유도 움직임 벡터를 이용한 움직임 보상 프레임율 향상 기법

*박범준 *유송현 **정제창
*한양대학교 전자컴퓨터통신공학과 **한양대학교 융합전자공학부
kkbbbj1@naver.com, 3069song@naver.com, jjeong@hanyang.ac.kr

Motion-Compensated Frame Rate Up-Conversion Using Guidance Motion Vector

*Bumjun Park *Songhyun Yu **Jechang Jeong
*Dept. Electronics and Computer Eng. Hanyang University
**Dept. Electronic Eng. Hanyang University

요 약

본 논문에서는 프레임율 향상 기법 (Frame Rate Up-Conversion, FRUC)에 사용되는 새로운 움직임 예측 (motion estimation)알고리즘을 제시한다. 제안된 알고리즘은 단 방향 움직임 예측(unilateral motion estimation)에 의해 순방향 및 역방향의 움직임 벡터(motion vector)를 독립적으로 추정한다. 움직임 벡터를 찾은 후, weighted motion vector smoothing(WMVS)가 적용된다. 다음으로, 보간 프레임 (interpolated frame)의 관점에서 현재 블록의 인접 블록들의 모션 벡터들을 후보들로 사용하여 현재 블록과 가장 잘 일치하는 움직임 벡터를 찾는다. 그 후, 선택된 움직임 벡터를 현재 블록의 유도 움직임 벡터 (guidance motion vector)로 정한다. 그런 다음 motion vector shifting error 를 없애기 위해 motion vector refinement (MVR)가 진행된다. 마지막 단계에서는 각 움직임 벡터의 신뢰도를 계산하여 순방향 및 역방향 움직임 벡터 중 최종 움직임 벡터를 선택한다.

1. 서론

프레임율 향상 기법 (Frame Rate Up-Conversion, FRUC)는 원본 동영상의 두 개의 인접한 프레임을 이용하여 새로운 보간 프레임 (interpolated frame)을 생성하는 기법이다. 즉, FRUC 기법을 사용하면 두 개의 원본 프레임을 사용하여 두 프레임의 시간 간격의 중간에 존재하는 새로운 가상 프레임을 얻을 수 있다. 이 기법을 통해 낮은 프레임율을 가진 동영상을 더 높은 프레임율을 가진 동영상으로 만들 수 있다.

연구 초기 단계에서 FRUC 는 원본 프레임을 복사하여 구현되었다. 그러나 이는 상용화 하기에 부족한 품질을 보여주어 움직임 보상 프레임율 향상 기법 (Motion-Compensated FRUC, MC-FRUC)이 제안되었다. MC-FRUC 는 주로 움직임 예측 (Motion Estimation, ME)과 움직임 보상 보간 (Motion-Compensated Interpolation, MCI)의 두 부분으로 구성된다 [1]. 많은 ME 알고리즘이 존재하지만, 그 중 소프트웨어와 하드웨어 모두에서 구현이 용이한 Block Matching Algorithm (BMA)가 주로 사용된다 [2]. BMA 는 모든 프레임을 동일한 크기 블록으로 나눈 후, 블록 간의 유사도를 비교하여 움직임 벡터 (motion vector, MV)를 구하는 알고리즘이다. MC-FRUC 는 BMA 를 이용하여 단 방향 ME (unilateral ME, UME)와 양 방향 ME(bilateral ME, BME)의 두 가지 방법으로 MV 를 찾는다. UME 는 한 원본 프레임의 현재 블록과 가장 비슷한 블록을 다른 원본 프레임에서 찾음으로써 MV 를 찾는다. 이때, 이전 프레임의 현재 블록과 가장 비슷한 블록을 현재 프레임에서 찾는 과정을 순방향 ME (forward ME)라고 한다. 또한, 현재 프레임의 현재 블록과 가장 비슷한 블록을 이전 프레임에서 찾는 과정을 역방향 ME (backward ME)라고 한다. UME 와는 다르게 BME 는 보간 프레임의 시점에서 2 개의 원본 프레임으로 MV 를 찾는다. 일반적으로 UME 는 BME 보다 정확한 MV 를 찾는데 이는 주기적 또는 평활한 배경이 있는 경우 BME 가 부정확한 MV 를 찾을 수 있기 때문이다. 그러나 UME 는 BME 보다 계산상으로 복잡하며 MV 정보가 없거나 MV 정보가 2 개 이상인 영역이 생길 수 있다.

