공간 상관성을 이용한 적응적 움직임 추정 알고리즘

(An Adaptive Motion Estimation Algorithm Using Spatial Correlation)

박 상 곤*, 정 동 석
인하대학교 전자공학과 멀티미디어 연구실
Sang-Gon Park, Dong-Seok Jeong
Multimedia Lab., Dept. of Electronics, Inha Univ.
g1991203@inhavision.inha.ac.kr*, dsjeong@dragon.inha.ac.kr

Abstract

In this paper, we propose a fast adaptive diamond search algorithm(FADS) for block matching motion estimation. Fast motion estimation algorithms reduce the computational complexity by using the UESA (Unimodal Error Search Assumption) that the matching error monotonically increases as the search moves away from the global minimum error. Recently many fast BMAs(Block Matching Algorithms) make use of the fact that the global minimum points in real world video sequences are centered at the position of zero motion. But these BMAs, especially in large motion, are easily trapped into the local minima and result in poor matching accuracy. So, we propose a new motion estimation algorithm using the spatial correlation among the adjacent blocks. We change the origin of search window according to the spatially adjacent motion vectors and their MAE(Mean Absolute Error). The computer simulation shows that the proposed algorithm has almost the same computational complexity with UCBDS(Unrestricted Center-Biased Diamond Search)[1], but enhance PSNR. Moreover, the proposed algorithm gives almost the same PSNR as that of FS(Full Search), even for the large motion case, with half the computational load.

Ⅰ. 서론

음성이나 동영상과 같은 멀티미디어 데이터의 경우 엄청난 데이터 양 때문에 전송이나 저장 시 높은 압축율이 필요하다. 동영상의 경우 프레임 간 상관성이 높다는 성질을 이용하면 데이터량을 현격히 줄일 수 있다. 프레임 간 상관성을 이용한 압축 방법 중 전역 탐색(Full Search, FS)의 경우 제한된 탐색 영역 내에서 정합 에러가 가장 적은 움직임 벡터를 찾을 수 있으나 많은 계산량이 필요한 단점이 있다. 이에 계산량을 줄이면서 전역 탐색의 정합에러에 근사한 값을 유지하는 고속 움직임 추정 알고리즘들이 제안되고 있다. 그 중 UESA를 가정하는 알고리즘의 경우 효과적으로 많은 계산량을 줄일 수는 있으나 지역 극소점에 빠져 많은 정합에러를 발생시키는 단점도 갖고 있다. 또한 4단계 탐색(Four Step Search)[2]이나 DS(Diamond Search)[3], UCBDS의 경우 대부분의 움직임 벡터들의 크기가 크지 않다는 점을 이용하여 더 적은 연산으로 PSNR을 향상시킬 수 있으나, 움직임이 빠른(즉 움직임 벡터가 큰)영상에 적용할 경우 정합에러가 커지는 문제점이 있다. 이에 본 논문에서는 이러한 문제점을 개선하기 위해 공간적으로 인접한 블록들의 움직임 벡터를 적응적으로 이용하는 새로운 방법을 제안한다.

Ⅱ. 제안하는 알고리즘

그림 1은 전역 탐색과 UCBDS의 PSNR을 비교한 것이다. 프레임 번호 앞부분과 뒷부분을 보면, FS의 PS-

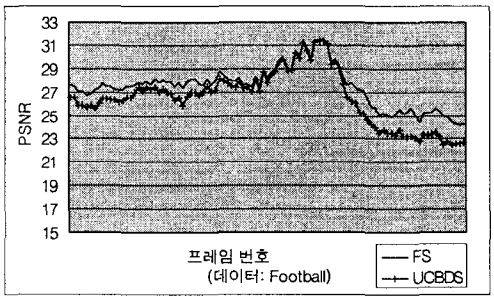

그림 1. 전역 탐색과 UCBDS의 PSNR 비교
Fig. 1. Comparison of PSNR between FS and UCBDS

NR에 비해 UCBDS의 PSNR이 많이 저하되는 것을 볼 수 있다. 이 부분의 실제 영상을 보면 다른 영상들에 비해 상대적으로 움직임이 빠른 부분이다. 즉 움직임이 빠른 영상에 대해 UCBDS는 지역 극소점에 쉽게 빠지고 그 결과 PSNR이 나빠진다.

