로 구성된 9개의 점을 기본 탐색점으로 하여 9개의 점 중 SAD가 가장 작은 점을 다음 단계의 중심점으로 잡고, 첫 단계에서 설정한 탐색 거리의 반을 갖는 8점을 찾는다. 이렇게 3단계까지 수행 후, 3단계에서 가장 작은 SAD 값을 갖는 점을 정합 위치로 결정한다. FSS(Four Step Search)는 3단계 수행 후, 마지막 단계를 한 번 더 수행하게 한다.



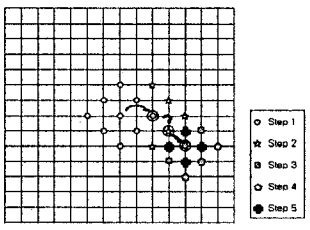
<그림 2> 3 단계 탐색 방법 (Three Step Search - TSS)

DS와 HEXBS는 각각의 패턴을 가지고 탐색을 수행하며 그 패턴은 <그림 3>에 나타내었다.



<그림 3> (a) 다이아몬드 패턴, (b) 육각 패턴, (c) 작은 다이아몬드 또는 육각 패턴

DS는 <그림 4>와 같이 <그림 3> (a)의 다이아몬드 패턴을 이용해서 중심에서의 SAD가 최소값을 가질 때까지 위치를 이동하며 탐색을 진행한다. <그림 4>처럼 3번의 중심 이동 후 중심에서 최소값을 갖는다면 <그림 3> (c)의 작은 다이아몬드 패턴을 이용하여 인접한 상하좌우 위치에서 최소 SAD 값을 찾는다. HEXBS는 DS와 동일한 탐색 방법을 사용하지만 각 스텝에 <그림 3> (b)의 육각 패턴을 이용하고 마지막 단계에서는 <그림 3> (c)의 작은 육각 패턴을 사용하여 최소 SAD 값을 찾아낸다.



<그림 4> 다이아몬드 탐색 방법 (Diamond Search - DS)

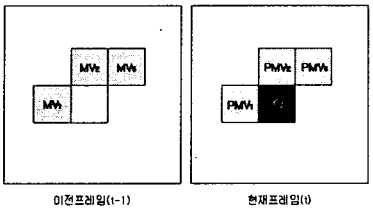
2.2 제안하는 고속 탐색 방법

DS나 HEXBS는 영상이 수직 또는 수평방향으로 원점에서 많이 움직이지 않는다는 가정에 따라 만들어진 탐색 방법이기 때문에 원점 주변의 국부적 최소값에 빠질 수 있는 단점이 있다.

2.2.1 영역에 따른 탐색 패턴을 이용한 움직임 벡터 예측

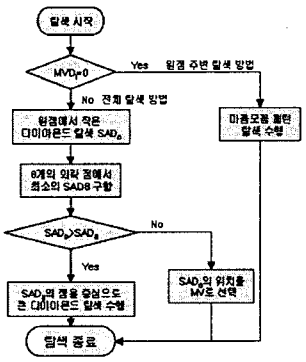
제안하는 방법은 먼저 영상간의 움직임을 예측해본다. 이 예측은 영상의 시간적 및 공간적 중복성을 이용하여 <그림 5>와 같이 현재 탐색 할 매크로블록(MB)의 MV 결정에 사용되는 $PMV_{(1\sim3)}$과 이전 프레임에서 동일한 위치의 $MV_{(1\sim3)}$사이의 값을 식 (2)를 이용하여 $MVD_{(1\sim3)}$을 구하여 그 값의 크기에 따라 현재 MB의 탐색 패턴을 결정한다.

$$MVD_i = MV_i - PMV_i \ , \ i = 1,2,3 \qquad (2)$$



<그림 5> 현재 MB의 움직임 예측 방법

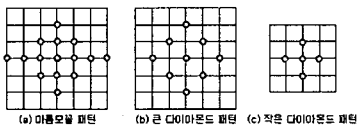
제안한 탐색 알고리즘은 <그림 6>에 흐름도로 나타내었다.



