프레임제거 트랜스코딩을 위한 개선된 움직임 벡터 구성 알고리즘

이성진[0*] 이화세[**] 박시용[*] 이승원[*] 정기동[*]
부산대학교 전자계산학과[*] 밀양대학교 컴퓨터공학과[**]
(arcus, sypark, swlee, kdchung)@melon.cs.pusan.ac.kr[*], hslee@mnu.ac.kr[**]

An Improved Motion Vector Composition Algorithm for Frame Rate Reduction Transcoding

Seong-Jin Lee[0*] Hwa-se Lee[**] Si-Yong Park[*] Seung-Won Lee[*] Ki-Dong Chung[*]
Dept. of Computer Science, Pusan National University[*]
Dept. of Computer Engineering, Miryang National University[**]

요 약

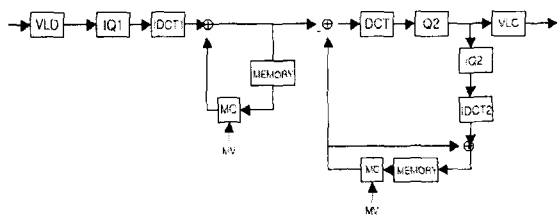
비디오 트랜스코딩을 수행할때, 트랜스코더에서 움직임 벡터를 새로 추출하는 과정은 많은 계산 처리량이 요구된다. 그러므로 실시간으로 비디오스트림을 트랜스코딩하기에는 문제점이 발생한다. 비디오 트랜스코더는 계산복잡도를 감소시키기 위해서 트랜스코더의 인코더부분에서 추출된 움직임 벡터를 트랜스코더의 디코더 부분으로 바로 전송하는 기법을 이용하고 있다. 이러한 기법은 비디오 화질저하를 발생시키므로, 해결책으로 기존의 움직임 벡터를 트랜스코더에서 탐색영역을 작게 하여 정재하는 기법에 관한 연구가 있었다. 본 논문에서는 프레임을 감소시키기 위한 트랜스코딩을 수행할 때 움직임벡터를 구성하는 방법에 대해서 기존의 연구들과 비교하여 개선된 알고리즘을 제안한다. 제안한 알고리즘은 매크로블록이 차지하고 있는 각각의 블록에 대해서 가중치를 적용한다.

1. 서 론

멀티미디어 서비스의 눈부신 발전으로 인해 사용자는 이동환경에서도 비디오스트림을 서비스 받을수 있게 되었다. 하지만 무선망이라는 특성상 제공해줄수 있는 대역폭이 유선망에 비해 월등하게 낮기 때문에 아직까지 이동컴퓨터가 비디오스트림을 서비스받기에는 문제점들이 존재한다. 이에 높은 비트스트림으로 전송되는 비디오 스트림을 낮은 비트율로 변환하는 장치가 필요하다. 트랜스코더는 클라이언트의 환경에 최적화 되도록 비디오 스트림을 변환하는 장치이다[5][6]. 예를 들면, 유선망의 환경에 맞는 비트스트림이 트랜스코더에 입력되면, 트랜스코더는 비트스트림을 완전하게 디코더하여 낮은 대역폭 및 무선망의 환경에 맞게 비트율을 조절하여 클라이언트에게 비디오 서비스를 제공해줄수 있다.

일반적인 트랜스코더구조가 [그림1]에 보여진다. 트랜스코더 구조는 계산복잡성을 감소시키기 위해 여러가지의 구조로 나누어 지지만 DCT영역 트랜스코더를 포함한 다른 트랜스코더는 클라이언트의 환경에 맞게 비트스트림을 조절할 수가 있는 유연성이 부족하기 때문에 본 논문에서는 연쇄적인 픽셀영역 트랜스코더 구조를 사용한다. 일반적으로 계산복잡성을 감소시키기 위해 트랜스코더로 입력되는 움직임벡터는 재사용된다. 이는 상당한 처리량을 감소시킬뿐만 아니라 비디오 화질 역시 다시 움직임추정을 하는것에 비해 크게 차이가 나지 않기 때문이다. 또한, 필요에 따라 트랜스코더로 입력되는 움직임벡터를 정재하여 사용할 수도 있다[1][5]