이에 본 논문에서는 움직임 벡터들의 공간적 상관성을 이용하여 탐색 시작점을 변화시키는 방법을 제안한다. 주위 블록들의 공간적 상관성을 이용할 경우, 주위 블록들의 움직임과 현재 블록의 움직임이 상관성이 있는지 고려해 주어야 한다[4]. 이를 위하여 주위 블록을 이용한 움직임 벡터의 MAE를 구하고 이 크기에 따라 적응적으로 판단하여 주위 블록의 움직임 벡터를 이용할 지 여부를 결정한다. 공간적으로 참조할 블록 선정에 있어 공간적 상관성을 충분히 이용할 수 있도록 J-unavit[5]의 방법을 참고하였다. 하지만 본 논문에서는 시간적 상관성은 이용하지 않았다. 왜냐하면, 실제 동영상 압축의 경우 참조하는 프레임 간격이 반드시 1이 아닌데, 본 논문에서 사용하는 UCBDS는 프레임 간격이 1 이상인 경우에 계산량과 정합 에러가 커지기 때문이다[1]. 본 논문에서는 그림 2처럼 영상을 유형 1, 2, 3의 8×8 블록으로 나누고 각 유형에 따라 네 개의 주변 블록을 이용하여[5] 각각의 MAE를 구하고 이 값을 가중치로 하여 새로운 움직임 벡터를 구한다. 예를 들어 유형 2일 경우 그림 3 (a)처럼 대각선 방향의 네 개의 블록을 참조한다. 네 개의 블록 $B_i(i=1,2,3,4)$ 각각의 움직임 벡터 $V_i(i=1,2,3,4)$에 대해 $MAE_i(i=1,2,3,4)$를 구하고 이 값들을 가중치로 하여 새로운 움직임 벡터 V_v를 예측한다. 이 움직임 벡터 V_v에 대해 MAE_v를 구한다. MAE_v와 MAE_0을 비교하여 MAE_v가 작을 경우에만 공간적 상관성이 있다고 판단한다. 이 방법으로 주위 블록들의 움직임과 현재 블록의 움직임이 서로 상관성이 있는지 판단한다.

제안하는 알고리즘을 각 단계별로 설명하면 다음과

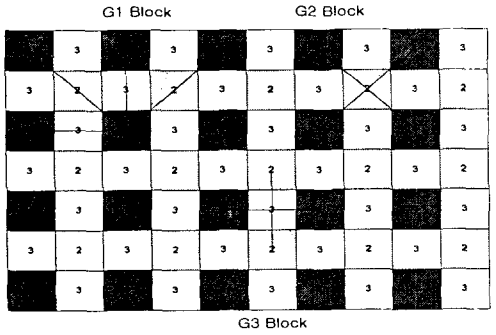

그림 2. 각 블록 유형과 참조 블록
Fig. 2. Block pattern and Reference Block

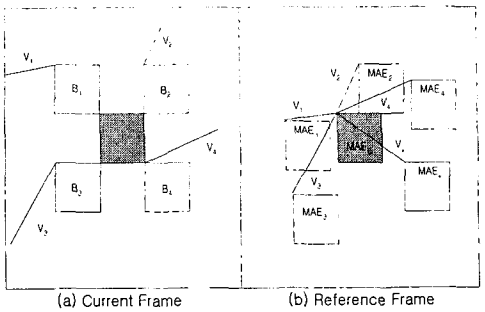

그림 3. 참조 블록의 움직임 벡터와 MAE 관계
Fig. 3. Relation between motion vector of reference block and MAE

같다.

