쌍방향 움직임 예측을 이용한 움직임 보상 보간 기법에서 효율적인 움직임 벡터 보정 방법

준회원 박 지 윤*, 종신회원 이 창 우*

Efficient Motion Vector Correction Method in Motion Compensated Interpolation Technique Using Bilateral Motion Estimation

Ji-Yoon Park* *Associate Member*, Chang-Woo Lee* *Lifelong Member*

요 약

동영상 신호의 프레임율 증가를 위해서 움직임 보상 보간(motion compensated interpolation) 기법이 많이 사용된다. 특히 쌍방향 예측을 이용한 움직임 추정 기법은 움직임 추정 과정에서 빈 공간이나 겹쳐지는 문제를 해결함으로써 중간 삽입 프레임 생성 과정에서 좋은 성능을 보인다. 그러나 이와 같은 움직임 추정 과정에서 잘못된 움직임 벡터를 선택할 경우 왜곡된 블록을 생성할 수 있다. 본 논문에서는 쌍방향 움직임 예측을 기반으로 하는 움직임 보상 보간 기법의 움직임 추정 과정에서 선택되는 움직임 벡터가 올바른 추정인지를 판별하고 인접한 움직임 벡터와 병합한 블록을 이용하여 1/2 화소 단위로 움직임 벡터를 보정하는 새로운 기법을 제안한다. 또한, 제안하는 기법이 기존의 움직임 벡터 추정 기법에 비해서 우수한 성능을 보이는 것을 모의 실험을 통하여 확인한다.

Key Words : Motion compensated interpolation, Frame rate, Up conversion, Bidirectional motion estimation, Distributed video coding

ABSTRACT

The motion compensated interpolation method is widely used to increase video frame rates. Especially, the bilateral motion estimation technique provides the improved results, since it doesn't make the overlapping and missing blocks in the interpolated frame. However, the motion vectors, which are obtained by the bilateral motion estimation, sometimes require further correction. In this paper, we propose the efficient motion vector correction method for the bilateral motion estimation technique. By comparing the motion vectors of neighboring blocks and searching the new motion vector after merging the neighboring blocks, the erroneous motion vectors are efficiently corrected. It is shown that the proposed method provides better results, compared with the conventional methods.

Ⅰ. 서 론

부호화된 동영상 신호의 프레임율 증가 변환(frame rate up conversion)을 위해서 많은 연구가 진행되어 왔는데, 이러한 기법은 프레임율을 증가시킴으로써 화질의 향상을 가져올 수 있으며 최근 많

※ 본 연구는 2009년도 가톨릭대학교 교비 연구비 지원으로 이루어졌습니다.
* 가톨릭대학교 정보통신전자공학과(wmdnls79@catholic.ac.kr), (changwoo@catholic.ac.kr)
논문번호 : KICS2008-12-551 접수일자 : 2008년 12월 15일, 최종논문접수일자 : 2009년 6월 10일

은 연구가 진행되고 있는 DVC(distributed video coding)기법의 부가정보 (side information) 생성에 도 이용된다[1-5].

이를 위해 기존의 많은 프레임율 증가 방법이 제안되어 왔는데, 가장 기본적인 방법으로는 움직임을 고려하지 않고 이전 프레임과 현재 프레임의 같은 위치에 있는 화소값의 평균을 내어 중간 프레임을 생성하는 방법과 이전 화소값을 그대로 반복하여 사용하는 방법이 있다. 이러한 방법은 움직임이 거의 나타나지 않는 영상에서만 사용될 수 있는 한계를 가진다. 따라서 움직임을 고려한 중간 삽입 프레임 생성 방법이 필요하며 이를 위한 연구로 움직임을 고려한 움직임 보상 보간(motion compensated interpolation)방법이 제안되고 있다. 이를 위해서는 정확한 움직임 벡터를 찾는 것이 가장 중요하다[6-11].

Park 등은 중간 프레임 생성에서 순방향 움직임 추정기법을 이용하여 움직임 벡터를 우선적으로 선택하고, 인접한 블록들의 움직임 벡터를 후보로 선정하여 가장 일치하는 움직임 벡터를 찾는 방법을 제안하였다[6]. 그러나 순방향 움직임 예측 방법은 프레임의 겹쳐지는 부분이나 빈 공간을 만들어내는 단점이 존재한다. 이를 보완하기 위해 다양한 방법들이 제시되고 있다. 순방향 움직임 예측 방법이 현재 프레임을 중심으로 이전 프레임을 이용해 움직임 벡터를 찾아 중간 삽입 프레임을 생성하였다면, 생성하고자 하는 중간 프레임을 기준으로 이전 프레임과 현재 프레임간의 대칭성을 이용한 최적의 움직임 벡터를 찾는 쌍방향 예측 방법이 제안되었다[8-9]. 쌍방향 움직임 예측 방법은 기존의 순방향 움직임 예측 방법에서 나타나는 프레임의 겹쳐지는 부분과 빈 공간을 만들지 않는 장점이 있다. 그러나 움직임 벡터를 추정하는 과정에서 잘못된 움직임 벡터를 추정할 가능성이 있다.

