물체 움직임의 통계적 특성에 기반한 움직임 추정 방법

임동근, 호요성

광주과학기술원 정보통신공학과
광주광역시 북구 오룡동 1 번지

A Fast Block Matching Motion Estimation Algorithm based on Statistical Properties of Object Displacement

Dong-Keun Lim and Yo-Sung Ho

Kwangju Institute of Science and Technology (K-JIST)
1 Oryong-Dong Puk-Gu, Kwangju, 500-712, Korea

요약

영상압축 분야에서 움직임 추정은 영상 시퀀스에 존재하는 시간적인 상관성을 이용하기 위해 많이 사용된다. 움직임벡터를 추정하기 위해 전역탐색 블록정합 알고리즘은 많은 계산량을 요구한다. 이 문제를 해결하기 위해 여러 가지 고속 블록정합 알고리즘이 제안되었지만, 그러한 방법들은 복원영상의 화질을 다소 감소시킨다. 본 논문에서는 시간적으로 인접한 두 화면 사이에서 물체 움직임의 통계적인 특성과 정합기준을 고려하여 새로운 블록정합 움직임 추정 방법을 제안하였다. 각 블록의 움직임 정도와 예측 오류 사이의 관계를 이용하여 해당 블록에 적절한 탐색패턴을 가변적으로 설정하였다. 탐색패턴을 적응적으로 변화시켜 다른 고속 블록정합 알고리즘들보다 계산량은 감소하였고 움직임 예측 성능은 향상되었다.

1. 서론

최근 무선 혹은 인터넷을 이용한 실시간 영상 통신에 대한 관심이 증가하고 있다. 이와 같이 전송 대역폭이 제한된 채널에 멀티미디어 데이터를 효과적으로 전송하기 위해 여러 가지 데이터 압축 알고리즘이 사용되고 있다. 특히, 영상 신호에는 시간적으로 연속하는 화면 사이에는 중복되는 정보가 많으므로, 이러한 중복 정보를 움직임 추정 기법을 사용하여 효과적으로 제거할 수 있다.

영상 신호에서 물체의 움직임을 추정하기 위해 블록정합 알고리즘이 가장 많이 사용되고 있다. 블록정합 알고리즘에서는 각 화면을 사각형 블록으로 나누고, 현재 화면의 각 블록과 가장 가까운 블록을 정합기준(criteria)에 따라 이전 화면의 탐색영역에서 찾아낸다. 선택된 블록은 현재 블록을 예측하는데 사용하며, 두 블록 사이의 상대적인 위치 차이는 현재 블록의 움직임벡터로 정의된다.

전역탐색 방법은 영상 부호화를 위한 블록 단위의 움직임 추정에 널리 사용되어 왔다. 전역탐색 방법은 오차함수값을 최소화하는 움직임벡터를 구하기 위해 가능한 모든 경우를 탐색하기 때문에 일반적으로 좋은 성능을 나타낸다. 그러나, 탐색영역 전체에서 적합한 움직임벡터를 찾기 때문에 많은 계산이 필요하다. 이러한 계산량을 줄이기 위해 여러 종류의 고속탐색 방법이 제안되었다[1-6]. 그러나, 대부분의 고속탐색 방법들은 경험예측적인(heuristic) 접근방법을 사용하여 계산량을 줄이기 때문에 복원된 영상의 화질을 다소 저하시킨다.

본 논문에서는 움직임벡터를 찾는데 요구되는 계산량을 줄이도록 블록정합 알고리즘을 위한 정합기준을 개선하였고, 물체의 움직임과 화면간 차이의 통계적인 특성을 이용하여 큰 움직임과 작은 움직임에 따라 탐색패턴을 적응적으로 변경하였다. 제안한 방법은 전역탐색 방법과 유사한 움직임 예측성능을 가지며, 요구되는 계산량은 다른 고속탐색 방법과 비교하여 크게 줄었다.

