움직임 추정을 위한 예측 방향성 탐색 알고리즘

A Predicted Direction Search Algorithm for Block Matching Motion Estimation

서 재 수*, 남 재 열*, 곽 진 석**, 이 명 호**
*계명대학교 컴퓨터 · 전자 공학부
E-mail : jynam@kmucc.keimyung.ac.kr
**한국 전자 통신 연구원 무선 · 방송 기술 연구소
E-mail : jskwak@video.etri.re.kr

Abstract

Due to 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 image frames, 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 new predicted direction search algorithm (PDSA) for block matching motion estimation is proposed in this paper.

I 서 론

MPEG-4는 멀티미디어 정보를 사용자가 대화형으로 쉽게 접근, 편집, 처리 등을 수행할 수 있도록 다양한 기능을 제공하고 있다.

요즈음 폭발적으로 성장하고 있는 인터넷망, 이동 통신망 등에서의 다양한 멀티미디어 서비스 및 대화형 영상서비스 분야에 있어서 MPEG-4 기술은 필수적인 핵심기술로 떠오르고 있다. 특히 차세대 이동 통신 기술로 각광받고 있는 IMT-2000용 단말기에 MPEG-4 방식의 핵심기술들을 이용한다면 그 시장 잠재력과 효용성은 막대할 것이다.

그러나 MPEG-4 방식의 기술들을 IMT-2000용으로 이용하기 위해서는 비디오 압축방식을 One-Chip으로 개발하여야 한다.

특히, MPEG-4 비디오 압축에 있어서 계산량의 많은 부분을 차지하는 움직임 추정 기법을 위한 효율적이고 간단한 알고리즘이 개발되어야 한다.

일반적으로, 블록 정합 움직임 추정 기법에서 프레임은 $N \times N$ 화소의 블록으로 구성되고, 각 블록은 탐색 과정에서 탐색 영역이 최대 w 화소의 이동 거리를 갖게 되어 $2N+w$의 크기에 포함되는 모든 점들을 조사하게 된다.

대표적인 블록 정합 움직임 추정 기법으로는 전역 탐색 알고리즘 (Full search algorithm)이 있다. 전역 탐색 알고리즘은 최상의 움직임 벡터를 찾기 위해서 현재 프레임 블록과 같은 좌표를 갖는 이전 프레임 블록의 위치를 중심으로 탐색 영역에 포함되는 모든 점들을 조사하여 움직임 벡터를 찾는다. 그러나 전역 탐색 알고리즘은 움직임 추정 기법들 중에서 최적의 움직임 벡터를 찾는 장점이 있는 반면에 탐색 영역에 포함되는 모든 점들을 검색하기 때문에 [표 1]에서 보는 것과 같이 움직임 벡터를 찾기 위해서 다른 움직임 추정 기법들에 비해서 계산량이 너무 많다는 단점이 있다. 따라서, 계산량을 줄일 수 있는 3단계 탐색 알고리즘 (Three-Step Search Algorithm) [1], 4단계 탐색 알고리즘

(Four-Step Search Algorithm) [2], 가운데 중심적인 다이아몬드 탐색 알고리즘 (Unrestricted Center-Biased Diamond Search Algorithm) [3], 예측 탐색 알고리즘 (Prediction Search Algorithm) [4], 등과 같은 고속 탐색 알고리즘들이 많이 연구되었다.

[표 1] 각 블록에 대한 탐색 횟수
(탐색영역 ±15)

	움직임 추정 방식				
	FSA	TSS	FSS	UCBDS	PSA
MIN	961	27	17	13	12
MAX	961	27	36	…	…

일반적으로, 연속된 영상의 경우 공간적으로 많은 연관이 있기 때문에 이전 프레임 블록에 대한 충분하고 다양한 정보를 갖고 있다면 블록의 움직임 추정을 위해서 필요한 많은 계산량을 줄일 수 있다.

그러나 고속 탐색 알고리즘들은 특정한 몇개의 점들만 조사하여 움직임 벡터를 찾기 때문에 국부적인 탐색을 하게 된다. 즉, 현재의 탐색 단계에서 몇몇의 점들이 가장 작은 SAD 값을 갖는 점과 비슷한 SAD 값을 갖게 되면 다음 탐색 단계의 탐색 방향이 정확하지 않게 되어서 국부적인 탐색을 하게 되고[5], 결국에는 압축된 영상의 성능이 저하되는 단점이 있다. 이처럼 움직임 추정 기법에 있어서 압축 알고리즘의 계산량과 압축된 영상의 성능은 상호보완적인 관계에 있다.

