프레임률 상향 변환에서 1차 보간영상의 공간적 특징을 활용한 움직임 벡터 보정 방법

김상철◦ 오두희 낭종호
서강대학교 컴퓨터 공학과
smaslayer1@nate.com, doohllo7@naver.com, jhnang@sogang.ac.kr

Motion Vector Correction Method Using Spatial Relation in Interpolated Frame for Frame Rate Up Conversion

SangChul Kim◦ Doo Hee Oh
Department of Computer Science and Engineering, Sogang University

요 약

Frame Rate Up Conversion(FRUC)는 프레임율을 높여 화면을 더 부드럽게 보이는 기술로 객체의 움직임을 판단하여 해당 움직임을 이용해 보간 하여 프레임을 생성한다. 이 때 객체의 움직임이 잘못 추정되면 보간된 프레임의 품질이 떨어지기 때문에 잘못 추정된 움직임을 보정해주는 작업이 필요하다. 기존의 방법들은 추정된 MV(Motion Vector)만을 이용하여 움직임을 보정하였기 때문에 잘못 추정된 MV들에 의해서 보정 품질이 저하된다. 본 논문에서는 1차 보간된 영상의 국소 공간적 연관성을 MV를 보정하는 방법을 제안하였다. 보간 영상의 블록들의 연결이 부드러운지 판단하여 부드럽지 않을 경우 MV를 보정하는 방법을 적용하였고, 이 때 신뢰할 만한 MV들만 보정작업에 관여하게 하여 보정 품질을 높이도록 유도하였다. 실험을 통하여 제안한 방법의 보간 영상 품질이 기존의 방법보다 높음을 알 수 있었다.

1. 서 론

FRUC (Frame Rate Up Conversion)은 영상의 프레임들 사이에 가상의 프레임을 삽입하여 자연스러운 재생을 통해 품질을 높이는 기술이다. FRUC는 ME(Motion Estimation) 단계와 MCI (Motion Compensated Interpolation) 단계로 구성한다. 객체가 얼마나 움직였는가를 나타내는 MV(Motion Vector)를 구하는 것이 ME단계이고 MV를 이용해 Interpolation Frame을 만드는 것이 MCI 단계이다. ME를 하는 방법으로 Uni-direction FRUC와 Bi-lateral FRUC가 있다. Uni-direction FRUC는 MV를 올바르게 찾지 못했을 경우 hole 현상, overlap 현상이 발생하며[1], 이러한 hole현상과 overlap 현상이 발생하지 않도록 개선된 방식이 Bi-lateral FRUC이다. 생성될 프레임을 기준으로 잡고 이전 프레임과 현재 프레임 양 방향에서 MV를 가져오는 방식이다[2]. ME에서 올바른 MV를 찾지 못하는 이유는 Motion Estimation Formula에 있다. 영상에 존재하는 노이즈, 또는 영상 내 객체가 회전하거나 빠른 속도로 이동하는 부분엔 취약하기 때문에 이 경우 올바른 MV를 찾는 데 한계가 있다. 따라서 이것을 보정해주기 위한 작업이 필요하며 이 문제를 보정해준다면 FRUC의 결과물을 향상시킬 수 있을 것이다. 이러한 노력으로 [3][4][5]은 FRUC를 위해 MV를 보정하는 방법에 대해 제안하였지만, 모두 주변부보다 특이하게 튀는 MV를 고려하여 보정하는 방식으로, 실제 결과물과는 상관없이 추정된 MV들로 보정을 하기 때문에 보정 결과를 신뢰할 수 없다. 따라서, 본 논문에서는 그림 1과 같이 잘못된 MV를 찾아 보정하는 방법을 제안한다. 1차 결과물에서 블록간의 경계가 자연스럽지 않다면, 보정해주는 작업을 한다. 제안한 방법은 실험을 통해 기존의 Bi-lateral FRUC의 MV 보정 방식보다 좋은 결과를 가져오는 것을 볼 수 있다. 2장에선 MV를 보정하는 관련 연구를 설명하고 3장에선 제안한 방법에 대한 설명을 한다. 4장에선 실험 및 결과를 논의하고 5장에서 향후 연구에 대해 설명한다.