2. 블록정합 방법의 정합기준

영상블록의 정합은 평균절대값차이(MAD), 평균제곱오류(MSE), 그리고 정합화소수(MPC)등의 다양한 정합기준을 사용하여 이루어질 수 있다. $I_t(k,l)$을 현재 화면의 휘도성분, $I_{t-1}(k,l)$를 이전 화면의 휘도성분이라 하고, (MV_x, MV_y)가 움직임벡터라고 하자. N×N 크기의 사각형 블록이 움직임 예측에 사용된다면, 이러한 정합기준은 다음과 같이 정의된다.

(a) 상관함수(Cross-Correlation Function , CCF)

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

(b) 평균제곱오류(Mean Squared Error, MSE)

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

(c) 평균제곱차이(Mean Absolute Difference, MAD)

$$MAD(i,j)=\frac{1}{N^2}\sum_{k=1}^{N}\sum_{l=1}^{N}|I_t(k,l)-I_{t-1}(k+i,l+j)| \quad (3)$$

(d) 정합화소수(Matching Pel Count, MPC)

$$T(k,l,i,j)=\left\{\begin{array}{ll}1, & if\ |I_t(k,l)-I_{t-1}(k+i,l+j)|\le THS\\ 0, & otherwise\end{array}\right\} \quad (4)$$

$$MPC(i,j)=\sum_{k=1}^{N}\sum_{l=1}^{N}T(k,l,i,j)$$

CCF 와 MPC 는 최대값을 가지는 위치를 움직임벡터로 잡으며, 다른 방법에서는 최소값인 경우를 움직임벡터로 잡는다. 이 가운데 CCF 와 MSE 는 곱셈과 덧셈을 필요로 하는 반면에, 다른 것들은 비교와 덧셈을 필요로 한다. 곱셈은 일반적으로 비교 동작에 비하여 더 복잡한 하드웨어가 필요하므로, 곱셈을 구현하는 데 비용이 더 든다. MAD 는 계산이 단순하기 때문에 많이 사용되며, MSE 를 사용할 때와 거의 비슷한 성능을 보인다. 비록 MPC 는 MAD 보다도 하드웨어가 간단하지만, MPC 의 성능은 선택된 임계값(THS)에 크게 영향을 받는다.

본 논문에서는 블록정합 동작의 하드웨어 복잡도를 줄이기 위하여 MAD 와 MPC 를 결합한 또 다른 정합기준을 다음과 같이 정의하였다.

$$T(k,l,i,j)=\left\{\begin{array}{ll}1, & if\ |I_t(k,l)-I_{t-1}(k+i,l+j)|\ge THS\\ 0, & otherwise\end{array}\right\} \quad (5)$$

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

새로운 정합기준은 휘도값의 차이가 어느 정도 큰 화소들의 차이만을 고려한다. 휘도값의 차이가 작은 화소들은 고려하지 않기 때문에 계산량을 줄일 수 있다. 새로운 정합기준을 적용할 때 어려운 점은 적절한 임계값 THS 선택이다. 인간시각 시스템(Human Visual System, HVS)은 휘도의 큰 차이에 대해서는 민감한 반면에, 작은 차이에는 상대적으로 둔감하다. 따라서 SAD 를 계산할 때, 화소 위치에서 화소 휘도간의 절대값 차이가 인접 블록 사이에서 인간시각이 휘도 차이를 인식할 수 있는 최소 경계값인 JND (Just Noticeable Difference) 이상의 값을 가지면 움직임 보상 후에도 그 영향이 나타나므로 JND 를 임계값 THS 로 사용하였다. 작은 움직임을 가지는 블록에 대해서는 THS 를 0 으로 설정하며, 이때 식(5)는 식(3)과 똑같아진다.

3. 새로운 고속 블록정합 방법

고속 움직임 추정을 위해 여러 가지 블록정합 방법들이 제안되었지만, 그것들은 계산량을 줄이기 위해 예측 정밀도를 약간 희생시켰다.

