논문 99-24-5B-17 한국통신학회논문지 '99-5 Vol.24 No.5B

움직임 보상형 프레임간 내삽을 위한 적응적 움직임 추정 기법

준회원 성 원 락* 정회원 강 응 관*, 김 동 욱**, 최 종 수*

Adaptive motion estimation technique for motion compensated interframe interpolation

Won Rak Sung*, Eung Kwan Kang*, Dong Wook Kim**, Jong Soo Choi* *Regular Members*

요 약

프레임간 내삽은 동영상에서 두 장의 연속한 프레임 사이에 하나 이상의 프레임을 삽입하는 기법이다. 기존의 움직임 보상형 프레임간 내삽 기법은 수신측에서 프레임 내삽을 위한 영역의 예측을 위해 새로운 탐색을 행함으로써 계산량이 증가할 뿐만 아니라 블록 단위의 움직임 벡터를 이용해 건너뛴 프레임을 내삽하기 때문에 내삽된 영상에서 심한 블록 효과가 나타난다. 본 논문에서는 수신측에서 행하는 과도한 탐색에 의한 계산량을 줄일 수 있는 고속 움직임 추정 기법과 내삽된 영상의 블록 효과를 줄일 수 있는 프레임간 내삽 기법을 제안한다.

ABSTRACT

Interframe interpolation is the technique that inserts one or more frames between two consecutive frames in video sequence. Conventional motion compensated interframe interpolation algorithm requires additional computation because it does new search procedure at the receiver for region prediction needed for interframe interpolation. Also, the algorithm causes severe block effect because it interpolates the skipped frames by using block based motion vectors. Therefore, in this paper, we propose the new algorithms that reduce the computational complexity by the excessive search at the receiver and the block effect in the interpolated frames.

I. 서 론

프레임간 내삽은 두 장의 기준프레임 사이에 하나 이상의 프레임을 삽입하는 기법으로, 동영상 압축이나 프레임율 변환 또는 느린 화면 재생을 위한 목적으로 사용될 수 있다[1][2]. 프레임율 변환은 프레임율이 서로 다른 이기종간의 동영상 통신을 위해 필요한 기술이며, 기존의 동영상을 느린 속도로 재생할 경우에도 일정한 프레임율을 유지하기 위해서는 프레임간 내삽이 필요하다. 내삽 기법은 동영상 전송시 전송비트율을 낮추기 위한 목적으로도 사용될 수 있는데, 송신측에서는 몇 장의 프레임을 건너 뛰면서 부호화해 전송하고, 수신측에서 건너뛴 프레임을 재구성한다.

따라서, 프레임간 예측 부호화에서 움직임 추정은 중요한 역할을 한다. 정확한 움직임 정보를 얻기 위해, 다양한 움직임 추정 알고리즘이 광범위하게 연구

* 중앙대학교 전자공학과(jschoi@candy.ee.cau.ac.kr)
** 전주대학교 전자매체공학부
논문번호: 98469-1027, 접수일자: 1998년 10월 27일

되고 개발되어 왔다. 성공적인 움직임 추정을 위한 관건은 동영상에서 존재하는 시간적 상관성을 효과적으로 줄이는 것인데, 이러한 움직임 추정 알고리즘 중에서 하드웨어 구현의 용이성 측면에서 블록 정합 기법이 MPEG, H.261, H.263 등의 많은 동영상 압축 표준에 의해 채택되어 왔다. 그 중 전역 탐색 블록 정합 알고리즘(Full search block matching algorithm : FSBMA)은 움직임 추정을 위해 가장 단순한 기법이지만 계산량이 많아진다는 단점이 있는데 특히 탐색영역이 커지면 커질수록 더욱 더 많은 계산적 부하를 요구하게 된다. 또한, 3단계 탐색(Three-step search : TSS)[3], OTS(One-at-a-time)[4], 2-D logarithmic search[5] 등 탐색 영역 내의 탐색 점들의 수를 제한함으로써 계산량을 줄이는 많은 고속 알고리즘들이 제안되어 왔다. 이들 고속 알고리즘들은 고정된 탐색 영역에서 탐색을 시작하며 제한된 수의 탐색점을 사용해서 움직임 추정을 행하기 때문에 추정 정확도가 떨어지게 된다.