본 논문에서는 시간당 프레임률을 감소시킬 때 움직임 벡터를 구성하는 개선된 알고리즘을 제안한다. 시간당 프레임률을 감소시키는 이유는 클라이언트의 대역폭이 낮은경우이거나 필요에 따라 전체적인 화질 저하보다는 프레임을 제거함으로써 낭아있는 프레임에 더 많은 비트를 할당함으로 인해 비디오 화질을 높일수 있기 때문이다. 또한, 구성된 움직임 벡터는 필요에 따라 탐색영역을 줄임으로써(약 ±3 내외) 전탐색을 행함으로 인해서 발생하는 상당한 계산복잡성을 줄일수가 있다. 비디오화질 역시 움직임벡터를 새로 추출하는것에 비해 크게 차이가 나지 않는다.



[그림1] Cascaded Pixel-Domain Transcoder

2. 관련연구

2.1 움직임 벡터의 재사용

인터넷상에 있는 서버에서 전송되는 비디오스트림을 트랜스코더가 입력받으면 트랜스코더는 상황에 따라 일정 프레임을 제거시킬수가 있다. 즉, 출력 대역폭이 입력대역폭보다 적을경우, 그리고 클라이언트가 완전한 프레임을 디코딩할 능력이 없을경우, 제한된 비트량으로 특정프레임에 더 많은 비트를 할당할 필요가 있는 경우 등 프레임을 감소시키는 트랜스코딩역할은 매우 유용하다. 하지만, 감소된 프레임 이전의 프레임과 가

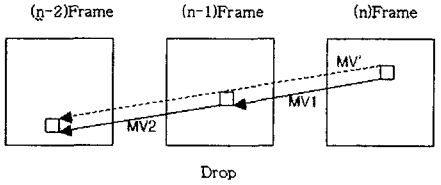
정보통신부에서 지원하는 대학기초연구지원사업으로 수행

장 잘 매칭되는 매크로 블록을 찾아내는 것은 상당한 계산량이 요구된다.

[그림2]는 (n-1)프레임이 제거될때 움직임 벡터를 다시 추출하는 상황을 나타낸다. 즉, (n)프레임의 블록은 (n-2)프레임에서 탐색영역을 정하여 가장 잘 매칭되는 블록을 찾아 MV'를 구한다. 이런과정을 수행하는 것은 처리량이 많아 실시간 트랜스코딩에는 적절하지 못함으로 (1)과 같이 MV1과 MV2를 재사용함으로 인해 계산량을 줄일수가 있다[1].

MV' = MV1 + MV2 ------ (1)

여기서 각 움직임 벡터를 더한 MV'는 탐색영역을 줄임으로 인해 계산량을 감소시키는 것과 동시에 비디오 화질을 높일수가 있다. 즉, 인터넷상에 있는 서버에서 탐색영역을 ±15로 했으면 트랜스코더에서는 탐색영역을 ±2 로 함으로 인해 비디오 화질은 충분히 인터넷상에 있는 서버에서의 움직임벡터를 구한것과 거의 근접하게 된다.

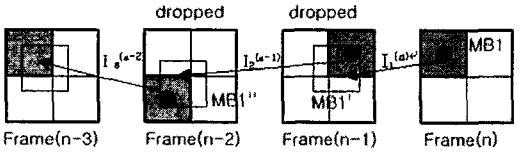


[그림2] 프레임이 제거되었을시 움직임벡터 추출

하지만, 매칭블록은 각 매크로블록의 경계부분에 있지 않기 때문에 문제점이 발생한다.