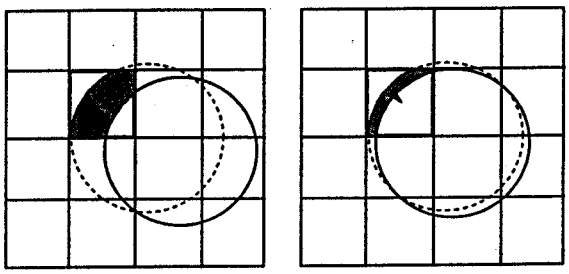
본 논문에서 제안한 방법에서는 초기 탐색패턴을 블록의 움직임 예상도에 따라 계산 복잡도의 관점에서 적절히 변화시켰다. 탐색점들의 분포는 더욱 조밀해졌지만, 탐색의 다음 단계에서 추가되는 탐색점의 갯수를 다른 방법들보다 감소시켰다. 즉, 탐색의 다음 단계에서 추가되는 탐색점의 갯수는 이전 단계의 최소점의 위치에 따라 결정된다.

영상회의 시퀀스에서 각 화면에 있는 블록들의 대부분은 배경과 같은 정적(stationary)이거나 거의 정적(quasi-stationary)인 블록이다. 이러한 블록들의 움직임벡터들은 탐색중심 영역에 밀집되어 있다. 그렇지만 어느 블록이 정적 블록인지 아닌지를 결정하는 것은 그렇게 간단하지 않다.

그림 1 과 같은 경우를 고려해 볼 때, 많은 움직임은 탐색영역 안에서 일반적으로 큰 블록차이(Block Frame Difference, BFD)를 가져온다. 그렇지만 여기에는 다음과 같은 몇 가지 예외 상황이 있다.

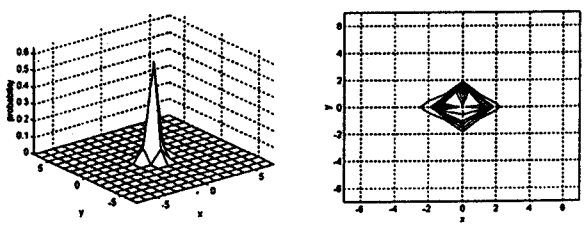
경우 1: BFD 는 임계값보다 작지만, 움직임은 큰 경우로 판정될 수 있다. 이 경우는 정적 영역에 잡음과 같은 것이 있는 때이므로, BFD 를 관측하여 이 블록들은 작은 움직임을 가진다고 판단한다.

경우 2: BFD 는 임계값보다 크지만, 움직임은 작은 경우로 판정될 수 있다. 이런 경우는 배경과 물체 사이의 화소값의 차이가 큰 경우에 발생할 수 있다. 만약 블록을 특징짓는 화소의 갯수가 적으면 이 경우가 발생했다고 예상할 수 있고, 해당 블록은 작은 움직임을 가진다고 판단한다.

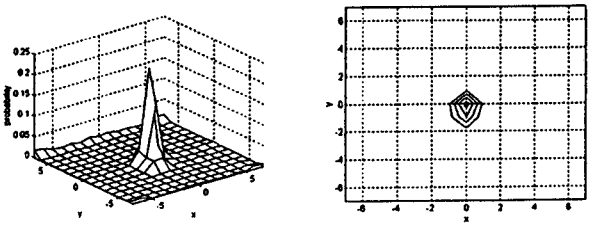


(a) 큰 움직임 (b) 작은 움직임

그림 1. 블록에서 움직임과 BFD 의 상호관계



(a) MV 분포(Miss America) (b) 2 차원 분포



(c) MV 분포(Football) (d) 2 차원 분포

그림 2. 움직임벡터(MV)의 통계적 특성

동영상에서 움직임벡터의 분포를 살펴보면, 그림 2와 같이 대부분 중심 부근에 집중되어 있다. 특히, 수평과 수직선상으로 많은 움직임이 있음을 알 수 있다. 따라서 탐색패턴은 이와 같은 특성을 고려한 다이아몬드형의 구조일 때 가장 효과적이다.