또한 연속된 영상의 움직임 벡터들은 가운데를 중심으로 80%정도가 ±3픽셀 이내에 포함되고, 거의 움직임이 없는 영상의 경우 90% ~ 99%정도가 ±3픽셀 이내에 포함된다[1]. 따라서, 영상의 움직임 벡터는 가운데 중심적인 특성을 갖는다,

본 논문에서 이용하고자 하는 특성인 연속된 두 프레임에서 같은 좌표를 갖는 블록들의 움직임 벡터의 방향성, 즉 같은 좌표를 갖는 블록들의 움직임 벡터가 같은 방향으로 이동하는 상관성은 [표 2]에서 보는 것과 같이 매우 높다.

[표 2] 같은 좌표의 연속된 프레임 블록에서 움직임 벡터가 같은 방향을 갖는 블록의 개수 (총 396개)

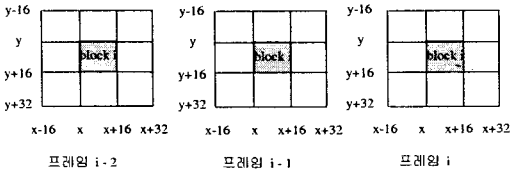
영상	같은 블록수	백분율
Coastguard	349	88.1%
Foreman	235	59.3%
Mother & Children	333	84.1%
News	368	92.9%
Stefan	308	77.8%
Table	345	87.1%
평 균	323	81.6%

본 논문에서는 알고리즘의 효율과 압축된 영상의 성능 사이의 상호 보완적인 관계, 움직임 벡터의 가운데 중심적인 특성, 연속된 프레임에서 블록에 대한 방향의 일치성 등을 이용하여 다른 고속 탐색 알고리즘 보다 성능을 향상시키면서도 부족한 정보로 인해서 야기되는 국부적인 탐색을 방지할 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, 탐색 횟수, 전역 탐색 알고리즘의 블록과 비교하여 같은 좌표의 움직임 벡터를 갖는 블록의 개수 등과 같은 기준을 통해서 다른 탐색 알고리즘과 성능을 비교 분석하였다. 제 4절에서는 결론을 내리고 현재 연구가 진행중인 움직임 추정 알고리즘을 간단히 소개한다.

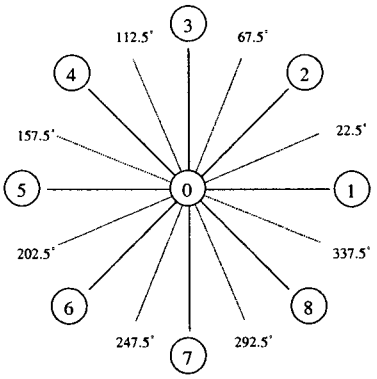
II 예측 방향성 탐색 알고리즘

최대 탐색 영역을 ±7로 하여 제안된 예측 방향성 탐색 알고리즘은 이전 연속된 프레임 블록의 움직임 추정 과정에서 얻어진 방향을 이용해서 그 블록의 방향으로 처음 탐색 위치를 2화소 이동하여 이 점을 중심으로 9개의 탐색 점들을 갖는 3×3의 탐색 영역을 시작으로 움직임 벡터를 찾게 된다. 이와 같이 현재 블록의 움직임 벡터를 찾기 위해서는 이전 연속된 2개 프레임들의 블록에 대한 방향성을 알아야 하고, 이러한 연속된 프레임에서 같은 좌표를 갖는 블록은 [그림 1]과 같다.



[그림 1] 연속된 프레임에서 같은 좌표의 블록

연속된 2개의 이전 프레임들에서 같은 좌표를 갖는 블록에 대한 방향을 알기 위해서는 [그림 2]와 같이 9개의 방향을 고려하게 된다.



[그림 2] 각도에 따른 방향성

즉, 블록의 움직임 벡터가 x, y축으로 이동한 거리를 갖고 식 (1)을 이용해서 계산되는 각도에 따라 [표 3]에서 나타나는 것과 같이 블록의 방향을 결정하게 된다.