본 논문에서는 정확한 움직임 벡터 생성을 통해 올바른 중간 삽입 프레임을 복원하는 방법을 제안한다. 쌍방향 움직임 예측 방법을 바탕으로 하여 중간 삽입 프레임을 생성할 때 인접한 블록들이 갖는 움직임 벡터와의 상관성을 이용하여 잘못된 움직임 벡터를 판별하고 잘못된 움직임 벡터로 판별된 경우, 이를 보정하기 위하여 왜곡된 블록을 중심으로 이전에 올바르게 생성된 인접한 블록들과 병합시켜 다시 쌍방향 예측을 함으로써 움직임 벡터를 보정하는 방법을 제안한다.

본 논문의 구성은 다음과 같다. 본 논문의 2장에서는 기존에 제안된 움직임 보상을 이용한 프레임 생성 방법을 설명하고 제 3장에서는 쌍방향 움직임 예측 과정에서 올바른 움직임 벡터를 찾는 새로운 알고리듬을 제안한다. 제 4장에서는 제안하는 알고리듬과 기존의 알고리듬을 비교하기 위해 PSNR 성능을 비교 분석한다. 마지막으로 제 5장에서는 결론을 맺는다.

Ⅱ. 움직임 보상 프레임 보간 기법

순방향 움직임 예측 방법을 이용한 프레임 보간 기법은 그림 1에 제시된 것과 같이 현재 프레임을 기준으로 이전 프레임을 이용하여 추정된 움직임 벡터를 v라 할 때 이전 프레임에서 v/2에 위치한 블록과 현재 프레임에서 -v/2에 위치한 블록의 평균값을 얻어 중간 삽입 프레임을 생성한다[6-7]. 즉, 식 (1)과 같이 현재 프레임(f_{n+1})과 이전 프레임(f_{n-1})을 이용하여 중간 삽입 프레임(f_n)의 한 블록을 생성한다.

$$f_n(x,y) = \frac{f_{n-1}(x+v_x/2,y+v_y/2)+f_{n+1}(x-v_x/2,y-v_y/2)}{2} \quad (1)$$

Park 등은 현재 프레임의 현재 블록을 기준으로 이전 프레임에서 최적의 움직임 벡터를 추정하고 예측할 블록의 인접한 블록이 가지고 있는 움직임 벡터와 zero 벡터를 후보 벡터로 정하고 이들 중 SAD (sum of absolute difference) 값이 가장 작은 값을 나타내는 후보 벡터를 최적의 벡터로 정하고 이를 이용하여 중간 삽입 프레임을 생성하는 방법을 제안하였다[6]. 이 과정에서 블록 간에 중복되어 나타나는 부분이 발생할 수 있고 화소값이 채워지지 않는 부분이 생길 수 있다. 따라서 Park 등은 블록과 블록간의 겹치는 부분에 대한 해결책으로 BAD(boundary absolute difference) 값을 정의하여 블록의 경계값을 고려함으로서 겹쳐지는 현상을 방

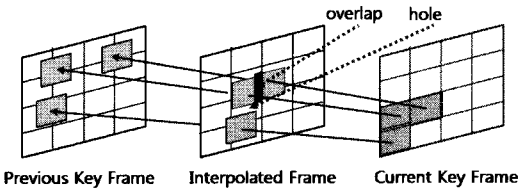


그림 1. 순방향 움직임 예측을 이용한 프레임 보간 기법

지한다. 그러나 움직임 벡터를 선정하기 위해 너무나 많은 연산을 수행해야 하고 정확한 경계값을 정하는 기준이 불명확하며 테스트 영상마다 최적의 중간 삽입 프레임을 얻기 위해 많은 실험을 통해 경계값을 정해야 한다.

이런 순방향 예측의 단점을 보완하고자 Choi 등은 그림 2와 같이 중간 삽입 프레임을 기준으로 이전 프레임과 현재 프레임의 쌍방향 예측을 통한 움직임 예측 방법을 제시하였다[8]. 중간 삽입 프레임의 생성해야 할 블록을 중심으로 이전 프레임과 현재 프레임에서 각각 탐색 범위를 대칭적으로 이동하여 다음 식 (2)를 최소화하는 최적의 벡터를 선택한다.

$$SAD(dx,dy) = \sum_{x\in S_x}\sum_{y\in S_y}|f_{n-1}(x-dx,y-dy)-f_{n+1}(x+dx,y+dy)| \quad (2)$$

이 때 dx와 dy는 각각 행 방향, 열 방향으로 이동한 움직임 크기를 나타낸다. 그리고 S_x, S_y는 탐색 블록 크기를 나타낸다. 위와 같이 쌍방향 움직임 예측을 이용하여 중복되는 블록을 없애고 움직임 벡터를 효율적으로 구할 수 있다. Choi 등은 쌍방향 움직임 예측을 통해 얻어지는 SAD 값과 예측 블록과 인접한 블록과의 경계값을 식 (3)과 같이 SMD(side match distortion)로 정의하여 계산하고 SAD와 SMD를 이용한 식 (4)를 통해 최적의 움직임 벡터를 추정한다.