BFD가 임계값보다 크면 탐색을 수행하기 전에 그림 3과 같은 초기탐색 패턴을 사용한다. 그림 3에서 각 탐색패턴은 탐색패턴의 주변에 있는 초기탐색에서 빠진 위치를 다음 단계에서 점검하게 된다. 만약 다음 단계에서 초기탐색 패턴에서 찾은 최소값보다 더 작은 값을 가지는 위치를 주변에서 찾게 되면 그 위치가 그림 4에 있는 탐색중심의 새로운 후보가 된다.

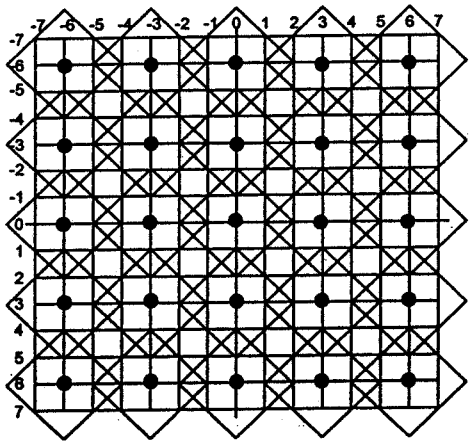


그림 3. 큰 움직임블록을 위한 초기 탐색패턴

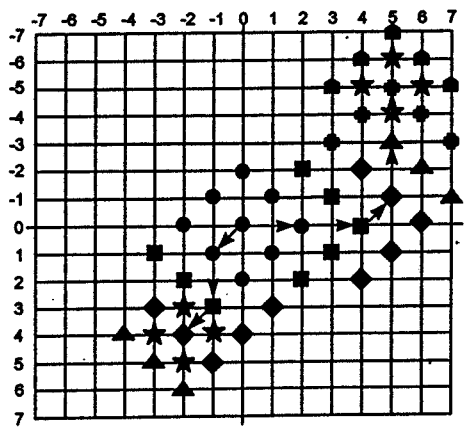
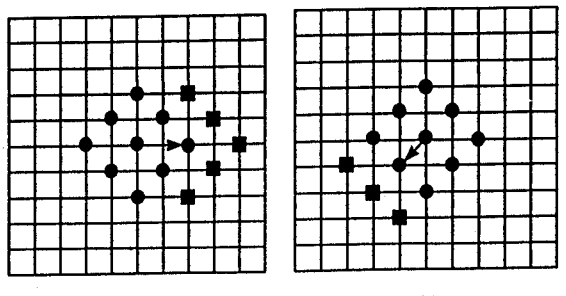


그림 4. 제안한 방법의 탐색패턴



(a) (b)

그림 5. 큰 움직임 블록에 대한 탐색패턴

그림 4는 최대 탐색영역의 크기가 ±7일 때, 탐색방법에 대한 두 가지 예를 보여 주고 있다. 탐색이 다음 단계로 진행됨에 따라 3개 혹은 5개의 후보 탐색점이 추가된다. 그림 4의 오른쪽 예에서 볼 수 있듯이, 탐색이 진행되면서 탐색방향을 급격히 변화할 수도 있다. 즉, 만약 지역 최소값에 빠지지 않으면 탐색영역 전체에서 최소값을 찾게 된다.

제안한 방법은 다음과 같이 정리할 수 있다.

1단계: 원점(0,0)을 중심으로 9개의 탐색점에 대하여 최소값을 찾는다. 탐색패턴은 수평과 수직 방향으로는 2점의 간격을 두고 배치되며, 대각선상에 있는 다른 점들은 1점 간격으로 배치된다. 만약 최소값이 원점에서 발견되면 3단계로 진행한다.

2단계: 이 단계에서는 이전 단계의 최소값 점의 위치에 따라서 두 종류의 탐색패턴을 가진다. 이 단계에서는 현재 탐색 패턴의 원점에 최소값이 이를 때까지 반복적으로 수행된다.

a) 만약 이전 단계의 최소값 점이 이전 탐색영역의 수평 혹은 수직 방향에 있는 것이라면, 그 점을 따라서 5개의 탐색점이 다음 단계의 후보 탐색점이 된다. 이 과정은 그림 5(a)에 그려져 있다.

b) 만약 이전 단계의 최소값 점이 이전 탐색영역의 대각선 방향에 있는 것이라면, 그 점을 따라서 3개의 탐색점이 다음 단계의 후보 탐색점이 된다. 이 과정은 그림 5(b)에 그려져 있다.