한편, MPEG과 같은 일반적인 동영상 부호화 시스템을 고려할 때, 순방향 움직임 벡터는 예측프레임의 재구성을 위해 송신측으로부터 전송 받을 수 있지만, 역방향 벡터는 그렇지 않다. 따라서 수신측에서 역방향 벡터를 얻기 위해서는 전역 탐색 블록 정합 알고리즘과 같은 별도의 탐색과정이 필요한데, 이는 지나치게 많은 계산량을 요구하며, 고속 탐색 알고리즘을 사용해 움직임 추정을 행하게 되면, 제한된 수의 탐색점 때문에 국부 최소값(local minima)에 봉착하기 때문에 움직임 추정의 정확도가 떨어지게 된다. 또한 이들 알고리즘들은 동영상의 시공간적 상관성을 고려하지 않기 때문에 잘못된 예측 결과를 가져오게 된다.

따라서 본 논문에서는, 프레임간 내삽을 위해 송신측으로부터 수신된 이미 알고 있는 순방향 움직임 벡터를 이용해서 역방향 움직임 벡터를 간단히 얻을 수 있는 새로운 고속 알고리즘을 제안한다. 고정된 탐색점의 수를 사용하는 기존의 고속 알고리즘과는 달리 제안하는 알고리즘은 동적인 탐색 영역 내에서 움직임 추정을 행하게 된다.

또한 내삽된 영상의 블록 효과를 줄이기 위해 움직임 경계에 위치한 블록을 영역 분할하고 분할된 영역이 같은 블록에 속할지라도 서로 다른 움직임을 가질 수 있게 함으로써, 기존의 블록 기반형 움직임 보상형 프레임간 내삽 기법에서 나타나는 블록 효과를 줄이는 방법을 제안한다.

II장에서는 동적인 탐색 영역을 이용해 전송된 순방향 움직임 벡터로부터 역방향 움직임 벡터를 구하는 방법에 대해 설명하고, III장에서는 움직임 경계에 위치한 블록을 영역 분할하고 각 영역들에 대해 움직임 벡터를 할당하는 방법을 제시한다. 그리고, IV장에서는 실험 결과를 보이며, 마지막으로 V장에서 결론을 맺는다.

II. 역방향 움직임 벡터의 추정

위에서도 언급했듯이 건너뛴 프레임들을 내삽하기 위해서는 프레임내의 각 화소들은 네 영역(정지 영역, 가려지는 영역, 드러나는 영역, 움직이는 영역)으로 적절히 잘 분류되어야 하며, 각 영역들은 서로 다른 방법으로 처리가 되어야 한다[1][6][7]. 일반적인 동영상 부호화 기법을 고려해볼 때, 수신측은 송신측으로부터 순방향으로 추정된 움직임 벡터만을 전달받는다. 여기서 순방향 움직임 벡터란, 시간적인 순서로 볼 때, 앞에 위치한 프레임을 참조 프레임으로 하여 뒤에 올 프레임을 예측하기 위해 사용되는 움직임 벡터를 뜻하고, 역방향 움직임 벡터란 그 반대를 의미한다. 그런데, 순방향으로 추정된 움직임 벡터만을 사용할 경우, 그림 1에서 볼 수 있는 것처럼 가려지는 영역은 잘 예측할 수 있는 반면 드러나는 영역은 어렵게 된다. 따라서, 가려지는 영역뿐 아니라 드러나는 영역을 예측하기 위해서는 순방향 움직임 벡터뿐만 아니라 역방향 움직임 벡터도 필요하게 된다. 그러나, 기존의 전역 탐색 블록 정합 알고리즘이나 고속 알고리즘 같은 움직임 추정 기법을 사용할 경우 위에서 언급한 문제점을 나타내게 된다. 그러므로 본 논문에서는 송신측으로부터 받은 순방향 움직임 벡터로부터 쉽게 역방향 움직임 벡터를 얻는 방법을 제시한다.

그림 2에서, F_1, F_2는 송신측으로부터 전송 받은 이미 알고 있는 기준 프레임이고, $MB_{F2}(i, j)$와 $MB_{F1}(i, j)$은 F_2프레임 내의 (i, j)번째 블록과 F_1프레임 내의 (i, j)번째 블록을 각각 의미한다. 그리고 $MV_{F2}(i, j)$는 F_2프레임 내의 블록 $MB_{F2}(i, j)$에 할당된 순방향 움직임 벡터이고 $MB_{F1}(i, j)$는 블록 $MB_{F2}(i, j)$가 참조하는 영역을 나타낸다. 그림 2에서 보듯이, F_2프레임 내의 블록 $MB_{F2}(i, j)$의 F_1프레임에서의 참조 블록은 대개 블록 격자들과 맞지 않는다. 그 대신 보통 네 개의 블록에 걸치게 된다. 먼저 $MB_{F1}(i, j)$에 의해 겹쳐지는 블록들 즉, $OB_{F1}(k, l)$, $OB_{F1}(k, l+1)$, $OB_{F1}(k+1, l)$, $OB_{F1}(k+1, l+1)$에 $-MV_{F2}(i, j)$를 할당한다. 그리고 이러한 조작을 F_2프

레임에 속하는 모든 블록에 대해서 반복하면 F_1프레임의 각각의 블록들에는 여러 개의 후보 벡터가 할당된다. 이 후보 벡터들을 이용해 탐색 영역을 결정하게 된다.