<그림 6> 제안하는 탐색 알고리즘

1) 원점 주변 탐색

$MVD_{(1\sim3)}$의 차이가 없다면 <그림 7> (a) 마름모꼴 패턴의 탐색점을 이용하여 원점 주변에서 MV를 찾는다. 제안한 마름모꼴 패턴은 큰 다이아몬드 패턴과 유사하다. 그러나 마름모꼴 패턴 탐색점을 이용한 원점 주변 탐색은 DS와 달리 원점 주변에서 단 한번만 탐색한다는 것이 큰 차이점이다.



<그림 7> 제안하는 탐색 방법에 쓰이는 여러 가지 패턴들

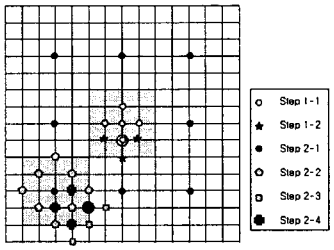
마름모꼴 패턴은 영상이 수직 방향보다 수평 방향으로의 움직임이 더 많다는 것을 기반으로 제안하였다. 이것은 동영상 샘플 11개를 이용하여 평균적인 MV의 위치를 조사하여 만들었다. <표 1>에서 보듯이 음영 처리된 마름모꼴 내에서 MV가 발생할 확률은 90%이상임을 확인할 수 있다. 제안된 마름모꼴 패턴은 15개의 탐색점을 가지며 TSS가 25개의 탐색점을 찾는 것에 비하여 10개나 단축하는 효과를 보인다. DS가 원점에서 최소점을 가지게 되는 경우와 비교할 때에도 단지 2개 증가한 것과 같다.

<표 1> 영상이 가지는 MV 분포도

	-4	-3	-2	-1	0	1	2	3	4
-4	0.04	0.01	0.01	0.01	0.03	0.01	0.01	0.00	0.00
-3	0.04	0.07	0.02	0.04	0.08	0.02	0.01	0.01	0.02
-2	0.01	0.04	0.11	0.13	0.25	0.12	0.04	0.03	0.02
-1	0.04	0.06	0.14	0.47	1.41	0.56	0.20	0.10	0.11
0	0.31	0.60	1.33	2.27	81.92	2.17	0.48	0.39	0.28
1	0.07	0.09	0.20	0.42	1.09	0.52	0.22	0.06	0.03
2	0.04	0.02	0.04	0.06	0.18	0.08	0.08	0.04	0.01
3	0.01	0.01	0.01	0.01	0.06	0.02	0.01	0.03	0.03
4	0.00	0.00	0.01	0.02	0.03	0.01	0.01	0.00	0.02

2) 전체 탐색

각각의 $MVD_{(1\sim3)}$ 값이 있다면 움직임이 존재하는 MB라고 가정하여 한다. 먼저 <그림 8>과 같이 원점을 제외하고 탐색 변위의 1/2의 위치에, 8개의 외곽 점을 정의하여 SAD를 비교하고 최소점을 찾아낸다. 원점 주변에서는 작은 다이아몬드 패턴을 가지고 탐색하여 최소점을 찾는다. 다음 단계에서 원점 주변에서 찾은 최소점과 원점 밖의 외곽 점에서 찾은 최소점을 비교하여 외곽 점의 값이 더 작다면 그 곳에서 큰 다이아몬드 패턴을 가지고 탐색을 하게 된다. 이는 움직임이 큰 영상이 가지는 특성을 고려하여 탐색을 하는 것이다. 반대의 경우에는 원점 근처 지역의 중요성을 고려하여 조밀하게 최소점을 찾아는 것이다.



<그림 8> 각각의 MV가 같지 않을 때의 탐색 방법

2.2.2 영역에 따른 탐색 패턴을 이용한 움직임 벡터 예측 결과

본 결과는 Core 2 Duo E6750, 2GB Ram, XP Pro의 환경 하에서 시뮬레이션을 통해 얻어진 것이다.