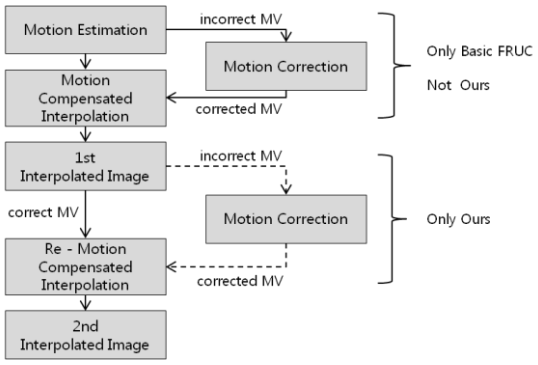

그림 1. 제안하는 모션 벡터 보정 방법의 프레임워크

2. 관련 연구

2.1. Bilateral-FRUC

생성될 프레임을 기준으로 객체의 움직임은 시간적 대칭성이 존재한다는 가정하에 양방향으로 대칭적인 MV를 추정하여 보간하는 방법이다[2]. 보간할 때에 블록간 매끄럽지 않은 연결로 인해 후보정 과정에서 영상의 이미지가 흐릿해지는 모션 블러 현상이 나타나는 문제점이 있다.

2.2 Interpolation 이전의 MV 보정 방법들

[5]는 모든 블록들이 주변 블록들의 MV와의 median filter를 통과하여 튀는 MV를 중간값으로 보정해주는 방법으로 잘

찾아진 MV들이 튀는 MV들에 의하여 오히려 오보정 되는 경우가 많다. [4]은 주변 블록들과의 MV들의 분산을 이용하여 신뢰할만한 MV를 추려내어 이들로 보정하는 방법으로 픽셀값이 아닌 MV들간의 분산을 이용하는 것이기 때문에 애초에 잘못 찾아진 MV의 경우에 오보정 되는 문제점이 존재한다.

그림 2. 보간된 영상에서 잘못 찾아진 모션벡터로 인해 생성된 화면의 예시

3. 블록 가장 자리를 이용한 모션 벡터 보정 방법

```
Algorithm: CorrectMVUsingBoundaryCheck()  {
        TFTable[blockSize][blockSize];
        for (i = 0, j = 0; j < blockSize; i++){
          if(BlockBoundaryDiff(i,j) < thr_boundary)
              TFBlock[i][j] = true;
          if(i > blockSize){
            i = 0; j++;
          }
        }//end for
        while(∀TFBlock != true){
        for (i = 0, j = 0; j < blockSize; i++){
          Block[i][j].motionVector
      = getAverageMVSimilarNeighbourBlk(i, j);
          if(Block[i][j].motionVector is exist)
            TFBlock[i][j] = true;
          if(i > blockSize){
            i = 0; j++;
          } //end for
        }//end while
}//end algorithm
```

그림 3. 블록 가장자리를 이용한 MV 보정 방법의 알고리즘

잘못 찾아진 MV를 이용하여 프레임을 보간하면 그림2와 같이 매끄럽지 않게 연결된 블록들이 존재한다. 따라서, 보간된 프레임에서 블록들간의 가장자리들이 매끄럽게 연결된 지역은 잘 찾아진 MV일 것이고, 그렇지 않은 부분들은 잘 못 찾아진 MV일 것이다.

3.1. T&F Block table 생성

그림 3은 블록의 가장자리의 변화율을 이용하여 잘 못 검출된 MV들을 분류하고, 분류된 블록들에 MV를 재 보정해주는 알고리즘의 슈더코드를 나타낸다. 블록 간의 경계가 부자연스러운 블록 간의 색상 차이가 크다는 것이므로 MV를 잘못 찾았다고 할 수 있다. 따라서 식(1)에 따라 인접한 두 블록 간의 경계의 차이를 구한다. 식(1)은 $row = i, col = j$ 블록의 가장자리의 변화율을 계산하는 함수다. n은 해당 블록의 주변부 블록을 나타내는 인덱스이며, top, bottom, left, right 4개의 블록과의 변화가 매끄러운지를 판단한다. $Pel_i(Block_{Bound_{i,j}})$은 i,j번째 블록과 인접해 있는 블록과 붙어 있는 가장자리의 픽셀을 나타낸다. b는 블록의 사이즈를 나타낸다. 이렇게 4개의 주변부 블록과의 픽셀값의 변화량의 총량이 $thr_{boundary}$보다 작을 경우에 해당 블록은 매끄럽게 이어진 것으로 판단하여 테이블에 true를 매핑한다.