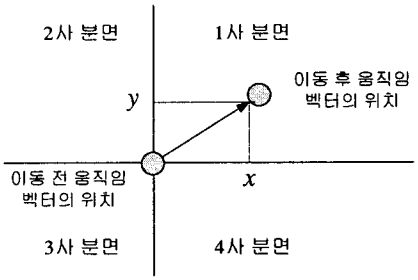
[표 3]. 각도에 따른 방향성

각 도	방 향
움직임 벡터의 이동이 없을 경우	⓪ 방향
0 °~22.5 ° & 337.5 °~360 °	① 방향
22.5 ° ~ 67.5 °	② 방향
67..5 ° ~ 112.5 °	③ 방향
112.5 ° ~ 157.5 °	④ 방향
157.5 ° ~ 202.5 °	⑤ 방향
202.5 ° ~ 247.5 °	⑥ 방향
247.5 ° ~ 292.5 °	⑦ 방향
292.5 ° ~ 337.5 °	⑧ 방향

[그림 2]와 [표 3]에서와 같이 이전 프레임에서 블록의 움직임 벡터가 이동이 없을 경우 방향은 ⓪이 되고, 움직임 벡터의 이동이 있을 경우 x, y축으로 이동한 거리에 의해서 계산되는 각도에 따라 방향을 결정하게 되는데 이러한 방향을 결정하는데 중요한 역할을 하는 각도의 계산과정은 식 (1)을 이용한다.

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

[그림 3]과 같이 이전 프레임 블록의 움직임 벡터가 움직임이 있다면 x, y축으로의 이동된 좌표를 알 수 있게 되고, 그러면 움직임 벡터가 처음 위치에서 이동한 위치까지의 상대적인 이동 거리를 알게된다.



[그림 3]. 움직임 벡터의 위치

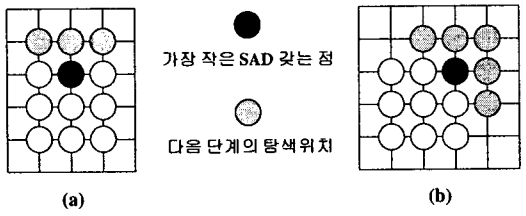
그러한 이동 거리를 가지고 식 (1)을 이용해서 움직임 벡터의 각도를 알 수 있고 이 각도를 이용해서 [표 3]과 같이 움직임 벡터의 방향을 결정할 수 있게 된다. 식 (1)에서 변수 *MVAngle* 값이 움직임 벡터의 각도가 되고 [표 4]와 같이 몇 사분면인가를 고려하여 변수 *FMVAngle* 값을 계산하게 되는데 이 *FMVAngle* 값이 그 움직임 벡터의 방향을 결정하는 최종적인 각도가 된다.

[표 4]. 움직임 벡터의 위치를 고려한 최종적인 각도 계산

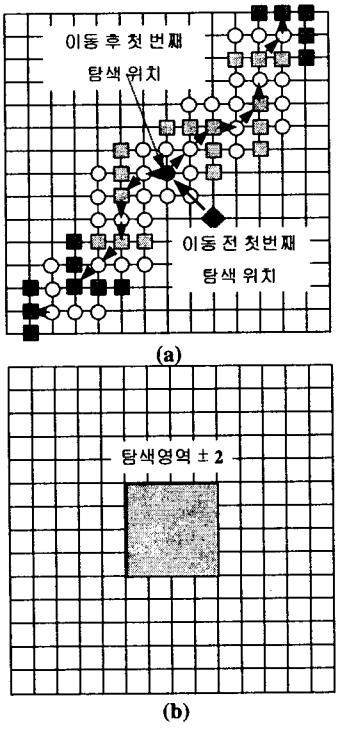
벡터의 위치	각 도
1사 분면	*FMVAngle* = *MVAngle*
2사 분면	*FMVAngle* = 180 - *MVAgle*
3사 분면	*FMVAngle* = 180 + *MVAgle*
4사 분면	*FMVAngle* = 360 - *MVAgle*

이와 같이 블록의 움직임 벡터가 움직임이 있다면 예측 방향성 탐색 알고리즘은 식 (1)과 [표 3]의 관계를 적용하여 블록의 방향을 결정하게 된다. 따라서, 그 방향으로 2화소를 이동하여 이 점을 중심으로 하는 9개의 탐색 점들을 갖는 3×3의 탐색 영역을 갖고 탐색을

시작하게 된다. 9개의 탐색 점들을 갖고 움직임 벡터를 찾는 과정에서 가장 작은 SAD값을 갖는 점이 수평·수직 방향인지 대각선 방향인지에 따라 [그림 4] (a), (b)와 같은 탐색 패턴을 갖고, [그림 5] (a), (b)와 같은 탐색 과정을 통해서 움직임 벡터를 찾게 된다.