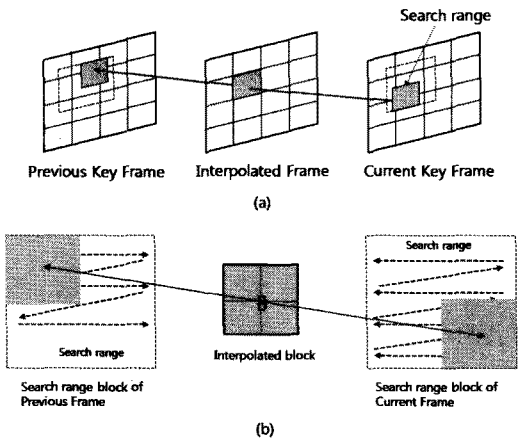


그림 2. 쌍방향 움직임 예측을 이용한 프레임 보간 기법
(a) 공간적인 모형 (b) 평면적 모형

$$SMD(B_{i,j},v) = \frac{1}{N}\sum_{k=0}^{N-1}|\hat{f}_n(g_k,v)-\hat{f}_n(h_k)| \quad (3)$$

$$v_{i,j} = \min_{v}\left\{\mu\times SAD[B_{i,j},v]+(1-\mu)\times SMD[B_{i,j},v]\right\} \quad (4)$$

식 (3)에서 $B_{i,j}$는 움직임 벡터를 찾아야 하는 블록이고 v는 후보 벡터를 나타낸다. 또한 g_k와 h_k는 각각 예측 블록과 인접 블록의 경계 부분에서 k번째 화소값을 의미하고, N은 경계 부분에서의 화소들의 수를 나타낸다. 따라서 식 (4)의 값이 가장 작게 나타나는 벡터값을 선택한다. Choi 등이 제안한 알고리듬에서는 채워지지 않는 부분 및 블록과 블록 간의 겹쳐지는 문제를 해결하고 움직임 벡터 추정에 대한 정확성을 높였지만 식 (3)과 식 (4)를 이용한 반복적인 계산을 필요로 하기 때문에 계산량이 많아지는 단점이 있다. Kang 등은 쌍방향 움직임 예측을 이용하였지만 Choi 등이 제안한 블록간의 경계부분 연속성을 이용하지 않고 중간 삽입 프레임에 존재하는 블록의 크기를 세분화하여 움직임 벡터를 찾아낸다[9]. 따라서 기존의 블록이 갖는 움직임 벡터가 한 개라면 Kang 등이 제안하는 방법에서는 여러 개의 움직임 벡터를 가질 수도 있기 때문에 블록을 세분화하여 움직임 벡터를 다시 분리하는 보정 방법을 사용한다.

본 논문에서는 그림 3과 같이 보간한 영상에서 가장자리 블록들 간의 중복되는 현상을 처리하고 그림 4와 같이 쌍방향 움직임 예측에서 잘못된 움직임 벡터로 인해 블록 왜곡 현상이 나타나는 것을 보정하기 위해 다음 3장에서 정확한 쌍방향 움직임 예측을 이용한 움직임 벡터 생성과 그 보정을 위한 효율적인 알고리듬을 제안한다.

그림 3. 쌍방향 예측에서 블록 왜곡 현상
(a) 보간된 영상 (b) (a)에서 나타난 왜곡현상을 표시한 영상

(a) (b)

그림 4. 잘못된 움직임 벡터에 의한 블록 왜곡 현상
(a) Foreman 영상 (b) Carphone 영상

Ⅲ. 제안하는 알고리듬

본 장에서는 쌍방향 움직임 예측을 이용하여 중간 삽입 프레임을 생성할 때 움직임 추정 과정에서 잘못 선택되어지는 움직임 벡터를 효율적으로 보정하는 기법을 제안한다. 먼저, 쌍방향 움직임 예측을 할 때 그림 3과 같이 가장자리 부분에서는 이전과 현재의 탐색 범위가 작기 때문에 대칭적으로 움직임 벡터를 찾는 것이 어렵다. 따라서 우선적으로 프레임의 가장자리 부분의 화소값을 바깥쪽으로 확장을 시킴으로써 대칭적으로 움직임을 찾을 수 있도록 하였다. 기존에 제안한 방법은 임의의 중간 삽입 프레임 전체를 생성한 후 다시 보정하는 방식이다. 이는 많은 계산량을 요구하는데 본 논문에서는 쌍방향 예측을 이용하여 움직임 벡터를 보정함과 동시에 중간 삽입 프레임을 생성하는 효율적인 방법을 제안한다. 제안하는 방법을 그림 5에 도시하였다.