단계 1) 현재 블록을 기준으로 이미 구한 주위 블록들의 움직임 벡터 $V_i(i=1,2,3,4)$만큼 떨어진 블록과 현재 블록 사이의 차이 값 $MAE_i(i=1,2,3,4)$를 구한다.

단계 2) 위에서 구한 MAE_i값들과 주위 블록들의 움직임 벡터들을 이용해서 새로운 움직임 벡터 V_v를 계산한다.

$$V_v = \frac{\sum_{i=1}^{4}(1-\frac{MAE_i}{MAE_{sum}})\,V_i}{4} \qquad 1-(1)$$

$$MAE_{sum} = \sum_{i=1}^{4} MAE_i \qquad 1-(2)$$

단계 3) V_v에 대한 MAE_v를 계산한다.

단계 4) MAE_v와 MAE_0을 비교하여 MAE_v가 작을 경우 탐색 시작점으로 움직임 벡터 V_v를 이용하고, MAE_0이 작을 경우 탐색 시작점을 이동하지 않는다.

단계 5) 위에서 결정한 움직임 벡터를 탐색 시작점으로 UCBDS를 적용하여 최종 움직임 벡터를 결정한다.

단계 1~3에서 MAE를 계산할 때는 계산량을 줄이기 위해 subsampling을 사용하였다. 단계 1~4까지만 적용한 예측 방법의 효과를 검사하기 위해, 예측된 탐색 시작점과 전역 탐색을 이용하여 구한 움직임 벡터의 차를 비교해 보았다. 그림 4는 football의 100개 프레임에 대해 전역 탐색을 적용하여 찾은 움직임 벡터의 분포이다. 그림 5는 전역 탐색을 이용하여 구한 움직임 벡터와 예측한 탐색 시작점과 차를 나타낸 것이다. 그림 4에 비해 움직임 벡터들이 가운데로 더 모여 있다. UCBDS와 같이 탐색 영역의 가운데서부터 탐색을 시작하는 알고리즘의 경우 탐색 시작점과 최종적으로 찾아야 하는 움직임 벡터의 거리를 줄임으로서 지역 극