[그림 4] 알고리즘의 두 가지 탐색 패턴. (a) 수평·수직 방향의 점이 가작 작은 SAD 값을 갖는 경우의 탐색 패턴, (b) 대각선 방향의 점이 가장 작은 SAD 값을 갖는 경우의 탐색 패턴



[그림 5] 탐색 알고리즘의 두 가지 탐색 과정. (a) 이전 프레임 블록의 움직임 벡터가 ④방향의 움직임이 있을 경우 탐색 과정, (b) 움직임 벡터의 움직임이 없을 경우의 탐색 과정

예측 방향성 탐색 알고리즘의 전체적인 탐색 과정을 살펴보면 다음과 같다.

단계 1: 이전 프레임 블록의 움직임 벡터가 이동이 있는 가를 결정. 만약 현재의 블록과 같은 좌표를 갖는 이전 프레임 블록의 움직임 벡터가 이동이 있으면, 이동 거리를 계산하고 단계 2로 이동, 이동이 없으면 블록의 방향은 ⓪이고 단계 5로 이동

단계 2: 이전 연속된 두 프레임 블록의 방향성이 다르면 블록의 방향은 ⓪이고 단계 5로 이동, 방향이 같으면 식 (1)과 [표 4]를 이용하여 움직임 벡터의 각도를 계산하고 이 각도를 이용하여 [표 3]과 같이 블록의 방향을 결정

단계 3: 단계 2에서 얻어진 방향으로 2 화소를 이동하여 그 점을 처음 3×3 탐색 영역의 가운데 점으로 정한다.

단계 4: 단계 3에서 알아낸 처음 탐색 위치를 중심으로 9개의 탐색 점들을 갖는 3×3의 탐색 영역을 갖고 탐색을 시작한다. 탐색 과정에서 이전 단계에서 가장 작은 SAD를 갖는 점과 현재 탐색 과정에서 가장 작은 SAD를 갖는 점이 같으면 탐색을 끝내는 Halfway Stop방법을 이용하고 탐색 점들 중에서 탐색 영역 ±7에 이르면 탐색을 끝내고 단계 5로 이동, 그렇지 않으면 다음 과정을 반복 수행한다.

a) 현재의 탐색 과정에서 가장 작은 SAD를 갖는 점이 수평·수직 방향이면 다음 탐색 패턴은 [그림 4] (a)와 같이 3개의 점을 추가하여 움직임 벡터를 탐색한다.

b) 현재의 탐색 과정에서 가장 작은 SAD를 갖는 점이 대각 방향이면 다음 탐색 패턴은 [그림 4] (b)와 같이 5개의 점을 추가하여 움직임 벡터를 탐색한다.

단계 5: 전체적인 탐색 과정을 끝내고 블록에 대한 움직임 벡터를 찾는다.

Ⅲ. 실험 결과

본 논문에서 제안한 예측 방향성 탐색 알고리즘의 성능을 평가하기 위해서 실험 영상은 300프레임 짜리 6개의 *CIF(352×288)* 실험 영상을 사용하여 모의 실험을 수행하였다.

MPEG-4 비디오 압축 *S/W*를 이용하여 비트율이 384kbits에 맞추어 부호화 했다. 그리고 총 300개의 프레임 중에서 3개 마다 1개씩 선택하여 100개의 프레임을 부호화 하였다. 위와 같은 실험 환경으로 여러 가지 탐색 알고리즘들의 성능을 아래 평가 기준에 따라 비교 분석하였다.

1) 각 블록에 대한 평균 PSNR
2) 각 프레임에 대한 평균 MSE
3) 각 블록에 대한 평균 탐색 횟수
4) 각 프레임에서 FS 방식과 비교해서 같은 좌표를 갖는 블록의 움직임 벡터가 이동 후 같은 좌표를 갖는 블록의 평균 개수

3프레임 마다 1프레임씩 선택하기 때문에 아래 실험결과 표들과 같이 전역 탐색을 제외한 모든 움직임 추정 기법의 전체적인 성능은 떨어진다. 그러나 전체적으로 예측 방향성 탐색 알고리즘은 다른 빠른 탐색 알고리즘에 비해서 PSNR, MSE, FS 알고리즘과 비교해서 움직임 벡터가 같은 좌표를 갖는 개수에서 좋은 성능을 나타낸다.