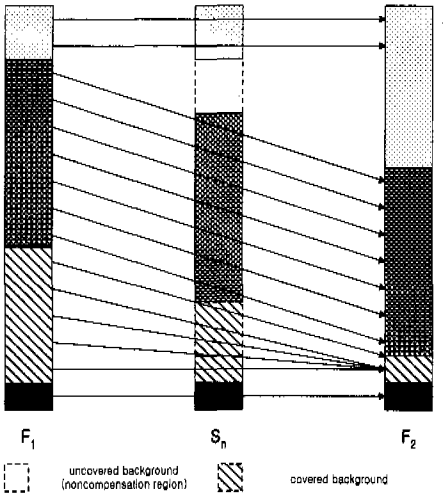


그림 1. 순방향 움직임 벡터를 사용한 프레임간 내삽

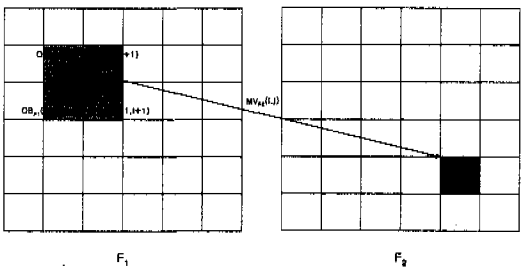


그림 2. 전송된 순방향 벡터와 참조 블록

우선, 새로운 탐색 영역을 (X_1, X_2, Y_1, Y_2)라고 가정을 하면 이 새로운 탐색 영역을 사용해서 F_1프레임의 블록들에 대해서 역방향 움직임 벡터가 추정된다. 여기서 $X_1 < X_2$, $Y_1 < Y_2$이고 (X_1, X_2)와 (Y_1, Y_2)는 각각 수직, 수평 방향으로의 탐색 영역의 시작점과 끝점을 나타낸다. 그리고, F_1프레임의 좌표점 (x, y)에 위치한 (i, j)번째 블록 MB(i, j)의 후보 움직임 벡터들을 $\{x_k(i, j), y_k(i, j)\}$라고 가정한다. 이 때 k = 1, 2, , n이고 n은 (i, j)번째 블록에 할당된 후보 벡터들의 수를 나타낸다. 그러면, (i, j)번째 블록의 탐색 영역을 다음과 같이 나타낼 수 있다.

$$X_1(i,j) = x + \min\{ x_k(i,j)\} \quad (1)$$

$$X_2(i,j) = x + N + \max\{ x_k(i,j)\} \quad (2)$$

$$Y_1(i,j) = y + \min\{ y_k(i,j)\} \quad (3)$$

$$Y_2(i,j) = y + N + \max\{ y_k(i,j)\} \quad (4)$$

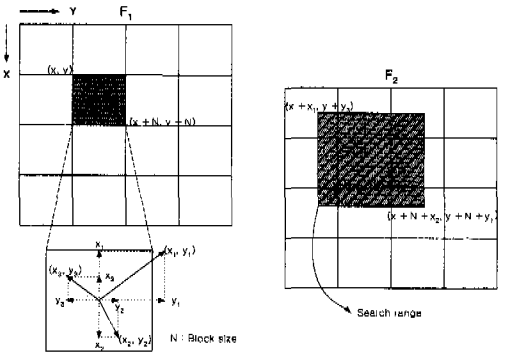


그림 3. 탐색 영역의 결정

그림 3에 예를 들었는데 여기서 후보 벡터들은 (x_1, y_1), (x_2, y_2), (x_3, y_3)이며 $X_1 = x + x_1$, $X_2 = x + N + x_2$, $Y_1 = y + y_3$, $Y_2 = y + N + y_1$으로 결정된다. 여기서 N은 블록의 크기를 의미한다.