2.2 FDVS 및 ADVS기법

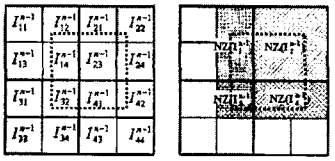
[2]에서 제안한 방법은 매크로블록의 움직임 벡터를 구할 때 가장 영역을 많이 차지하는 부분의 움직임 벡터를 선택하는 알고리즘이다. 즉, [그림3]에서 보는거와 같이 (n-1)번째 프레임과 (n-2)번째 프레임이 제거 되었을 때, (n)번째 프레임의 매크로 블록MB_1과 가장 잘 일치하는 (n-1)번째 매크로 블록MB_1'의 움직임 벡터는 (n-1)번째 프레임에서 가장많은 영역을 차지하는 두번째 매크로 블록의 움직임벡터 $I_2^{(n-1)}$를 선택하게 된다. 마찬가지로 (n-1)번째 매크로블록과 가장 잘 일치하는 (n-2)번째 매크로블록MB_1"의 움직임벡터는 가장 많은 영역을 차지하는 세번째 매크로블록의 움직임 벡터 $I_3^{(n-2)}$을 선택한다.



[그림3]Forward dominant vector selection(FDVS) composition

이렇게 선택된 매크로 블록들의 움직임 벡터들은 다음식처럼 $I_3^{(n-2)} + I_2^{(n-1)} + I_1^{(n)}$ 와 같이 모두 더해져서 정재되어진다.

[3]에서 제안한 방법은 매크로 블록의 움직임 벡터를 선택할 때 단순하게 가장 많은 영역을 차지하는 매크로 블록의 움직임 벡터를 선택하는 것이 아니라 각 블록의 DCT계수의 수를 파악한다. 즉, [그림3]과 같이 가장 잘 일치하는 매크로 블록이 네개의 인접 블록에 걸쳐 있을 때, 일치하는 매크로 블록(점선부분)이 포함되어 있는 블록의 DCT계수의 수를 구한다. 각 블록의 DCT계수의 수를 구한다음, 매크로 블록의 경계에 한해서 각 매크로 블록이 포함되는 8X8블록의 DCT계수의 수를 모두 더한다. 그림에서는 일치하는 매크로 블록이 네개의 블록을 모두 포함하면 매크로 블록 전체의 DCT계수의 수를 구하고, 두개의 블록만 포함한다면 네개의 블록중 두개의 블록의 DCT계수의 수만 구한다. 이렇게 하는 이유는 0이아닌 DCT계수의 수가 많은 블록일수록 많은 정보가 포함되어 있기 때문에 단순하게 많은 영역만 차지하는 매크로 블록의 움직임 벡터를 선택하는 것보다 훨씬 현실적이다.

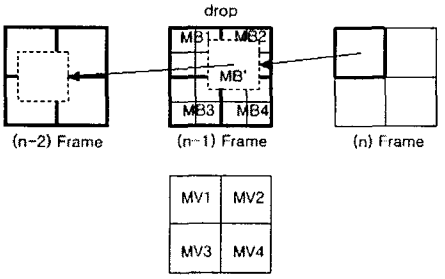


[그림 4] Activity-dominant vector selection(ADVS)

즉, 네개의 블록을 포함하는 매크로 블록의 움직임벡터보다는 두개의 블록만 포함하는 매크로 블록의 움직임 벡터가 선택될 수 있다.

3. 제안하는 움직임 벡터 구성 알고리즘

본 논문에서는 한 개이상의 프레임을 제거하였을 때 제거된 프레임에서 최적의 움직임 벡터를 구하는 알고리즘을 제안한다. 그림에서와 같이 (n-1)프레임이 제거 되었을시 가장 잘 매칭되는 매크로 블록 MB' 의 움직임벡터는 MV1 - MV4 중에 어느것을 선택하는가 하는 것이 문제이다.



[그림 5] 프레임이 제거될 때 움직임 벡터 구성

매크로블록 MB1 - MB4 각각은 8X8블록 네개로 구성되어 있다. 각 블록은 0이 아닌 양자화된 DCT계수를 가진다. 즉, 0이아닌 DCT계수의 개수를 구할수 있다. 이것은 트랜스코더로 입력되는 비디오스트림이 VLD과정을 거친후에 추출되어 진다. 여기서 네개의 매크로 블록의 총 비율을 100으로 본다. 그러면 각 매크로블록은 공평하게 25씩 가중치가 분배되어 진다. 다시, 한 개의 매크로블록은 4개의 블록으로 구성되어 있기 때문에 각 블록의 가중치는 6.25로 설정한다. 그러면, 제거된 프레임에서 n번째 프레임과 가장잘 매칭되는 매크로 블록 MB' 가 4개의 매크로 블록에 걸쳐 있을 때, 다음과 같이 적용한다.