[표 5] 각 프레임에 대한 평균 PSNR

Image / ME Method	Sequence Image							
	Coastguard	Foreman	Mother	News	Stefan	Table	Average	percentage
FS (±15)	30.251	34.859	41.608	40.449	26.943	36.488	35.100	100%
3SS	28.238	31.217	40.259	40.300	25.161	35.615	33.465	95.34%
4SS	28.495	31.826	40.413	39.809	25.263	35.913	33.620	95.78%
UCBDS	28.528	31.796	40.911	40.404	25.534	35.457	33.780	96.24%
PSA	28.360	32.094	40.869	40.380	25.627	35.820	33.869	96.49%
PDSA	28.642	33.525	41.584	40.443	25.677	36.432	34.420	98.06%

[표 6] 각 블록에 대한 평균 MSE

Image / ME Method	Sequence Image							
	Coastguard	Foreman	Mother	News	Stefan	Table	Average	percentage
FS (±15)	1885.95	1377.28	414.56	419.82	3227.60	950.17	1379.23	100%
3SS	3560.25	3071.02	645.18	729.76	5817.80	1607.14	2571.86	186.47%
4SS	3580.86	3171.91	653.70	726.42	5848.08	1713.27	2615.71	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 7] 각 블록에 대한 평균 탐색 횟수

Image / ME Method	Sequence Image							
	Coastguard	Foreman	Mother	News	Stefan	Table	Average	percentage
FS (±15)	961	961	961	961	961	961	961	100%
3SS	27	27	27	27	27	27	27	2.81%
4SS	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 8] 각 프레임에서 FS 방식과 비교해서 움직임 벡터가 같은 좌표를 갖는 블록의 평균 개수

Image / ME Method	Sequence Image							
	Coastguard	Foreman	Mother	News	Stefan	Table	Average	percentage
FS (±15)	396	396	396	396	396	396	396	100%
3SS	28.90	49.17	278.94	326.06	67.67	247.47	166.37	42.01%
4SS	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%

Ⅳ 결론

본 논문에서 제안한 PDSA방식은 현재 프레임과 이전 프레임 사이의 상관성, 즉 연속된 영상의 움직임 벡터의 방향성은 일관성을 가진다는 특성을 이용함으로써 실험 평가 기준들 중 PSNR, MSE, FS 알고리즘과 비교하여 움직임 벡터가 같은 좌표를 갖는 블록의 개수에서 다른 고속 탐색 알고리즘들보다 우수한 탐색 결과를 나타내고 있다. 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 블록의 방향으로 2화소씩 일정하게 이동한다. 그러나 일정하게 2화소씩 이동하는 방식보다는 이전 프레임 블록의 움직임 벡터의 이동 거리를 조사하여 그 이동 거리를 이용하면 보다 좋은 압축 성능을 얻을 수 있을 것이다. 그래서 움직임 벡터의 이동 거리를 조사하여 적응적으로 첫 번째 탐색 위치를 정하는 방식을 연구하고 있다.

Ⅴ. 참 고 문 헌

[1] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. Circuit Syst. Video Technol.*, Vol. 4, pp. 438-442, Aug. 1994.

[2] Lai-Man Po and Wing-Chung Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 6, pp. 313-317, June. 1996.

[3] 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. Circuits Syst. Video Technol*, Vol 8, pp. 369-377, Aug. 1998.

[4] Lijun Luo, Cairong Zou, Xiqi Gao, Member, IEEE, Zhenya He, Fellow, IEEE, "A new prediction search algorithm for block motion estimation in video coding," *IEEE Trans, Circuit Syst, Video Technol*, Vol 43, No 1, pp. 56-61, Feb. 1997.

[5] 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, Circuit Syst, Video Technol*, Vol 3, No 1, pp. 85-87, Feb. 1993.

[6] Alexis M. Tourapis, Oscar C. Au, Ming L. Liou, "Fast motion estimation using circular zonal search," *Visual Communications and Image Processing*, Vol 3653, pp. 1496-1504, Jan. 1999.

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

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

[9] Chun-Hung Lin and Ja-Ling Wu, "A lightweight genetic block-matching algorithm for video coding," *IEEE Trans. Circuits Syst. Video Technol.*, Vol 8, No 4, pp. 386-392, Aug. 1998.

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

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