본 논문에서는 중간 삽입 프레임에 해당하는 블록을 생성할 때 먼저 움직임 추정 과정에서 잘못된 움직임 벡터를 생성하는지 그 여부를 판별한다. 움직임 추정 과정에서 움직임 벡터들을 찾아 중간 삽입 프레임을 생성할 때 올바른 움직임 벡터들로 생성된 인접한 블록들은 행방향 움직임 벡터값 또는 열방향 움직임 벡터값의 차이가 작거나 같다. 따라서 행 방향으로 인접한 블록간의 움직임 벡터값의 차이와 열 방향으로 인접한 블록간의 움직임 벡터값의 차이, 그리고 대각선으로 인접한 블록의 움직임 벡터 간의 차이를 이용하여 그 차이값이 경계

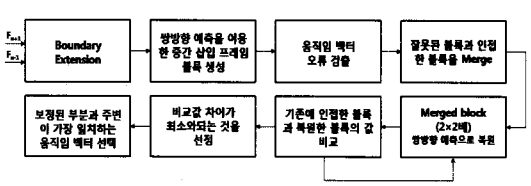


그림 5. 쌍방향 예측을 기반으로 제안하는 알고리듬

범위 이상을 가질 때 예측하는 현재블록이 잘못 예측되었는지를 결정한다. 이 기준이 되는 차이값을 경계값 T로 정의한다.

그림 6(a)에서 현재 블록의 움직임 벡터 v'와 이전에 생성된 중간 삽입 프레임의 블록들 중 인접한 네 개의 블록이 갖는 움직임 벡터들의 상관성을 가지고 있다는 특성을 이용하여 현재 블록의 움직임 벡터 v'의 오류를 판별하기 위한 인접한 블록을 보여준다. 또한 움직임 벡터의 오류를 판별하기 위하여 다음과 같은 식 (5)~(8)의 값을 계산한다.

$$D_{row} = \left|(v_{x_1} - v_{x_2})^2 - (v_{x_4} - v_{x'})^2\right| \quad (5)$$

$$D_{com} = \left|(v_{y_1} - v_{y_4})^2 - (v_{y_2} - v_{y'})^2\right| \quad (6)$$

$$D_{dia} = \left|(v_{x_2} - v_{x_4})^2 - (v_{x_3} - v_{x'})^2\right| + \left|(v_{y_2} - v_{y_1})^2 - (v_{y_3} - v_{y'})^2\right| \quad (7)$$

$$\alpha(D_{row} + D_{com}) + (1-\alpha)D_{dia} \geq T \quad (8)$$

식 (5)~(7)에서 나타낸 v_x는 x축 움직임 벡터값을 나타내고 v_y는 y축 움직임 벡터값을 나타낸다. 예를 들어, v_{x1}은 블록 v_1의 x축 움직임 벡터값이고, v_{y1}은 블록 v_1의 y축 움직임 벡터값이다. D_{row}는 행 방향에 위치한 블록간의 움직임 벡터들에 대한 차이값을 말하며 D_{com}는 열 방향에 위치한 블록간의 움직임 벡터들에 대한 차이값을 말한다. 또한 D_{dia}는 대각선 방향에 위치한 블록간의 움직임 벡터들에 대한 차이값을 말한다. 이를 바탕으로 식 (8)에서는 행 방향과 열 방향에서 얻어진 차이값과 대각선에서 얻어진 움직임 벡터들의 차이값에 가중치를 주어 계산된 값이 경계값 T보다 클 경우 움직임 벡터 v'가 잘못 추정되었다고 판단한다. 그렇지 않

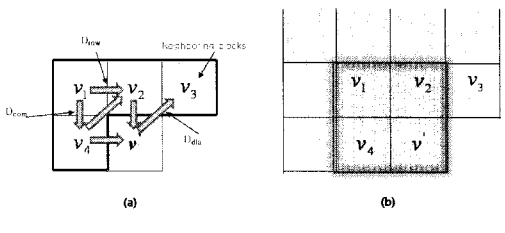


그림 6. 인접한 블록들을 이용한 움직임 벡터 오류 판별 및 블록 병합 (a)움직임 벡터 오류 판별 (b)병합될 블록(2×2배)

다면 움직임 벡터가 올바르다고 판단하고 그대로 움직임 벡터를 사용하여 중간 삽입 블록을 생성한다.

그러나 경계값보다 큰 값을 가질 경우 움직임 벡터 추정을 올바르게 보정하기 위해서 그림 6(b)와 같이 왜곡된 움직임 벡터를 가지는 블록 v'을 중심으로 인접한 블록 v_1, v_2, v_4과 함께 병합을 한다. 병합한 블록에서 1/2화소 보간 탐색영역에서 그림 7과 같이 쌍방향 움직임 예측으로 생성되는 블록과 그림 6(b)에서 블록 v'를 제외한 나머지 블록 v_1, v_2, v_4 부분을 비교하여 가장 일치하는 움직임 벡터를 구한다. 이 때 구한 움직임 벡터를 블록 v'에 움직임 벡터로 사용하여 1/2화소 보간 영역에서 중간 삽입 블록을 생성한다.