ME 과정을 통해 MV 를 얻은 후, 이를 이용하여 MCI 가 진행된다. 이 과정에서 BMA 로 인해 블록화 현상 (blocking artifact)가 발생하는데, 이 문제를 해결하기 위해 Overlapped Block Motion Compensation (OBMC)과정이 적용된다 [3]. OBMC 는 쌍 선형 창 (bilinear window)을 사용하여 blocking artifact 를 제거한다.

현재까지 많은 MC-FRUC 알고리즘이 제안되었다 [4,5,6]. Kang 은 Extended Bilateral Motion Estimation (EBME)와 recursive MV smoothing 을 이용한 MC-FRUC 를 제안하였다 [6]. 하지만 BME 의 약점을 피할 수 없었으며 계산적 복잡성의 증가에도 불구하고 recursive MV smoothing 은 더 좋은 결과를 보여주지 못했다. BME 의 결함을 피하기 위해 Yoo 는 Direction-Select motion estimation (DSME)를 제안하였다 [7]. DSME 는 보다 나은 MV 를 찾기 위해 순방향 및 역방향 MV 를 찾고 UME 를 이용하여 찾은 MV 를 BME 로 치환하고 이를 통해 UME 와 BME 의 장점을 모두 활용하려고 노력했다. 그러나 MV 치환 오차 (motion vector shifting error)를

줄이기 위한 노력에도 불구하고 단 방향 MV (UMV)를 양방향 MV (BMV)로 치환하는 과정에서 생겨나는 오류를 줄이지 못했다.

본 논문에서는 MV 치환 오차를 줄이기 위한 유도 MV (guidance motion vector, GMV)를 제안한다. MV 를 이동시킬 때 기존 알고리즘 [7]은 현재 블록과 동일한 위치에 있는 하나의 후보 MV 만을 사용했다. 이는 UMV 를 BMV 로 치환할 때에 심각한 오차를 야기했다. 이 문제를 해결하기 위해 본 논문에서는 현재 블록의 BMV 의 후보 MV 로 현재 블록의 주변 블록 MV 를 고려하였고 후보 MV 중에서 가장 신뢰할 수 있는 MV 를 선택함으로써 MV 치환 오차를 줄일 수 있었다.

본 논문의 구성은 다음과 같다. 2 절에서는 제안된 알고리즘의 세부사항을 기술한다. 3 절에서는 기존의 알고리즘과 제안된 알고리즘을 비교한 실험 결과를 보여주고 4 장에서 결론을 맺는다.

2. 제안하는 알고리즘

그림 1 은 제안하는 알고리즘의 전체 흐름도이다. 본 논문에서는 Sum of Absolute Difference (SAD)을 계산함으로써 FME 및 BME 를 수행한다. 다음으로, Weighted MV smoothing (WMVS)을 수행한다. 이 단계에서, outlier MV 는 이웃 MV 의 가중 합으로 보정된다. 그 후, UMV 를 BMV 로 치환하는 MV shifting 을 수행한다. MV 를 치환 시킬 때에는 현재 블록과 겹치는 영역을 비교함으로써 9 개의 UMV 후보 중 GMV 를 선택한다. MV 를 치환 시킨 후, 잔여 오차를 줄이기 위해 작은 탐색 범위 (small search range)에서의 보정을 거친다. 마지막으로, Sum of Bilateral Absolute Difference (SBAD)를 구하여 현재 블록의 최종 MV 를 선택한다.