식 (1) ~ (4)는 F_1프레임의 임의의 한 블록의 탐색 영역은 송신측으로부터 받은 이미 알고 있는 순방향 움직임 벡터들인 후보 벡터들 중 최대와 최소의 움직임을 보이는 성분들에 의해 결정된다는 것을 의미한다. 그 결과 제안한 알고리즘은 비교적 적은 계산량을 가지고도 높은 움직임 추정 정확도를 보이게 된다. 특히 영상 회의와 같은 응용 분야에서는 배경 부분이 영상 내에 많은 부분을 차지하기 때문에 제안한 알고리즘의 효율성은 매우 높아진다. 또한 송신측으로부터 받은 순방향으로 추정된 움직임 벡터 중에서 역방향으로도 그 움직임이 일치하는 움직임 벡터만을 선택하게끔 탐색 영역이 결정되기 때문에, 이 과정에서 국부 최소값 때문에 잘못 구해진 움직임 벡터는 제거가 된다.

그림 4는 실험 결과인데 Salesman영상 중 비교적 큰 움직임을 보이는 10번과 12번 영상 사이에서 추정된 움직임 벡터 필드이다. 여기서 (a)는 FSBMA에 의해 추정된 역방향 움직임 벡터이고, (b)는 TSS에 의해 구해진 역방향 움직임 벡터이며, (c)는 제안한 알고리즘에 의해 얻어진 역방향 움직임 벡터이다. 그

림에서 알 수 있듯이, TSS 같은 고속 탐색 알고리즘은 제한된 탐색점에 대해 탐색을 행함으로써 국부 최소값으로 인한 오정합이 많이 발생한 것을 알 수 있다. 또한 FSBMA의 경우에는 탐색 범위내의 모든 점에 대해 탐색을 행하기 때문에 보다 나은 결과를 보이지만 물체의 빠른 움직임에 의한 흐려짐이 나타나는 부분에서는 역시 국부 최소값에 의한 오정합이 제거되지 않는 것을 알 수 있다. 그러나, 제안한 방법은 실제 물체가 움직인 영역 부근으로 탐색 범위가 결정되므로 오정합이 거의 모두 사라진 것을 알 수 있다.

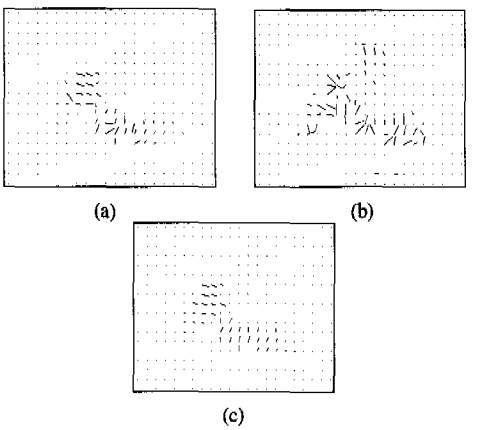


그림 4. 각 알고리즘에 의해 추정된 역방향 움직임 벡터 (a) FSBMA (b) TSS (c) 제안한 알고리즘

한편, 제안한 알고리즘을 앞에서 얻어진 역방향 움직임 벡터에 대해 다시 한번 적용함으로써 교정된 순방향 움직임 벡터를 얻을 수 있다.

Ⅲ. 움직임 경계 블록의 영역 분할

위에서도 언급한 것처럼, 물체의 움직임이 있을 때 움직이는 물체의 경계에서는 드러나는 영역과 가려지는 영역이 발생한다. 그런데 기존의 블록 기반 움직임 보상형 프레임간 내삽 기법은 동일한 블록 내에 있는 모든 화소들이 동일한 움직임을 갖는다는 가정 아래 건너뛴 프레임을 내삽한다. 움직이는 물체의 경계에 위치한 블록은 동일 블록 내에 서로 다른 둘 이상의 움직임을 포함할 수 있음에도 불구하고 하나의 움직임 벡터로 표현하기 때문에 드러나는 영역과 가려지는 영역을 제대로 보상할 수 없고, 이로 인해 블록 효과가 심하게 나타나게 된다[8]. 그러므로 움직임 경계에 위치한 블록들에 대해서만 밝기값에 근거해 영역을 분할하고 영역별로 서로 다른 움직임을 가질 수 있게 한다면 드러나는 영역과 가려지는 영역을 잘 표현할 수 있어 블록 효과를 감소시킬 수 있다.