Ⅳ. 모의 실험

제안하는 알고리듬의 타당성을 입증하기 위해서 표준 CIF 동영상과 QCIF 동영상에 대한 모의실험 결과를 제시한다. 제안하는 방법에서 잘못된 움직임 벡터 추정 여부를 판단하기 위해 8×8 블록 크기로 실험을 하였고 각 동영상에서 첫 번째 프레임부터 51번째 프레임을 사용하여 첫 번째와 세 번째 사이에서 두 번째 프레임을 생성하고 두 번째 프레임과 네 번째 프레임을 이용하여 세 번째 프레임을 생성하는 식으로 50프레임을 생성하였다. 식 (8)의 경계값 T를 5에서부터 50까지 변화시켰을 때 각 영상마다 성능 차이를 보였다. 이를 토대로 실험에 사용된 모든 동영상이 가지는 PSNR의 평균 대비 계산의 복잡도가 가장 좋은 경계값인 20으로 실험한 결과를 제시한다. 본 실험에서는 각 알고리듬 의 성능을 객관적으로 비교하기 위해 PSNR 결과로 성능을

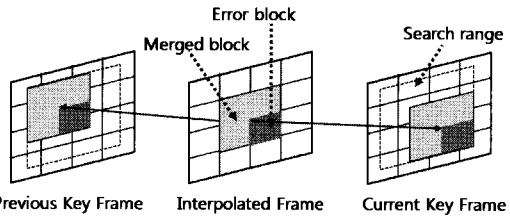


그림 7. 병합한 블록에 대한 쌍방향 움직임 예측

비교 분석한다. Choi 등이 사용한 블록 경계의 연속성을 이용한 방법, Kang 등이 제안한 블록의 조합을 세밀하게 나누어 후보 움직임 벡터를 구하여 1/2 배 크기의 블록 단위로 움직임 벡터를 보정하는 방법과 본 논문에서 제안한 방법으로 중간 프레임을 생성하는 방법을 비교 분석한다. 이 때 쌍방향 움직임 예측에서 경계부분 확장에 대한 동일한 조건하에 블록의 움직임 벡터의 오류를 정정하는 방법을 다르게 하여 성능을 비교한다. 그림 8~11은 Carphone(QCIF), Foreman(QCIF), Stefan(CIF), Mobile(CIF) 테스트 영상을 이용하여 각 알고리즘에 의해 구현된 중간 삽입 프레임들이다. Choi 등이 제안한 경계의 연속성을 이용하여 움직임 벡터의 오차를 보정한 후 중간 프레임을 생성한 영상을 보면,움직임이 적은 Carphone 영상의 경우 비교적 왜곡된 블록의 발생률이 작았지만 왜곡된 블록을 시각적으로 확인할 수 있다. 그림 8(a)와 그림 8(b)를 비교하면 Kang 등이 제안한 보정 방법을 이용했을 때 왜곡된 블록에 대해 보정이 되어 내부 블록에서 시각적으로 왜곡된 블록이 없는 것을 알 수 있다. 반면에 그림 8(c)의 제안하는 방법으로 생성된 중간 프레임을 보면 시각적으로 보이는 왜곡된 블록뿐만 아니라 시각적으로 왜곡이 확연히 보이지 않는 블록에 대해서도 움직임 벡터를 다시 찾아 블

(a) (b) (c)

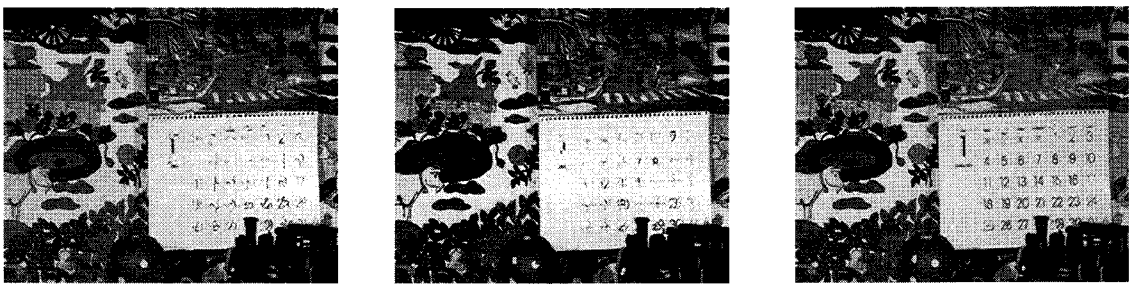
그림 8. Carphone 영상(QCIF)의 쌍방향 예측을 이용한 중간 삽입 프레임
(a) Choi 등이 사용한 예측방법 (b) Kang 등이 사용한 예측방법 (c) 제안하는 예측방법

(a) (b) (c)

그림 9. Forman영상(QCIF)의 쌍방향 예측을 이용한 중간 삽입 프레임
(a) Choi 등이 사용한 예측방법 (b) Kang 등이 사용한 예측방법 (c) 제안하는 예측방법

(a) (b) (c)

그림 10. Stefan 영상(CIF)의 쌍방향 예측을 이용한 중간 삽입 프레임
(a) Choi 등이 사용한 예측방법 (b) Kang 등이 사용한 예측방법 (c) 제안하는 예측방법