<표 2> CIF 영상 120장 비교

	foreman		highway		news		평균	
	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time
TSS	31.31	7.69	34.50	6.70	32.49	6.94	32.77	7.15
DS	31.40	8.69	34.48	7.28	32.51	6.84	32.80	7.67
HEXBS	30.96	7.14	34.25	5.98	32.46	6.62	32.56	6.67
proposed	31.53	9.59	34.59	7.93	32.53	6.94	32.88	8.05

<표 3> QCIF 영상 120장 비교

	foreman		highway		news		평균	
	PSNR	Time	PSNR	Time	PSNR	Time	PSNR	Time
TSS	28.80	2.20	32.67	1.78	30.68	1.72	31.68	1.78
DS	28.92	2.50	32.68	1.95	30.70	1.81	31.69	1.95
HEXBS	28.66	2.28	32.62	1.61	30.69	1.56	31.66	1.59
proposed	28.96	2.33	32.73	1.88	30.70	1.77	31.72	1.83

CIF와 QCIF의 영상을 120프레임 기준으로 각각 TSS, DS 그리고 HEXBS를 수행하여 제안된 방법과 비교하였다. 이전 프레임과의 MV 비교 결과 움직임이 없었다면 제안된 마름모꼴 탐색을 하여 다른 고속 방법들보다 탐색점의 수가 줄어들어 수행 시간이 빨라지고, 전체 탐색 기법의 시간적 증가를 보상해주게 되므로 고속 탐색 기법들과 비교하여 평균적 시간에서 큰 차이를 보이지 않게 되었다. 하지만 PSNR에서는 동영상 특성에 EK라 지역극소점에 빠지지 않게 되므로 제안된 방법이 QCIF에서는 최고 0.1dB, CIF에서는 최고 0.57dB가 좋아지는 것을 볼 수 있었다. 이런 현상은 빠른 영상에서 더욱 두드러진 효과가 있다.

3. 결 론

블록 정합 방법에서 가장 중요한 것은 빠르게 이전 프레임에서 현재 프레임의 MB와 가장 유사한 블록을 찾아내는 것이다. 그렇기 때문에 그 탐색점의 수가 가지는 비중이 높다. 그리고 영상의 특성에 따른 MV의 값도 변화하게 되는데 이전 프레임이 빠른 변화를 보였다면, 현재 프레임도 빠른 변화를 보이는 경향이 있다. 따라서 각 MB가 가지는 MV의 크기도 큰 값들을 가질 것이라고 예상할 수 있다. 이들의 변화를 분석하여 영역에 따라 달리한 움직임 벡터 예측 방법을 제안하게 되었고, 제안한 방법이 평균 0.47초의 시간이 느려졌지만, PSNR에서 평균 0.11dB, 최고 0.57dB의 성능 개선이 있었다.

[참 고 문 헌]

[1] Reoxiang Li, "A new three-step search algorithm for block motion es timation", Circuits and Systems for Video Technology, IEEE Transact ions on Volume 4, Issue 4, pp.438 - 442 , Aug. 1994

[2] Shan Zhu,"A new diamond search algorithm for fast block matching motion estimation", Information, Communications and Signal Processin g, 1997. ICICS., Proceedings of 1997 International Conference on Volu me 1, pp.292-296, 9-12 Sept. 1997

[3] Ce Zhu, "Hexagon-based search pattern for fast block motion estimat ion", Circuits and Systems for Video Technology, IEEE Transactions on Volume 12, Issue 5, pp.349-355, May 2002

[4] Chun-Ho Cheung, "Novel cross-diamond-hexagonal search algorithms for fast block motion estimation", Multimedia, IEEE Transactions on Volume 7, Issue 1, pp.16-22, Feb. 2005

[5] 김진욱, "적응형 임계값을 이용한 움직임 벡터 예측 방법", 전자공학회 논문지 제 43권 SP편 제 6호, pp.57-64, 2006년 11월