블록 효과는 원래의 영상과 움직임 보상 영상의 차가 상대적으로 큰 블록에서 발생한다고 볼 수 있다. 움직이는 물체의 경계 부근에서 화소들의 DFD 값이 크게 나타나게 된다. 따라서 프레임 내에 있는 어떤 블록이 움직임 경계 블록인지 아닌지를 판단하기 위해 다음과 같은 식을 이용할 수 있다.

$$|Diff(\ MB_F(i,j),\ MV_F(i,j))| \geq TH \tag{5}$$

이때, *Diff(a, b)*는 영역 *a*와, 영역 *a*가 움직임 벡터 *b*를 가지고 시간적으로 뒤에 위치한 프레임에서 참조하는 영역 사이의 차를 의미한다. 여기서 *Diff*값이 문턱치 TH보다 크면, 그 블록은 움직임 경계에 위치한 블록이라고 판단하고 이를 움직임 경계 블록이라 부르기로 한다.

일단, 어떤 블록이 움직임 경계 블록으로 판정되면 그 블록은 몇 개의 영역으로 분할된다. 분할된 각 영역의 움직임은 탐색에 의한 정합 방법을 이용해서 얻을 수 있으나, 앞에서도 언급한 바와 같이 이는 지나치게 많은 계산을 요구한다. 따라서 본 장에서는 2장에서 구한 블록에 기반한 순방향, 역방향 움직임 벡터를 이용해 간단하게 움직임 경계 블록 내에 있는 영역들의 움직임 벡터를 구하는 방법을 제안한다. 그림 5에서, 영역 분할된 움직임 경계 블록 MB(i, j)는 다음과 같이 나타낼 수 있다.

$$MB(i,j) = \sum_{k=1}^{K} R(i,j,k) \tag{6}$$

여기서 R(i, j, k)는 블록 MB(i, j)내에 있는 k번째 영역을 나타낸다. 영역 R(i, j, k)의 움직임 벡터 MV_R(i, j, k)는 아래와 같이 구할 수 있다.

$$NMV(i,j) = \{MV(i-1,j), MV(i,j-1), MV(i,j+1), MV(i+1,j)\} \tag{7}$$

여기서 NMV(i, j)는 블록 MB(i, j)의 4근방 블록들의 움직임 벡터들로 이루어진 후보 움직임 벡터들의 집합을 의미한다.

$$\text{if } R(i,j,k) \in MB(i,j),\quad MV_R(i,j,k) \in \{MV(i,j) \bigcup NMV(i,j)\} \tag{8}$$

즉, 블록 MB(i, j)내의 각 영역들은 MB(i, j)의 이웃한 블록 혹은 그 자신의 움직임 벡터로 보상될 수 있음을 뜻한다. 식 (8)을 만족시키면서 블록 MB(i, j)내의 각 영역에 대해 *Diff*를 최소화시키는 벡터를 그 영역의 벡터 $MV_R(i, j, k)$로 선택한다.

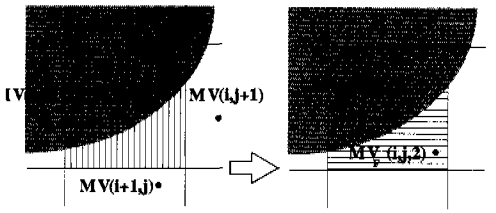


그림 5. 움직임 경계 블록 내에 있는 영역의 움직임 벡터 추정

2장에서 얻은 순방향 움직임 벡터와 역방향 움직임 벡터 그리고 3장에서 얻은 움직임 경계 블록의 영역별 움직임 정보를 이용해서 건너뛴 프레임을 내삽한다. 내삽하는 방법의 개요를 그림 6에 간략히 보인다. 2장과 3장에서 얻은 순방향 움직임 벡터들을 이용해 내삽될 프레임을 재구성할 경우에는 가려지는 영역이 보상이 되지 않으므로 가려지는 영역을 추정할 수 있으며 또한 2장, 3장에서 구한 역방향 움직임 벡터를 이용해 내삽될 프레임을 재구성하게 되면 드러나는 영역이 보상이 되지 않으므로 쉽게 드러나는 영역을 추정할 수 있다. 드러나는 배경 영역은 F_2프레임에서, 가려지는 배경 영역은 F_1프레임에서 그 값을 그대로 가져와 내삽한다.