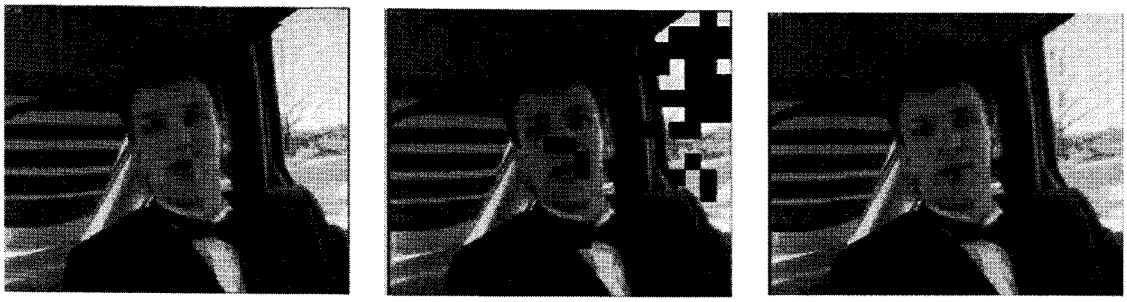


(a) (b) (c)

그림 11. Mobile 영상(CIF)의 쌍방향 예측을 이용한 중간 삽입 프레임
(a) Choi 등이 사용한 예측방법 (b) Kang 등이 사용한 예측방법 (c) 제안하는 예측방법

록을 보정하는 것을 알 수 있다. 그림 11의 Mobile 영상의 경우에도 기존의 방법으로 생성된 중간 삽입 프레임의 달력 안의 숫자들이 뚜렷하게 나타나지 못한 것에 비해 제안하는 방법으로 중간 삽입 프레임을 생성하였을 때는 비교적 선명하게 나타나는 것을 알 수가 있다. 다음 그림 12~14에서 제안하는 방법의 보정된 블록을 보면 시각적으로 왜곡된 블록만을 고치는 것이 아니라 눈으로 확인하기 어려운 부분에서도 움직임 벡터의 보정이 이루어졌음을 입증한다. 그림 12(b), 13(b)와 14(b)의 검게 칠해진 블록은 움직임 벡터가 보정된 블록을 나타낸다. 또한 좀 더 정확한 중간 삽입 프레임을 생성하기 위해 1/2 화소 보간법을 이용하여 블록을 생성하여 그 성능을 분석하였다.

다음 그림 15에서 (c)와 (d)는 제안하는 방법을 동일하게 이용하면서 1/2화소 단위로 탐색 영역을

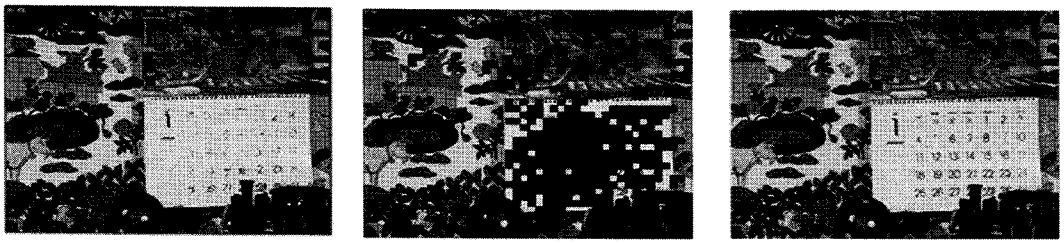


(a) (b) (c)

그림 12. Carphone 영상(QCIF)에서 보정된 블록 확인 (a) 쌍방향 예측을 이용한 중간 삽입 프레임 생성 (b) 제안하는 방법으로 보정된 움직임 벡터를 갖는 블록 표시 (c) 제안하는 방법을 이용한 중간 삽입 프레임 생성

(a) (b) (c)

그림 13. Foreman영상 (QCIF)에서 보정된 블록 확인 (a) 쌍방향 예측을 이용한 중간 삽입 프레임 생성 (b) 제안하는 방법으로 보정된 움직임 벡터를 갖는 블록 표시 (c) 제안하는 방법을 이용한 중간 삽입 프레임 생성



(a) (b) (c)

그림 14. Mobile 영상(CIF)에서 보정된 블록 확인 (a) 쌍방향 예측을 이용한 중간 삽입 프레임 생성 (b) 제안하는 방법으로 보정된 움직임 벡터를 갖는 블록 표시 (c) 제안하는 방법을 이용한 중간 삽입 프레임 생성

확장했을 때와 그렇지 않을 때를 비교하였다. 기존의 방법을 이용하여 생성된 프레임보다 제안하는 방법으로 생성된 프레임이 주관적인 화질이 뛰어났지만 제안하는 방법에서도 1/2 화소 단위 탐색 영역으로 확장했을 시에는 더 좋은 성능을 보인다. 그림 15을 살펴보면 달력위에 숫자들을 정확하게 찾지 못했던 기존의 방법들에 비해 제안하는 방법을 사용했을 때 주관적으로도 높은 성능을 보임을 알 수 있었다.