$$BlockBoundaryDiff(i,j) = \sum_{n=1}^{4}\sum_{i=0}^{b}\left(Pel_i(Block_{Bound_{i,j}}) - Pel_i(Block_{Bound_{k,l}})\right) \quad (1)$$

$$, n = \{\text{top, bottom, left, right}\}$$

$$\text{if } n = \text{top} \rightarrow k = i, l = j-1, \text{if } n = \text{bottom} \rightarrow k = i, l = j+1$$

$$\text{if } n = \text{left} \rightarrow k = i-1, l = j, \text{if } n = \text{right} \rightarrow k = i+1, l = j$$

```
Algorithm: getAverageMVSimilarNeighborBlk(i, j)  {
        cnt = 0; x_sum = 0; y_sum = 0;
        for (k = i − 1; k < i + 1; k++) {
          for (l = j − 1; l < j + 1; l++) {
            if(SAD(Block_{i,j}, Block_{k,l}) < ε &&
             TFBlock[k][l] == true) {
              cnt++;
              x_sum += Block[k][l].motionVector.x;
              y_sum += Block[k][l].motionVector.y;
            } //end if
          }//end for
        } //end for
        motionVector.x = x_sum/cnt;
        motionVector.y = y_sum/cnt;
        return motionVector;
} //end algorithm
```

그림 4. 신뢰 블록을 이용한 MV 보정 방법의 알고리즘

3.2. T&F Block table 의 MV 보정

T&F Block Table이 완성되면 이 중의 False인 블록에 대해 보정을 수행해야 한다. 이 때 신뢰할 수 있는 블록들의 MV들을 이용하여 이들의 MV의 평균으로 보정한다. 그림 4는 위에 설명한 방법에 대한 알고리즘의 슈더 코드이다. 이 때 동일 객체는 유사한 움직임을 가지는 성질과 국소 지역 내에서 유사한 색상을 보유하는 특성을 이용하여 신뢰할 수 있는 블록 선정을 T&F Block table의 값이 true이며, SAD(Sum of Absolute Difference)가 작은 블록들을 신뢰할 만한 블록으로 판단하고 이 들의 평균값을 취했다. 식 (2)는 SAD 계산식을 나타낸다.

$$SAD(Block_{i,j}, Block_{k,l}) = \sum_1^N \sum_1^N |Block_{i,j}(x,y) - Block_{k,l}(x,y)| \quad (2)$$

4. 실험 및 결과

4.1 추가 보정 작업에 따른 연산량 변화 분석

Block의 크기를 M, 개수를 N, Search range 크기를 r이라 할 때 보정 작업에서 Block마다 edge비교 횟수는 4번이므로

추가로 드는 복잡도는 $O(M \cdot N \cdot 4)$이다. 그러나 기존 FRUC의 복잡도 $O(M^2 \cdot N \cdot r^2)$에서 r^2부분이 높은 연산을 차지하기 때문에, 보정 작업에 대한 연산량은 미미하다고 할 수 있다.

4.2 영상에 따른 psnr 비교 분석

표 1은 보간한 영상의 각 방법에 따른 psnr 결과표이다. Stefan, Mobile, Foreman은 움직임이 큰 영상으로 그 중 Stefan, Foreman은 객체가 종, 횡 방향으로 이동하는 특성이 있고 Mobile은 전체적으로 종, 횡 방향과 더불어 줌 확대, 축소 효과도 나타난다. 따라서 대체로 psnr이 낮게 나타났다. 그에 비해 Akiyo, Bridge는 배경과 더불어 객체의 움직임이 작은 영상이기 때문에 psnr값이 높게 나온 걸로 보인다.

Stefan은 배경의 움직임이 적어서 대체로 잘 찾은 MV로 이루어지기 때문에 이 MV를 이용하여 잘못 찾은 MV를 보정하고, Foreman은 화면 자체가 횡 이동하는 특징이 있어 MV들의 성향이 비슷하므로 정확도를 높여 품질을 향상시킨 것으로 보인다. 그러나 Median 방법에선 고정 배경에서 객체가 움직이면 올바른 MV를 찾아도 주변 MV와 달라 이상 점으로 판단되는 문제가 있어 기존 방법보다 성능이 떨어지게 나타난다. 줌 현상이 일어나는 Mobile은 SAD 방식에 부적합하기 때문에 전체적으로 MV가 부정확한 1차 보간 이미지가 만들어진다. 따라서 그림 6에서 후 보정한 2차 보간 이미지도 잘못 찾은 MV들로 인해 비슷하게 나오는 것으로 보인다.