3단계: 이전 단계에서 찾은 탐색점을 중심으로 1점 간격의 주변 9개 탐색점 중에서 최소값의 위치를 찾는다.

만약 블록의 움직임이 작으면, 탐색패턴이 그림 6과 같이 좁은 지역으로 제한된다.

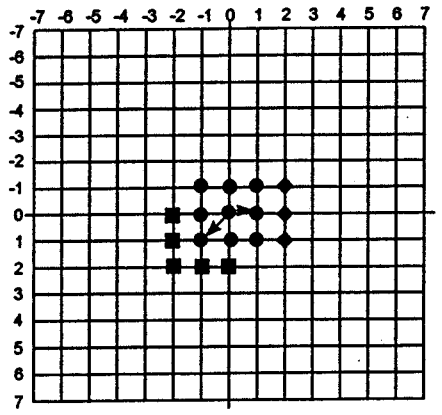


그림 6. 작은 움직임 블록에 대한 탐색패턴

고속 블록정합 방법에서 고려할 중요 사항들은 다음과 같다.

a) 지역적 최소값에 빠지지 말아야 한다.
b) 정확한 움직임 방향으로 탐색을 진행하기 위하여 초기 탐색점이 충분히 많이 배치되어야 한다.
c) 다음 단계에서 추가되는 탐색점의 갯수는 감소해야 하며, 이전 단계에서 검사된 탐색점은 제외되어야 한다.

4. 실험결과

ITU-T 시험영상인 MISS AMERICA, CLAIRE와 FOOTBALL 영상 각각 88 프레임을 사용하여 컴퓨터 모의시험을 수행하였다. 각 영상은 CIF 형식 (352x288 화소)의 휘도성분을 가진다. 프레임율은 원래는 30Hz인데, 모의시험에서는 10Hz로서 1번째, 4번째, 7번째, ..., 88번째 프레임을 사용하였다. 영상 화질은 다음과 같이 정의된 PSNR(peak signal-to-noise ratio)을 사용하여 비교하였다.

$$MSE = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}(I_t(m,n) - \tilde{I}_t(m,n))^2$$

$$PSNR = 10 \log_{10} \frac{255^2}{MSE} \quad [dB] \qquad (6)$$

여기서 CIF 형식에 대해서 M=352, N=288, BS (Block Size)=16 이다. I_t 는 원래 현재 화면을 나타내고 $\tilde{I}_t$ 는 움직임보상 예측된 현재 화면을 나타낸다.