다음 표 1에서는 각 영상에 대해서 50 프레임씩 기존의 방법과 제안하는 방법으로 움직임 벡터를 보정하였을 때 성능 비교 결과를 제시한다. 첫 번째로 기존의 움직임 예측으로 중간 삽입 프레임을 생성하는 경우의 성능을 분석하고 두 번째로 기존에 제안한 방법으로 프레임을 생성 할 때 블록의 경계 부분 처리와 움직임 벡터 추정의 정확성을 높이기 위해 인접한 블록들이 가지는 움직임 벡터들로 메디안 필터(median filter) 처리하여 중간 삽입 프레

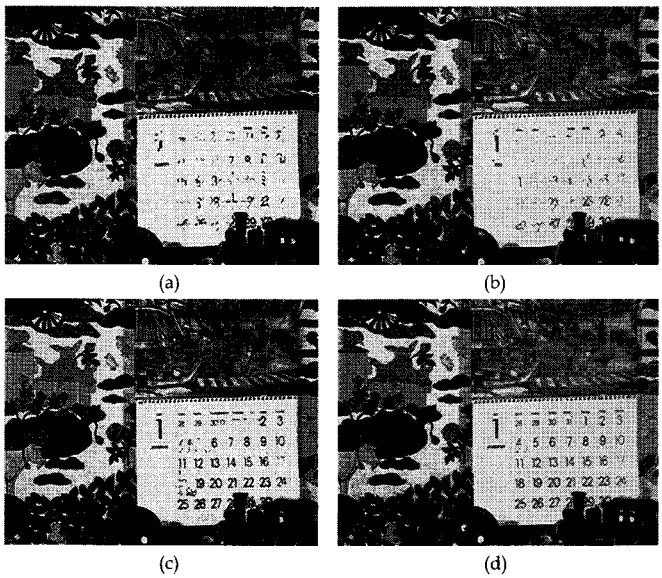


그림 15. Mobile 30번째 프레임 영상 (a)블록 경계 연속성을 이용한 생성 프레임 (b)움직임 벡터를 더 세분화한 생성 프레임 (c) 제안하는 방법에서 1/2 보간 탐색영역이 없이 생성한 프레임 (d) 제안하는 방법에서 1/2 보간 탐색영역을 이용한 생성 프레임

표 1. 중간삽입 프레임 생성 방법에 따른 성능비교 (PSNR)

실험 분류 / 영상의 종류	블록 경계 부분의 연속성을 이용한 쌍방향 움직임 예측방법		블록의 움직임 벡터를 세분화한 쌍방향 움직임 예측방법		본 논문에서 제안한 쌍방향 움직임 예측 방법	
	Choi 등이 제안한 방법 중 블록 경계 연속성을 이용한 예측	경계 처리 및 (Median Filter) 움직임 벡터 보정	Kang 등이 제안한 방법 중 블록 세분화를 이용한 예측	경계 처리 및 (Median Filter) 움직임 벡터 보정	본 논문에서 제안하는 방법	1/2 화소 보간을 이용한 보정
Carphone	27.73	33.50	28.28	33.10	33.66	33.73
Foreman	22.67	34.84	23.63	34.01	36.96	36.99
Miss_am	35.69	45.26	39.09	45.17	45.97	45.97
Mobile	22.74	22.73	20.89	22.29	24.15	26.78
Stefan	18.31	22.65	18.79	23.09	26.37	26.83
Football	21.51	22.68	21.38	22.71	24.37	25.45

임을 생성하는 경우에 대한 성능 분석을 한다. 마지막으로 쌍방향 움직임 예측을 바탕으로 본 논문에서 제안한 보정 방법만을 적용하여 중간 삽입 프레임을 생성하는 방법과 제안하는 보정 방법으로 움직임 벡터를 보정할 때 1/2 화소 단위로 보간하는 방법을 사용하여 중간 삽입 프레임을 생성하는 방법에 대한 성능을 비교하였다. 1/2 화소 단위로 보간 영역을 사용한 결과를 살펴보면 사용하지 않았을 때보다 성능이 향상됨을 알 수 있다. 특히, Mobile 영상의 경우에 가장 눈에 띄게 성능이 좋아진다. 그러나 움직임이 적은 Carphone 영상이나 Foreman 영상은 1/2 화소 단위로 보간 영역을 사용하지 않아도 화질이 많이 좋아졌기 때문에 화질 개선의 한계에 도달하여 성능이 크게 개선되지 않는다. 또한 움직임이 많은 Stefan 영상이나 Football 영상의 경우에서도 1/2 화소 보간의 사용 여부에 따라 PSNR 성능이 좋아짐을 알 수 있다. Mobile 영상의 경우 전체적으로 움직임 변화가 느리며 일정한 움직임 패턴을 가지고 있다. 따라서 다른 영상에 비해 1/2 화소 보간 방법을 사용하였을 때 더 좋은 성능을 얻을 수 있었다. 따라서 표 1에 제시한 결과에서 쌍방향 예측만으로 중간 삽입 프레임을 생성하였을 때보다 경계부분 처리와 움직임 벡터를 보정하였을 때 성능이 우수함을 알 수가 있다. 또한 본 논문에서 제안한 방법으로

생성한 중간 삽입 프레임이 기존의 방법보다 좋은 결과를 보임을 알 수 있다.