1. MB' 가 1개의 블록을 포함할 때,
   매크로 블록의 4개블록 DCT계수의 개수의 합 X 6.25
2. MB' 가 2개의 블록을 포함할 때,
   매크로 블록의 4개블록 DCT계수의 개수의 합 X 12.5
3. MB' 가 3개의 블록을 포함할 때,
   매크로 블록의 4개블록 DCT계수의 개수의 합 X 25

위의 방법데로 적용한 값에 다음과 같은 알고리즘을 적용한다.

- Block(Sum) = The Number of DCT Coefficient
- MB(Sum) = 4개의 Blcok(Sum)
- $MB1_{(Weight)}$, $MB2_{(Weight)}$, $MB3_{(Weight)}$, $MB4_{(Weight)}$ =
  - Select One { MB(Sum) X 6.25, MB(Sum) X 12.5, MB(Sum) X 25 }
- MB' (Best Matching Block) = Max{ $MB1_{(Weight)}$, $MB2_{(Weight)}$, $MB3_{(Weight)}$, $MB4_{(Weight)}$ }
- The best MV' = Best Matching Block 의 MV

[그림5]에서의 예를 들면, MB1에서 네개 블록의 0이아닌 DCT 계수의 개수 총합을 구한다. 마찬가지로 MB2,MB3,MB4 도 같은 방법을 적용한다. 다음으로, (n)번째 프레임의 매크로 블록과 가장 잘 매칭되는 매크로 블록 MB' 가 (n-1)번째 프레임에서 차지하고 있는 각 매크로블록의 블록수를 구한다. MB' 가 MB2처럼 네개의 블록모두를 포함하고 있는경우에는 가중치를 25로 준다. MB1과 MB4처럼 2개의 블록을 포함하고 있는경우는 가중치를 12.5로 준다. 마찬가지로 한 개의 블록을 포함하고 있는 MB3는 가중치를 6.25로 준다. 마지막으로 각 매크로블록의 0이아닌 DCT계수의 총합과 각각의 가중치를 곱하여 가장 높은 값을 가지는 매크로 블록의 움직임 벡터를 선택한다.

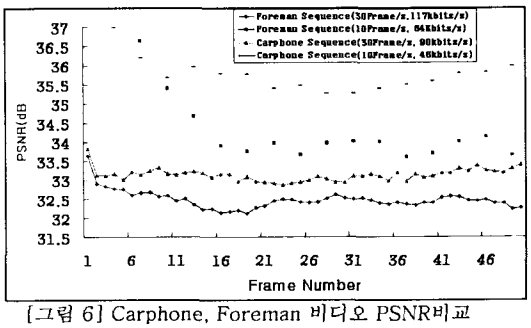
4. 실험

4.1 실험환경

실험을 위해서 본 논문에서는 Carphone과 Foreman 비디오를 사용하였다. 각 비디오는 H.263 QCIF포멧으로 되어있으며 처음 50프레임을 추출하여 PSNR값을 측정하였다. 실험환경으로는 각 프레임을 동등하게 양자화하여 프레임을 제거하지 않았을때와 2개의프레임을 제거하였을때의 PSNR값을 비교하여보았다. 그리고 각 비디오의 평균PSNR값을 측정해 보았다. 본 논문에서의 실험은 제안한 알고리즘을 적용하지 않았고, 프레임을 제거하였을때와 프레임을 제거하지 않았을때의 PSNR값을 서로 비트율을 변경하면서 비교해보았다. 우리는 제안한 알고리즘을 적용하기 위해 계속적으로 트랜스코더를 구현중에 있다.