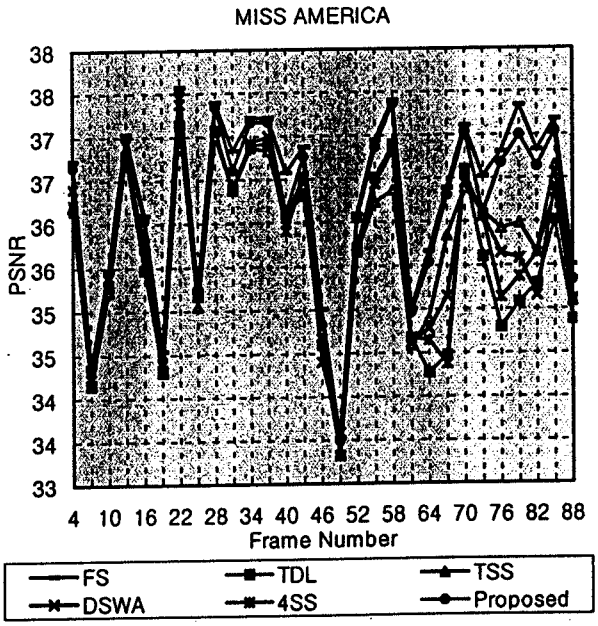


그림 7. 성능 비교

그림 7은 MISS AMERICA 영상에 대해 수행된 결과를 나타내고 있는데, 제한된 방법이 FS (full search), TDL (two-dimensional logarithmic search)[1], TSS (three-step search)[3], 4SS (four-step search)[5], 그리고 DSWA (dynamic search window adjust and interlaced search)[6] 방법들과 비교되어 있다. 제안한 방법과 전역탐색 방법의 예측 결과는 70 번째의 화면을 복원할 때까지는 거의 동일하지만, 70 번째에서 80 번째까지의 화면을 복원할 때에는 전역탐색 방법의 예측 결과보다 약간 떨어진다. 이에 비하여, 다른 고속 블록정합 방법들의 성능은 전역탐색 방법의 예측 결과와 상당히 큰 차이를 가진다. 즉, 제안한 방법이 움직임이 많은 화면들의 예측에서 보다 향상된 성능을 가짐을 알 수 있다.

표 1은 이러한 결과를 정리한 것인데, 제안된 방법이 전역탐색 방법에 매우 근사한 움직임 예측 성능을 가지면서, 다른 고속탐색 방법보다 평균 탐색점 수를 많이 줄임을 알 수 있다. 식 (5)에서 정의된 정합기준은, 탐색영역에 있는 매 블록의 후보 위치에 대한 정합값을 계산할 때 블록을 이루는 화소들 중에서 영향력이 큰 것들만을 고려함으로써 연산 회수를 감소시켜 움직임예측을 위한 전체 계산량을 줄이는데 기여한다.

표 1. 블록정합 방법들의 평균 성능 비교

추정 방법	PSNR(dB)	탐색점 수
전역탐색(FS)	36.29	225
3 단계 탐색(TSS)	35.78	25
4 단계 탐색(4SS)	35.81	20.91
2 차원 로그 탐색(TDL)	35.62	16.62
동적영역 가변탐색(DSWA)	35.77	19.79
제안한 방법	36.17	14.78

(주의) 블록크기 =16, 탐색영역 = [-7, +7]

5. 결론

본 논문에서는 새로운 정합방법을 사용한 고속 블록정합 방법을 제안하였다. 다양한 정합기준을 결합하여 효과적인 정합기준을 사용하였고, 각 블록별 움직임을 관측하여 서로 다른 탐색패턴을 적용하였다. 새로운 알고리즘은 필요한 계산량을 효과적으로 줄이면서 우수한 예측성능을 보였다.

감사의 글

본 연구는 광주과학기술원(K-JIST) 초고속광네트워크(UFON) 연구센터를 통한 한국과학재단 우수연구센터(ERC) 지원금에 의한 것입니다.

참고문헌

[1] J.R. Jain and A.K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Trans. Commun.*, vol. COM-29, pp. 1799-1808, Dec. 1981.

[2] H.G. Musmann, P. Pirsh and H.J. Grallert, "Advances in picture coding," *Proc. IEEE*, vol. 73, no. 4, pp. 523-548, Apr. 1985.

[3] R. Srinivasan and K.R. Rao, "Predictive coding based on efficient motion estimation," *IEEE Trans. Commun.*, vol. COM-33, no. 8, pp. 888-896, Aug. 1985.

[4] M. Ghanbari, "The cross-search algorithm for motion estimation," *IEEE Trans. Commun.*, vol. COM-38, pp. 950-953, July 1990.

[5] L.M. Po and W.C. Ma, "A novel four-step search algorithm for fast block motion etsimation," *IEEE Trans. Circuit and Systems for Video Technology*, vol. 6, no. 3, pp. 313-317, June 1996.

[6] L.W. Lee, et al., "Dynamic search-window adjustment and interlaced search for block-matching algorithm," *IEEE Trans. Circuit and Systems for Video Technology*, vol. 3, no. 1, Feb. 1993.