V. 결 론

본 논문에서는 쌍방향 움직임 예측을 이용하여 중간 삽입 프레임을 생성하는 과정 중에 잘못된 움직임 벡터의 유무를 판별하고 이를 보정하기 위한 효율적인 방법을 제안하였다. 쌍방향 움직임 예측은 순방향 움직임 예측에 비하여 중간 삽입 프레임을 생성할 때 블록 왜곡 현상이 발생하지 않도록 효율적으로 움직임 벡터를 추정하는 기법이다. 그러나 잘못 보간된 블록의 움직임 벡터를 보정해야 하는데 이를 위해 기존의 방법에서는 중간 프레임을 생성한 후 다시 보정하는 단점이 있다. 이를 개선하기 위해서 쌍방향 움직임 예측을 할 때 이미 생성된 인접한 블록들의 움직임 벡터간의 관계를 이용하여 순차적으로 블록을 생성함으로써 복잡도를 줄이고 기존의 방법에 비해 좀 더 정확한 예측을 하는 기법을 제안하였다. 제안하는 알고리즘의 성능을 테스트 영상을 이용하여 측정한 결과 제안하는 알고리듬이 기존 알고리즘에 비하여 개선된 성능을 보임을 확인하였다. 특히 제안하는 방법에서 1/2 보간 화소 단위 움직임 예측을 적용했을 때 좀 더 좋은 성능을 보임을 확인하였다.

참 고 문 헌

[1] A. Vetro, C. Christopoulos, and H. Sun, "Video transcoding architectures and techniques: an overview", *IEEE Signal Processing Magazine,* pp. 18-29, Mar, 2003.

[2] B. Girod, A. M. Aaron, S. Rane, D. Rebollo-Monedero, "Distributed video coding", *IEEE Proceedings,* vol. 93, pp.71-83, Jan. 2005.

[3] P. Puri, A. Majumdar, P. Ishwar, K. Ramchandran, "Distributed video coding in wireless sensor networks", *IEEE Signal Processing Magazine,* vol. 23, pp.94-106, July, 2006.

[4] L. Wei, Y. Zhao, A. Wang, "Improved Side-Information in Distributed Video Coding", *Proc. ICICIC,* vol.2, pp.209-212, Aug. 2006.

[5] W. J. Chien, L. J. Karam, G. P. Abousleman, "Distributed Video Coding with Lossy Side Information", *Proc. ICASSP*, vol. 2, pp. II69-II72, May 2006.

[6] K. Hilman, H. W. Park and Y. Kim, "Using motion-compensated frame rate conversion for the correction of 3:2 pull down artifacts in video sequences", *IEEE Trans. Circuits Syst. Video Technol.,* vol. 10, no. 6, pp. 867-877, Sep. 2000.

[7] S. Fujiwara, A. Taguchi, "Motion Compensated Frame Rate Up-Conversion Based on Block Matching Algorithm with Multi-size Blocks", *IEEE International Symposium on ISPCS.,* pp.353-356, Dec. 2005.

[8] B. D. Choi, J. W. Han, C. S. Kim, S. J. Ko,"Motion Compensated Frame Interp -olation Using Bilateral Motion Estimation and Adaptive Overlapped Block Motion Compensation", *IEEE Trans. Circuits and Systems for Video Technology*, vol 17, pp.407-416, April 2007.

[9] S. J. Kang, K. R. Cho, Y. H. Kim, "Motion Compensated Frame Rate Up Conversion Using Extended Bilateral Motion Estimation", *IEEE Trans., Consum.,* vol. 53 , no. 4, pp. 1759-1767, Nov, 2007.

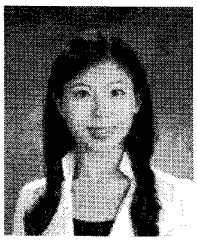
[10] R. Castagno, P. Haavisto and G.Ramponi, "A method for motion adaptive frame rate up conversion", *IEEE Trans. Circuits. System Video Techol.,* vol. 6, no. 5, pp. 436-446. Dec. 1996.

[11] J. Zhai, K. Yu, J. Li and S.Li, "A low complexity motion compensated frame interpolation method", *Proc. ISCAS,* vol. 5, pp. 4927-4930, May. 2005.

[12] D. N. Kwon, "Half-Pixel Accuracy fast Search in video coding", *IEEE International Symposium on Signal Processing and Applications Proc,* vol. 1, pp. 73-76, July 2003.

[13] D. K. Park, H. M. Cho, S. B. Cho, J. H. Lee, "A Fast Motion Estimation Algorithm for SAD Optimization in Sub-pixel", *Proc. ISIC*, pp. 528-531, Sept. 2007.

박 지 윤 (Ji-yoon Park) 준회원



2008년 2월 가톨릭대학교 정보통신전자공학부 학사

2008년 2월~현재 가톨릭 대학교 정보통신 전자공학과 석사과정

<관심분야> 신호처리, 영상통신, 영상처리

이 창 우 (Chang-woo Lee) 종신회원

현재 가톨릭대학교 정보통신전자공학부 교수

<관심분야> 영상통신, 영상처리