한편 그림 6에서 알 수 있듯이 정적인 배경 영역은 순방향, 역방향 움직임 벡터 모두 영벡터를 가진다. 그리고 그 나머지 영역은 움직이는 물체에 해당한다. 이 마지막 두 영역의 경우에는 다음 식과 같이 두 참조 프레임 즉, F_1, F_2프레임을 이용하여 내삽될 프레임의 화소값과 위치가 결정된다.

$$I(x_{s_n}, x_s) = \frac{(N+1-n)\cdot I(x_{F_1}, F_1) + n \cdot I(x_{F_2}, F_2)}{N+1} \quad (9)$$

$$x_{s_n} = \frac{(N+1-n)\cdot x_{F_1} + n \cdot x_{F_2}}{N+1}, \quad n = 1, \ldots, N \quad (10)$$

여기서 $I(x_{sn}, S_n)$은 프레임 S_n에서 x_{sn}에 위치한 화소의 밝기값을, x_{sn}은 내삽될 화소의 위치를 의미한다. 그리고 n은 내삽될 프레임이 위치하는 순서를 의미한다.

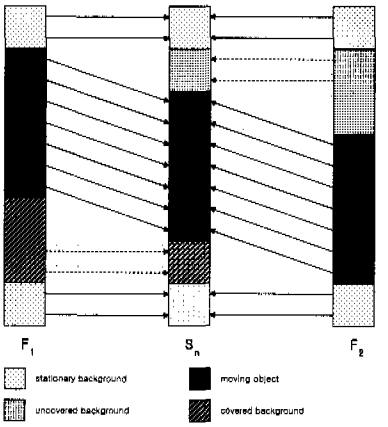


그림 6. 건너뛴 프레임의 재구성

Ⅳ. 실험 결과

본 장에서는 제안한 기법의 성능을 평가하기 위해 288 ×352 크기의 Salesman영상의 1번 프레임에서 100번 프레임과 Clair영상의 100번 프레임에서 130번 프레임을 대상으로 실험을 행했다. Salesman영상은 상품을 설명하는 비교적 큰 움직임을 보이는 영상이며, Clair영상은 작은 움직임을 보이는 head-and-shoulder영상이다.

제안한 기법을 FSBMA와 TSS알고리즘과 비교하였다. 순방향 움직임 벡터는 16 × 16 블록으로 FSBMA와 TSS에 의해 움직임 추정되었다. 실험 결과의 객관성을 보장하기 위하여 각각 FSBMA, TSS에 의해 움직임 벡터를 추정한 후, 두 경우 모두 3장에서 기술한 알고리즘을 적용하여 내삽을 행하여 내삽된 영상에 생기는 블록 효과를 줄이고자 하였다. 그리고 한 장의 프레임을 건너뛴 N = 1인 경우와 두 장의 프레임을 건너뛴 N = 2인 경우에 대해 각각 실험하였다. N = 1의 경우에는 탐색 영역을 15로 하였으며, N = 2인 경우에는 탐색 영역을 31로 하여 정소 화소 단위로 움직임 추정을 하였다. 그리고 움직임 경계 블록의 판정을 위한 문턱치 TH는 실험을 통해 적절한 상수 값을 선택하였다.

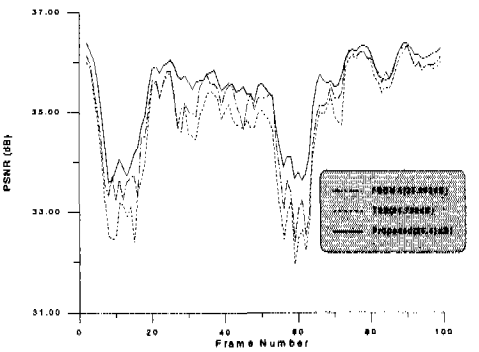
한편, 움직임 경계 블록으로 판단된 블록의 분할을 위해서는 일반적인 영역 분할(region segmentation)방법을 사용할 수 있으며 본 실험에서는 영역 병합(region merge)법을 사용하였다. 따라서, 움직임

경계 블록의 영역 분할 결과 생기는 영역의 개수는 그 블록의 특징에 따라 다르게 나오게 된다.

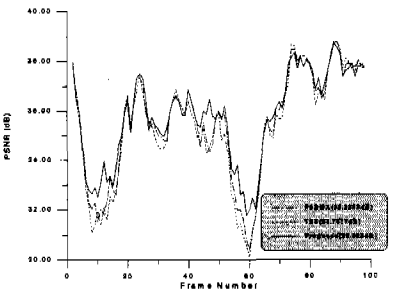
본 실험에서 건너뛴 프레임을 내삽하기 위해 필요한 참조 프레임으로는 내삽하고자 하는 프레임의 전, 후 프레임을 원 영상 그대로 사용하였다. 예를 들어, N = 1인 경우 10번 프레임을 내삽하기 위해서는 9번과 11번 프레임을 기준 프레임으로 사용했으며, 11번 프레임을 내삽하기 위해서는 10번과 12번 프레임을 기준 프레임으로 사용하였다. N = 2인 경우에는 먼저 내삽되는 프레임(n = 1인 내삽 프레임)과 나중에 내삽되는 프레임(n = 2인 내삽 프레임)으로 나누어 성능평가를 실시했다.