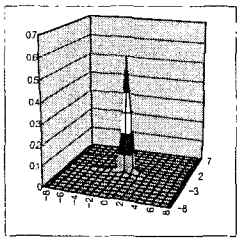

그림 4. 움직임 벡터 분포
Fig. 4. Distribution of MV

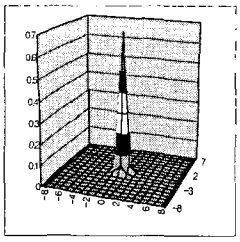

그림 5. 움직임 벡터와 예측 결과의 차
Fig. 5. Diff. betw. MV and predicted result

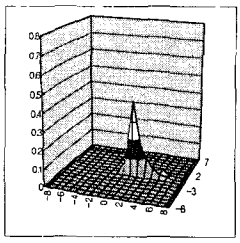

그림 6. 움직임 벡터 분포
Fig. 6. Distribution of MV

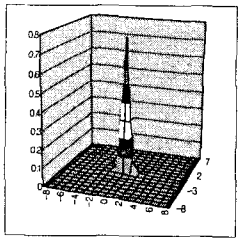

그림 7. 움직임 벡터와 예측 결과의 차
Fig. 7. Diff. betw. MV and predicted result

소점에 빠질 수 있는 경우를 줄일 수 있을 뿐 아니라 계산량을 줄이는 효과를 얻을 수 있다. 그림 5의 경우 움직임 벡터와 예측 결과의 차를 나타낸 것이므로 실제 범위는 [-16, 16]이지만 그림 4와 비교해 나타내기 위해 [-8, 8]까지만 나타냈다. 표 1, 2는 그림 4, 5에서 탐색 영역 [-8, 8]중 움직임 벡터들이 가장 많이 모인 [-3, 3]영역을 수치로 나타낸 것이다. 음영으로 표시된 다섯 점은 각각 (0,-1), (-1,0), (0,0), (1,0), (0,1)을 나타낸다. 이 다섯 점은 FADS의 단계 5에서 사용하는 U-CBDS로 최종 움직임 벡터를 찾는 과정에서, 가장 적은 수의 연산으로 움직임 벡터를 찾을 수 있는 점들이다. 즉 이 점들에 많은 움직임 벡터들이 모여 있을수록 계산량이 줄어드는 효과를 얻을 수 있다. 또한 U-ESA를 가정하는 알고리즘들의 경우 탐색 영역의 중심에서 먼 움직임 벡터를 찾을 경우 중간에서 지역 극소점에 빠질 확률이 높아진다. 음영으로 표시된 다섯 점의 경우 탐색점의 중심을 움직이지 않고 첫 단계에서 바로 마지막 단계로 넘어가므로 지역 극소점에 빠지지 않는다. Football에 전역 탐색을 적용한 경우, 음영 표시한 점들에 64.16%의 움직임 벡터가 모여 있는 반면 FADS의 탐색 시작점 예측의 결과 83.23%가 모여 있다. 그림 6, 7은 같은 실험을 garden에 대해 적용한 결

표 1. 움직임 벡터 분포

0.0005	0.0010	0.0016	0.0026	0.0021	0.0016	0.0011
0.0011	0.0015	0.0029	0.0047	0.0040	0.0033	0.0019
0.0018	0.0032	0.0064	0.0165	0.0094	0.0077	0.0033
0.0062	0.0140	0.0020	0.5689	0.0352	0.0105	0.0046
0.0020	0.0036	0.0078	0.0190	0.0107	0.0065	0.0040
0.0014	0.0025	0.0054	0.0068	0.0046	0.0025	0.0019
0.0007	0.0016	0.0020	0.0041	0.0027	0.0017	0.0009

표 2. 움직임 벡터와 예측 결과의 차

0.0004	0.0006	0.0009	0.0014	0.0012	0.0007	0.0004
0.0009	0.0012	0.0024	0.0032	0.0027	0.0010	0.0007
0.0009	0.0026	0.0095	0.0453	0.0103	0.0028	0.0012
0.0014	0.0029	0.0308	0.6868	0.0461	0.0034	0.0014
0.0009	0.0021	0.0085	0.0233	0.0126	0.0031	0.0011
0.0007	0.0009	0.0023	0.0034	0.0028	0.0014	0.0010
0.0003	0.0005	0.0011	0.0014	0.0012	0.0007	0.0006

표 3. 움직임 벡터 분포

0.0000	0.0000	0.0002	0.0004	0.0004	0.0003	0.0002
0.0002	0.0001	0.0003	0.0005	0.0006	0.0006	0.0004
0.0004	0.0006	0.0010	0.0020	0.0025	0.0026	0.0013
0.0015	0.0030	0.0076	0.0707	0.4153	0.1759	0.1008
0.0006	0.0009	0.0024	0.0034	0.0071	0.0068	0.0143
0.0004	0.0004	0.0005	0.0013	0.0013	0.0009	0.0009
0.0001	0.0002	0.0003	0.0007	0.0004	0.0004	0.0004

표 4. 움직임 벡터와 예측 결과의 차

0.0002	0.0001	0.0005	0.0007	0.0003	0.0002	0.0002
0.0005	0.0005	0.0007	0.0009	0.0007	0.0005	0.0003
0.0060	0.0065	0.0058	0.0074	0.0038	0.0016	0.0009
0.0022	0.0047	0.0313	0.7568	0.0708	0.0084	0.0043
0.0013	0.0030	0.0036	0.0107	0.0067	0.0029	0.0016
0.0004	0.0005	0.0012	0.0014	0.0011	0.0007	0.0007
0.0001	0.0002	0.0004	0.0008	0.0006	0.0003	0.0004

과이다. 표 3, 4는 그림 6, 7의 탐색 영역 중 [-3, 3]영역을 수치로 표현한 것이다. 음영으로 표시한 점에, 전역 탐색의 결과 49.90%가 모여 있는 반면, FADS의 예측 결과 87.70%가 모여 있어 PSNR의 향상과 더불어 계산량 감소의 효과를 거둘 수 있다.