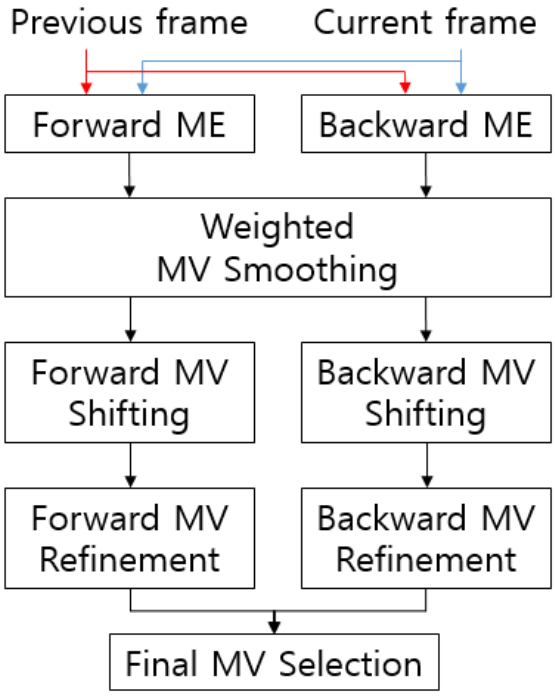


그림 1. 제안하는 알고리즘의 흐름도

Ⅰ. Forward and Backward Motion Estimation

본 논문에서는 순방향 및 역방향 ME 를 모두 고려한다. 이 때, 각 방향에서 탐색 범위 (search range) 내의 후보 MV 들의 SAD 를 계산하여 최소 SAD 를 가지는 MV 를 순방향 및 역방향 MV 로 정한다. 순방향 및 역방향의 SAD 와 MV 는 다음과 같이 정의된다.

$$SAD_f(dx,dy)=\sum_{x,y\in B}|f_{n-1}(x,y)-f_n(x-dx,y-dy)|$$
$$SAD_b(dx,dy)=\sum_{x,y\in B}|f_{n-1}(x-dx,y-dy)-f_n(x,y)| \quad (1)$$
$$v_f=\arg\min_{dx,dy\in R}\{SAD_f(dx,dy)\}$$
$$v_b=\arg\min_{dx,dy\in R}\{SAD_b(dx,dy)\}$$

수식 (1)에서 SAD_f 와 SAD_b 는 각각 순방향 및 역방향 SAD 를 나타낸다. (dx,dy) 는 후보 MV 위치, B 는 현재 블록을 나타내고 f_{n-1} 과 f_n 은 각각 이전 프레임과 현재 프레임을 나타낸다. v_f 와 v_b 는 순방향 및 역방향 MV 를 나타내고, R 은 탐색 범위를 나타낸다.

Ⅱ. Weighted Motion Vector Smoothing

ME 과정을 거친 후 WMVS 를 통해 outlier MV 를 보정한다. 그림 2 는 WMVS 과정에서 쓰이는 MV 들의 표기법을 보여준다. MV smoothing 은 outlier MV 검출 과정과 outlier MV 보정 과정으로 이루어져 있다. 이 때, outlier MV 는 수식 (2)를 계산하여 검출한다.

$$v_{avg}=\frac{1}{9}\sum_{i=0}^{8}v_i$$
$$D_i=abs(v_{avg}-v_i) \quad (2)$$
$$D_n=\frac{1}{8}\sum_{i=1}^{8}D_i$$

수식 (2)에서 v_{avg} 는 9 개의 MV 들의 평균을 나타내며, D_i 는 v_{avg} 와 v_i 의 차이를 나타낸다. D_n 은 주변 MV 들의 평균 MV 차이를 나타내고, D_0 가 D_n 보다 크면 현재 블록의 MV v_0 를 outlier MV 로 검출한다.

Outlier MV 를 검출 한 후, 주변 MV 를 이용하여 outlier MV 를 보정한다. 보정 할 때에는 주변 MV 의 신뢰도에 따라 다른 가중치를 적용하여 outlier MV 를 보정한다. 가중치 및 보정 방법은 다음과 같이 정의된다.

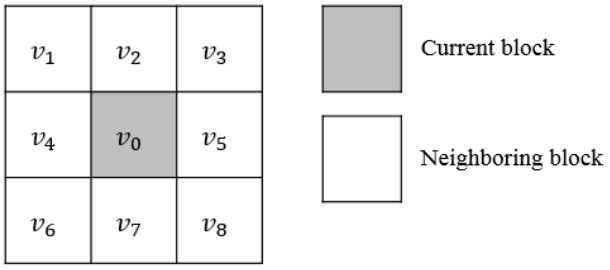


그림 2. WMVS 과정에서 MV 표기법

$$w_i = \begin{cases} \frac{1}{\sqrt{2}}, & i = 1,3,6,8 \\ 1, & i = 2,4,5,7 \end{cases} \qquad (3)$$

$$v_{0s} = \frac{\sum_{i=inliers} v_i \times w_i}{\sum_{i=inliers} w_i}$$

수식 (3)에서 w_i 는 주변 MV 에 적용하는 가중치를 나타내고, v_{0s}는 보정된 v_0를 나타낸다.