4.2 결과

비트율을 일정하게 유지하였을때의 각 비디오의 PSNR값이 [그림6]에 보여진다. Foreman비디오를 살펴보면, 117Kbps의 속도로 완전한 비디오스트림을 전송하였을 때보다 64Kbps의 속도로 2개의 프레임을 제거하였을때의 PSNR값이 더 높은 것을 알 수 있다. 이것은 비트율이 낮아지더라도 제거된 프레임에 할당할 수 있는 비트량을 남아있는 프레임에 더 많은 비트를 할당함으로 인해 비디오화질을 높일수 있다는 것을 보여준다. Carphone비디오 역시 같은결과를 보여준다.



[그림 6] Carphone, Foreman 비디오 PSNR비교

[표 1]은 각각의 H.263비디오의 프레임율과 비트율에 따라 평균 PSNR을 구한값을 나타낸 것이다. 역시 프레임을 제거하였을때의 비디오가 화질이 다소 우수한 것을 알수가 있다. 하지만, 각각의 프레임이 아닌 전체적인 비디오Quality를 고려하면 초당 30프레임의 비디오가 더 우수할 것이다.

[표 1] Carphone, Foreman 비디오의 평균 PSNR

H.263비디오	프레임율, 비트율	평균 PSNR
Foreman	30Frame/s, 117Kbps	32.47
	10Frame/s, 64Kbps	35.61
Carphone	30Frame/s, 90Kbps	33.14
	10Frame/s, 46kbps	36.86

5. 결론과 향후 연구방향

실시간 비디오 트랜스코딩에 있어서 계산복잡성을 감소시키는 것은 매우 중요하다. 대부분의 트랜스코더는 움직임 벡터를 재사용하고 있다. 이것은 처리량을 상당히 감소시킬뿐만 아니라 비디오 화질 역시 크게 저하되지 않기 때문이다. 하지만, 단순하게 움직임 벡터를 재사용하기에는 문제점들이 발생한다.

본 논문에서는 클라이언트의 환경에 맞게 비트스트림을 조절하기 위해 프레임을 감소시키는 트랜스코딩을 고려하였다. 프레임을 감소할 때 움직임 벡터는 새로 추출되는 것이 아니라 기존의 움직임벡터를 모두 합한값을 사용한다. 하지만, 가장 잘 매칭되는 매크로 블록은 항상 인접 매크로 블록의 경계부분에 있는 것은 아니다. 이에 본 논문에서는 연속적인 프레임들 사이에 제거된 프레임에서 가장 적합한 움직임 벡터를 선택하는 알고리즘을 제안하였다. 우리는 H.263비디오 인코더 및 디코더 코덱을 기반으로 프레임을 제거하는 트랜스코더를 구현중에 있다. 향후, 본 논문에서 제안한 움직임벡터 구성 알고리즘을 적용하여 그 효율성 및 우수성을 입증해 볼 것이다.

6. 참고문헌

[1] Jeongnam Youn, Ming-Ting Sun, and Chia-Wen Li n" Motion estimation for high-performance transcoders," *IEEE Trans. Consumer Electron.*, vol.44, pp.649-658, Aug.1998

[2] Jeongnam Youn, Ming-Ting Sun, and Chia-Wen Lin, "Motion vector refinement for high performance transcoding," *IEEE Trans. Multimedia*, vol. 1, no. 1, pp. 30-40, Mar. 1999.

[3] Mei-Juan Chen, Ming-Chung Chu, and Chih-Wei Pan, "Efficient Motion-Estimation Algorithm for Reduced Frame-Rate Video Transcoder" *IEEE Trans. Circuits and systems for video technology*, vol. 12, no. 4, pp. 269-275, Apr.2002

[4] ITU-T Rec. H.263, "Video codec for low bit rate communication," May 1996.

[5] Björk N. and Christopoulos C., " Transcoder Architectures for video coding" , IEEE Trans. Consumer Electronics, Vol. 44, No. 1, pp. 88-98, February 1998.

[6] R. J. Safranek, C. Kalmanek, and R. Garg, " Methods for matching compressed video to ATM networks," in *Proc. Int. Conf. Image*, Washington, DC, Oct. 1995