Ⅲ. 실험 결과

제안한 알고리즘의 성능을 평가하기 위해 352×240의 크기를 가지는 claire, foot ball, garden 영상을 사용하여 실험을 하였다. 사용한 블록의 크기는 8×8, 탐색 영역은 17×17로 하였고 그 결과는 표 5, 6과 같다. UCBDS에 비해, 움직임이 거의 없는 claire의 경우 약 0.01dB, 움직임의 방향이 일정한 garden의 경우 0.07 dB의 PSNR 향상이 있었다. 그리고 움직임이 불규칙하고 큰 football의 경우 0.63dB의 향상이 있었다. 그리고, 탐색 시작점을 예측하는 과정에서 필요한 MAE 계산으로 인해 계산량이 늘어났으나, 전체 계산량에서는 claire의 경우에 약간의 증가가 있었을 뿐 football과 garden의 경우는 오히려 감소하였다. 이는 탐색 시작점이 전체 극소점과 가까워짐으로 해서 계산량이 줄어든 결과이다.

표 5. PSNR 비교 (단위 : dB)

	Claire	Foot Ball	Garden
FS	43.192	27.470	27.877
TSS	42.731	26.052	25.647
4SS	41.992	25.791	24.866
UCBDS	43.093	26.492	26.843
FADS	43.103	27.117	26.916

표 6. 계산량 비교

	Claire	Foot Ball	Garden
FS	1	1	1
TSS	0.087	0.087	0.087
4SS	0.038	0.058	0.048
UCBDS	0.048	0.062	0.057
FADS	0.049	0.053	0.050

TSS : Three Step Search[6]

4SS : Four Step Search

Ⅳ. 결론

본 논문에서는 고속 움직임 추정을 위해 공간 상관성을 이용하여 탐색 시작점을 적응적으로 변화시키는 방법을 제안하였다. 탐색 시작점과 최적 움직임 벡터 사이의 거리를 줄임으로서 지역 극소점에 빠질 수 있는 확률을 낮추어 PSNR을 향상시켰다. 특히 대부분의 움직임 벡터가 크지 않다는 가정을 이용하는 알고리즘들이, 움직임이 빠른 영상에서는 전역 탐색에 비해 PSNR이 많이 저하되는데 이러한 단점을 개선할 수 있었다.

참고 문헌

[1] Jo Yew Tham, Surendra Ranganath, Maitreya Ranganath, Ashraf Ali Kassim, "A novel unrestricted center biased diamond search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, No. 4, pp. 369-377, Jun. 1998.

[2] Lai-Man Po, Wing-Chung Ma, "A novel four step search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, No. 3, pp. 313-317, Jun. 1996.

[3] Shan Zhu, Kai-Kuang Ma, "A new diamond search algorithm for fast block matching motion estimation," *IEEE Trans. Image Processing*, vol.9, No.2, pp. 287-290, Feb. 2000.

[4] Jong-Nam Kim and Tae-Sun Choi, "A fast motion estimation for software based real time video coding," *IEEE Transactions on Consumer Electronics*, vol. 45, No. 2, pp. 417-426, May. 1999.

[5] Junavit Chalidabhongse, C.C Jay Kuo, "Fast motion vector estimation using multiresolution spatio-temporal correlations," *IEEE Trans. Circuits Syst. Video Technol.* vol. 7, No. 3, pp. 477-488, Jun. 1997.

[6] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," in *Proc. Nat. Telecommun. conf.*, New Orleans, LA, pp. G5.3.1-5.3.5., Nov. 29-Dec. 3 1981.

[7] M.Rehan, P.Agathoklis, A.Antoniou, "A new motion estimation technique for efficient video compression," in *Proc. of the 1997 IEEE Pacific Rim Conf. on Communications, Computers and Signal Processing*, vol. 1, pp. 326-329, Aug. 1997.