기존의 방법과 제안한 방법의 객관적인 영상의 화질을 비교하기 위해 PSNR(peak signal-to-noise ratio)을 사용하였다. 그림 7에 Salesman영상에 대해 실험한 PSNR을 보였다. 그림 7에서 알 수 있듯이, 제안한 방법과 기존의 방법의 PSNR차이는 10번과 60번 프레임 근처의 비교적 움직임이 큰, 즉 드러나는 영역과 가려지는 영역이 많이 나타나는 프레임 근처에서 크게 나는 것을 볼 수 있다.

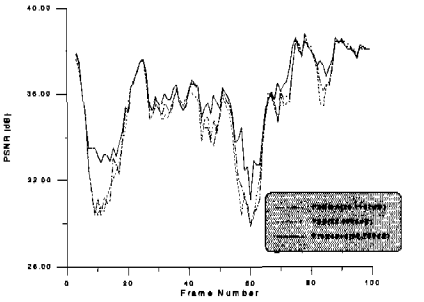
그림 8은 주관적 화질을 비교하기 위해, 기존의 방법과 제안한 방법에 의해 내삽된 영상을 보인 것이다. (b), (c), (d)는 기존의 방법과 제안한 방법에 의해 내삽된 Clair영상의 102번 영상이다. 기존의 고속 탐색 알고리즘으로 추정된 움직임 벡터에 의해 내삽된 영상의 화질이 심하게 열화되었음에 비해 제안한 기법은 움직임을 제대로 추정하기 때문에 내삽된 영상이 좋은 결과를 보인다는 것을 알 수 있다. 제안한 방법의 우수함을 보다 효과적으로 보이기 위해 그림 9에 Salesman영상의 오른손에 들고 있는 상품 부분, 넥타이 부분을 확대한 영상을 보였는 데, 제안한 기법의 내삽된 영상이 기존의 방법에 비해 실제 움직임에 보다 더 적응적임을 알 수 있다.



(a) N = 1

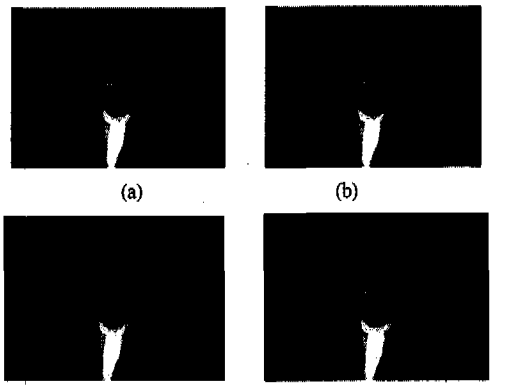


(b) N = 2, n = 1



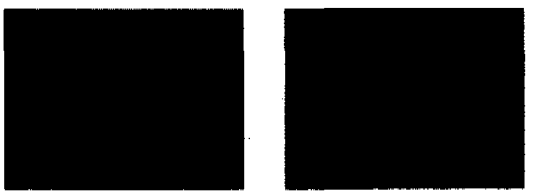
(c) N = 2, n = 2

그림 7. 내삽된 프레임의 PSNR(Salesman 영상)

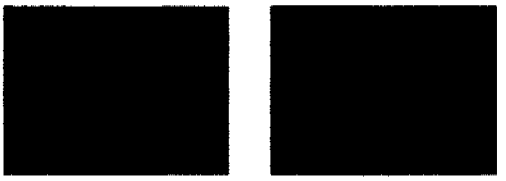


(a) (b) (c) (d)

그림 8. 내삽된 프레임(N = 1) (a) 원 영상 (b) FSBMA (c) TSS (d) 제안한 방법



(a) (b)



(c) (d)