Ⅲ. Motion Vector Shifting

이 단계에서는 UMV 를 BMV 로 치환한다. 기존의 알고리즘 [7]에서는 참조 프레임에서 현재 블록과 동일한 위치에 위치한 하나의 MV 만을 고려했다. 그러나 본 논문에서는 현재 블록의 MV 뿐만 아니라 주변 블록의 MV 도 고려한다. 그림 4 는 Motion Vector Shifting 의 예를 보여준다. 먼저, 9 개의 MV 가 참조 프레임의 현재 블록과 중첩되는 영역을 비교한다. 이 단계에서는 세 가지 종류의 사례가 발생한다. 첫째, 현재 블록과 겹치는 MV 가 없다면 현재 블록과 같은 위치에 있는 참조 프레임의 블록을 GMV 로 선택한다. 둘째, 가장 많이 겹치는 MV 가 하나만 있는 경우 그 MV 를 현재 블록의 GMV 로 선택한다. 마지막으로, 가장 많이 겹치는 MV 가 2 개 이상 있는 경우, 수식 (4)를 계산하여 각 MV 의 SBAD 를 현재 블록 위치에서 구하여 비교한다.

$$SBAD(v_{hx}, v_{hy}) = \sum_{x,y \in B} |f_{n-1}(x \mp v_{hx}, y \mp v_{hy}) - f_n(x \pm v_{hx}, y \pm v_{hy})| \qquad (4)$$

$$v_g = \arg\min_{v_i \in M} \{SBAD(v_{hx}, v_{hy})\}$$

수식 (4)에서 v_{hx} 와 v_{hy} 는 각각 v_i 의 x 방향 계수와 y 방향 계수의 절반을 나타낸다. 순방향 MV 의 경우 윗줄의 연산자를 사용하고 역방향 MV 의 경우 아랫줄의 연산자를 사용한다. SBAD 가 최소인 MV 를 GMV v_g로 선택한다.

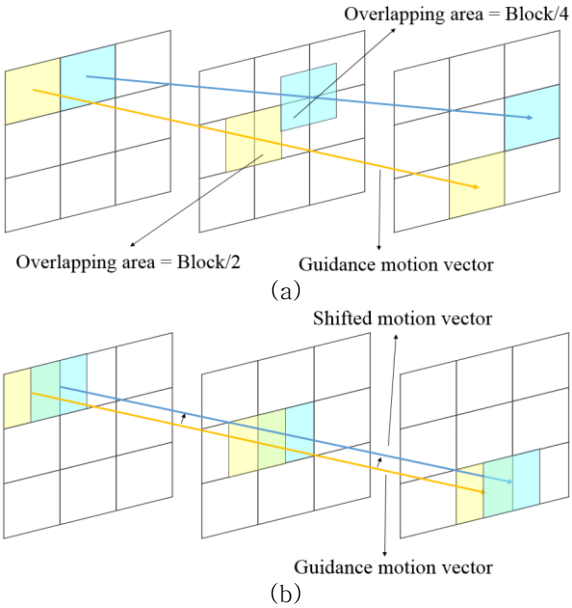


그림 4. Motion Vector Shifting 의 예시
(a) 중첩영역비교, (b) GMV 를 현재 블록에 적용

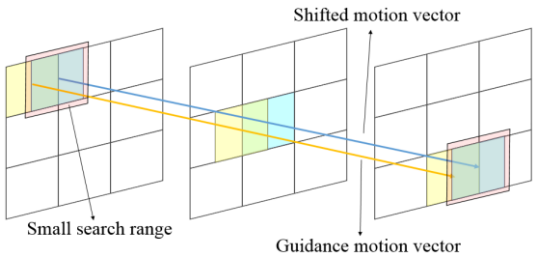


그림 5. Motion Vector Refinement 의 예시

Ⅳ. Motion Vector Refinement

본 논문에서는 UMV 를 BMV 로 치환 시키는 과정에서 오차를 줄이기 위해 GMV 를 사용하였다. 하지만 잔여 오차를 고려하여 본 논문에서는 MVR 과정을 거친다. MVR 과정에서는 작은 탐색 범위에서 GMV 를 보정한다. 그림 5 는 MVR 과정의 예시를 보여준다. 이 과정에서는 작은 탐색 범위에서 최소 SBAD 를 갖는 위치를 찾음으로써 GMV 를 보정하여 잔여 MV 치환 오차를 제거합니다. 보정 방법은 다음과 같이 정의된다.