전반적으로 수치가 높은 Akiyo, Bridge의 경우 Median 방법보다 psnr이 미세하게 떨어지지만, 사람의 눈은 미세한 수치의 차이를 못 느끼기 때문에 실제로는 비슷한 품질로 보일 것이며, 그림 7을 보면 기존의 프레임에서 잘못된 부분을 보정해 품질을 향상시켰으므로 합리적이라고 볼 수 있다.

표 1. psnr 비교 분석 표 ($thr_{boundary}$ = 180)

	Without Correction	Median Filter [5]	Ours
Stefan	22.7071	22.3730	23.7643
Mobile	20.5075	22.0962	21.7651
Foreman	28.3203	28.4529	28.6762
Akiyo	36.6024	37.2202	37.0134
Bridge	31.3783	31.5204	31.4026

4.3 $thr_{boundary}$에 따른 psnr 비교 분석

그림 5는 $thr_{boundary}$에 따른 psnr의 결과 비교이다. $thr_{boundary}$에 따른 결과는 제각각 인데, 이는 노이즈, 움직임 등 영상의 특징에 따라 적합한 $thr_{boundary}$가 따로 있다고 볼 수 있다. 따라서 영상마다 적합한 $thr_{boundary}$를 결정하면 더 좋은 품질이 나올 수 있을 것이다.

5. 결론 및 향후 연구

본 논문에서는 보간 영상의 공간적 특성을 고려한 MV 후보정 방법을 제안하였고, 실험 결과를 통해 제안한 방법이 기존의 Bi-lateral 방법보다 높은 성능을 보임을 알 수 있었다. 하지만, 실제로 잘못된 MV가 많을 경우 제안한 방법에선 이러한 MV로 후 처리를 하기 때문에 보정 후에도 품질이 떨어졌고, 영상에서 객체의 움직임이나 줌 확대, 축소와 같은 특성에 따라 성능이 다르게 나타났다. 따라서 보간 영상의 특성에 따른 적응적 보정 방법이 추가적으로 필요하며, 제안한 방식으로 해결할 수 없었던 부분을 추가 보정하기 위해 다른 방법을 통합시켜 개선하는 방식을 고려해봐야 한다. 이에 대한 연구를 추후에 진행할 예정이다.

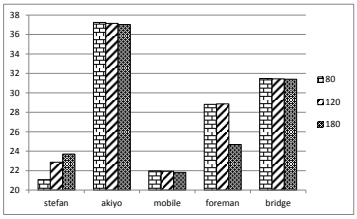

그림 5. $thr_{boundary}$에 따른 결과 비교 (psnr)

(a) (b) (c)

그림 6. Mobile 127번째 프레임 (a) 기존 Bilateral FRUC, (b) Median Filter, (c) 제안한 방법으로 보간한 영상

(a) (b)

그림 7. Akiyo 223번째 프레임 (a) 기존 Bilateral FRUC, (b) 제안한 방법으로 보간한 영상

6. 참고 문헌

[1] 김상철, 정현종, 송인선, 낭종호, “프레임율 변환을 위한 H.264코덱의 움직임 벡터 분석,” *2012 한국 컴퓨터 종합학술대회 논문집*, 제 39권, 제 1호, 164-166쪽, 2012.

[2] D. W. Vincent, A. Blanchfield, P. Klepko, and R.Commun, “Motion Compensated Frame Rate Up-Conversion Part I: Fast Multi-Frame Motion Estimation,” *IEEE Trans. Broadcast.*, Vol. 56, No. 2, pp. 133-141, 2010.

[3] B. D. Choi, J. W. Han, C. S. Kim, and S. J. Ko, “Motion compensated frame interpolation using bilateral motion estimation and adaptive overlapped block motion compensation,” *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 17, No. 4, pp.407-416, 2007.

[4] S. J. Kang, K. R. Cho and Y. H. Kim, “Motion Compensated Frame Rate Up-Conversion Using Extended Bilateral Motion Estimation,” *IEEE Trans. Consumer Electron.*, vol. 53, no. 4, pp. 1759-1757, 2007

[5] S. J. Kang, D. G. Yoo, S. K. Lee and Y. H. Kim, “Design and Implementation of Median Filter based Adaptive Motion Vector Smoothing for Motion Compensated Frame Rate Up-Conversion,” *in Proc. of ICSE ’09*, pp.745-748, 2009.