그림 9. 내삽된 프레임의 확대 영상 (a) 원영상 (b) FSBMA (c) TSS (d) 제안한 방법

V. 결 론

본 논문에서는 새로운 움직임 보상형 프레임간 내삽 기법을 제안하였다. 순방향 움직임 벡터와 역방향 움직임 벡터를 함께 사용함으로써 가려지는 영역과 드러나는 영역을 보다 더 정확히 예측, 보상함을 알 수 있었다. 실제 물체의 움직임에 보다 더 가까운 역방향 움직임 벡터를 얻기 위해 동적인 탐색 영역을 사용하였으며, 또한 다시 한 번 이것을 적용함으로써 순방향 움직임 벡터를 교정하였는데, 구해진 움직임 벡터는 실제 움직임에 매우 적응적임을 알 수 있었다. 그리고 본 논문에서는 기존의 블록 기반 움직임 보상형 프레임간 내삽 기법에서 발생하는 블록 효과를 감소시키기 위한 방법을 제안하였다. 즉, 움직임 경계 블록에 위치한 블록을 영역 분할해서 동일한 블록에 속했을지라도 서로 다른 움직임을 가질 수 있게 하였다. 실험 결과 객관적인 성능 뿐만 아니라 주관적인 화질 측면에서도 제안한 방법의 우수성을 확인할 수 있었다.

참 고 문 헌

[1] C. L. Huang and T. T. Choa, Motion-compensated interpolation for scan rate up-conversion, *Optical Engineering*, vol. 35, no. 1, pp. 166-176, Jan., 1996.

[2] P. Migliorati and S. Tubaro, Multistage motion estimation for image interpolation, *Signal Processing : Image Communication*, vol. 7, pp. 187-199, 1995.

[3] T. Koga, K. Iinuma, A. Hirano and T. Ishiguro, Motion-compensated interframe coding for video conferencing, in National Telecommunications Conference, pp. G5.3.1 -G5.3.5, 1981.

[4] R. Srinivasan and K. Rao, Predictive coding based on efficient motion estimation, *IEEE Trans. Comm.*, vol. COM-33, pp. 888-896, Aug., 1985.

[5] B. Furht, J. Greenberg and R. Westwater, *Motion estimation algorithms for video compression*, Kluwer Academic Publishers, 1997.

[6] J. R. Corbera and J. Sklansky, Interframe interpolation of cinematic sequences, *J. Visual Communication and Image Representation*, vol. 4, no. 4, pp. 392-406, Dec., 1993.

[7] P. Csillag and L. Boroczky, Estimation of accelerated motion for motion-compensated frame interpolation, *Proc. SPIE Visual Comm. and Image Proc.*, vol. 2727, pp. 604-614, Orlando, Florida, 17-20 Mar., 1996.

[8] C. K. Wong and O. C. Au, Fast motion compensated temporal interpolation for video, *Proc. SPIE Visual Comm. And Image Proc.*, vol. 2501, pp. 1108-1118, Taipei, Taiwan, 24-26 May 1995.

성 원 락(Won-Rak Sung) 준회원

1997년 2월 : 중앙대학교 전자공학과 졸업(공학사)

1999년 2월 : 중앙대학교 대학원 전자공학과 졸업(공학석사)

<주관심 분야> 동영상 부호화, 영상처리

강 응 관(Eung-Kwan Kang) 정회원

1993년 2월 : 중앙대학교 전자공학과 졸업(공학사)

1995년 2월 : 중앙대학교 대학원 전자공학과 졸업(공학석사)

1995년 1월~1997년 5월 : 현대전자 정보통신연구소 연구원

1995년 9월~현재 : 중앙대학교 대학원 전자공학과 박사과정 재학 중

<주관심 분야> 동영상 부호화, 멀티미디어 데이터베이스, 영상통신, 컴퓨터비젼

김 동 욱(Dong-Wook Kim) 정회원

1987년 2월 : 성균관대학교 전자공학과 졸업(공학사)

1992년 2월 : 중앙대학교 대학원 전자공학과 졸업(공학석사)

1996년 8월 : 중앙대학교 대학원 전자공학과 졸업(공

학박사)
1998년 3월~현재 : 전주대학교 전자매체공학부 조교수
<주관심 분야> 영상통신, 신호처리

최 종 수(Jong-Soo Choi) 정회원
1975년 2월 : 인하대학교 전기공학과 졸업(공학사)
1977년 2월 : 서울대학교 대학원 전자공학과 졸업(공학석사)
1981년 2월 : 일본 Keio대학 전기공학과 졸업(공학박사)
1981년 2월~8월 : Aloka(주) 연구소 연구원
1981년 9월~현재 : 중앙대학교 전자공학과 교수
<주관심 분야> 적외선 신호처리, 컴퓨터 시각, 영상압축 및 영상처리 등임