$$v_{gr} = \arg\min_{s_x, s_y \in S} \{SBAD(s_x, s_y)\} \qquad (5)$$

수식 (5)에서 v_{gr} 은 보정된 GMV 를 나타낸다. s_x 와 s_y 는 후보 보정 위치를 나타내고, S 는 작은 탐색 범위를 나타낸다.

Ⅴ. Final Motion Vector Selection

모든 과정을 순방향 및 역방향 MV 에 적용한 후, 이 과정에서는 두 후보 MV 중 최종 MV 를 선택한다. 이 때, 각 MV 의 SBAD 를 신뢰도로 사용하고, 신뢰도와 최종 MV 는 다음과 같이 정의된다.

$$r_f = SBAD(v_{fx}, v_{fy})$$

$$r_b = SBAD(v_{bx}, v_{by}) \qquad (6)$$

$$v_{final} = \min(r_f, r_b)$$

수식 (6)에서 r_f와 r_b는 각각 순방향 및 역방향 MV 의 신뢰도를 나타낸다. v_{fx}와 v_{fy}는 순방향 MV 의 x방향 및 y 방향 계수를 나타내고, v_{bx} 와 v_{by} 는 역방향 MV 의 x방향 및 y방향 계수를 나타낸다. v_{final} 은 최종 MV 를 나타낸다.

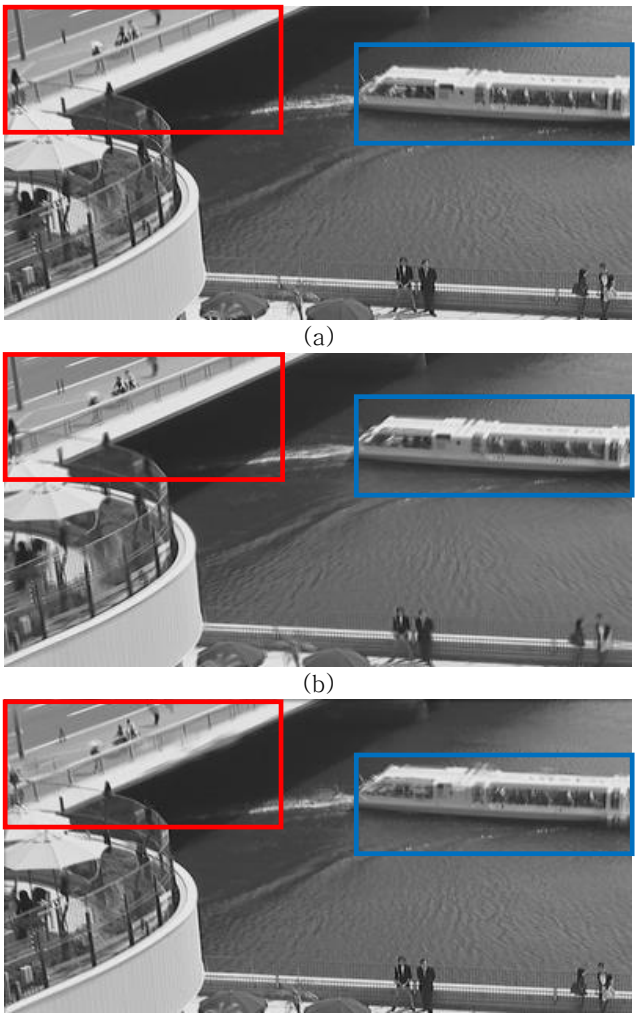
3. 실험 결과

본 논문에서는 짝수 번째 프레임을 원본 동영상의 두 인접 홀수 프레임으로 보간 하였다. 실험 영상은 HEVC 실험 영상을 사용하였다. 실험을 수행 할 때 블록 크기는 16 픽셀, 작은 탐색 범위는 1 픽셀로 설정하였다. MC-FRUC 알고리즘의 객관적 화질 평가에는 Peak Signal to noise ratio (PSNR)가 사용된다. 보간 된 짝수 프레임과 원본 짝수 프레임을 사용하여 PSNR 을 계산하였고 PSNR 의 단위는 dB 이다.

표 1. 기존 알고리즘과 제안하는 알고리즘의 PSNR(dB) 비교

	Sequence	DSME	EBME	Proposed
Class C	*BasketballDrill*	27.34	24.92	27.49
	BQMall	31.87	23.58	32.15
	PartyScene	27.15	26.66	29.85
	RaceHorses	24.66	21.1	24.92
Class D	*BasketballPass*	28.62	23.81	27.85
	BlowingBubbles	29.17	27.14	32.15
	BQSquare	28.14	25.11	31.65
	RaceHorses	27.61	21.64	27.34
Class E	*FourPeople*	37.76	33.45	37.43
	Johnny	38.45	33.35	38.55
	KristenAndSara	38.15	33.33	37.75
Average		**30.81**	**26.74**	**31.56**

표 1 은 기존 알고리즘과 제안하는 알고리즘의 PSNR 비교 결과를 보여준다. 일부 영상에서 제안된 알고리즘은 약간의 PSNR 손실을 보였다. 그러나 대부분의 영상에서는 향상된 성능을 보였으며 제안된 알고리즘은 평균 4.82 dB 의 PSNR 향상을 보였다. 그림 6 은 기존의 알고리즘 및 제안된 알고리즘의 결과 영상을 보여준다. 적색과 청색 상자에서 기존 알고리즘은 잘못된 MV 를 찾아 화질 저하를 보여준다. 그러나, 제안된 알고리즘은 정확한 MV 를 찾아 향상된 성능을 보여준다.



(a) (b) (c)

그림 6. 결과 영상
(a) 제안하는 알고리즘, (b) DSME, (c) EBME.

4. 결론

본 논문에서는 기존 알고리즘보다 우수한 성능을 보이는 MC-FRUC 알고리즘을 제안 하였다. 9 개의 후보 MV 중에서 가장 신뢰할 수 있는 MV 를 GMV 로 선택하는 새로운 MV 치환 방법을 통해 MV 치환 오류를 제거 할 수 있었고, 결과적으로 제안된 알고리즘은 평균 4.82 dB 의 PSNR 향상을 보였다.

감사의 글

본 연구는 미래창조과학부 및 정보통신기술진흥센터의 정보통신·방송 연구개발사업의 일환으로 수행하였음.[R0601-16-1063, ICT 장비용 SW 플랫폼 구축]

참고문헌

[1] C. Cafforio, F. Rocca, and S. Turbaro, “Motion compensated image interpolation,” *IEEE Trans. Communications*, vol. 38, no. 2, pp. 215 – 222, Feb. 1990.

[2] F. Dufaux and F. Moscheni, “Motion estimation techniques for digital TV : a review and a new contribution,” *Proc. IEEE*, vol. 83, pp. 858 – 876, June 1995.

[3] M.T. orchard and C.J. sullivan, “Overlapped block motion compensation : an estimation-theoretic approach,” *IEEE Trans. Image Processing*, vol. 3, no. 9, pp. 693 – 699, Sept. 1994.

[4] S.H. Lee, Y.C. Shin, S. Yang, H.H. Moon, and R.H. Park,"Adaptive motion-compensated interpolation rate-up-conversion," *IEEE Trans. Consumer Electronics*, vol. 48, no. 3, pp. 444 – 450, Aug. 2002.

[5] S.H. Lee, O. Kwon, and R.H. Park, "Weighted- adaptive motion compensated frame rate up-conversion," *IEEE Trans. Consumer Electronics*, vol. 49, no. 3, pp. 485 – 492, Aug. 2003.

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

[7] D.-G. Yoo, S.-J. Kang, and Y.H. Kim, “Direction-Select Motion Estimation for Motion-Compensated Frame Rate Up-Conversion,” *Journal of Display Technology*, vol. 9, no. 10, pp. 840 – 850, Oct. 2013.