움직임이 큰 비디오에 효율적인 비디오 분할 방법

박민호*, 박래홍**
서강대학교 전자공학과
e-mail: *mhpark@eevision1.sogang.ac.kr **rhpark@mail.sogang.ac.kr

An efficient method for segmentation of fast motion video

Min-Ho Park and Rae-Hong Park
Dept. of Electronic Engineering, Sogang University

요 약

기존의 비디오 분할 방법은 밝기의 변화가 큰 영상이나 움직임이 큰 영상에 대해서는 정확한 분할이 이루어지지 않았다. 본 논문은 움직임 정보를 이용하여 움직임이 큰 영상에서 좀 더 정확하게 비디오를 분할할 수 있는 방법을 제안한다. 이를 위해 블록 정합 알고리즘을 이용하여 얻어진 움직임 벡터로부터 움직임 유사도를 찾는 방법을 제안한다. 또 연속된 프레임에서 픽셀의 차이 값을 계산할 때 motion blur 로 생기는 오차를 각 블록의 움직임 크기로 보상하여 좀 더 정확한 픽셀의 차이 값을 계산하는 방법을 제안한다. 이렇게 얻어진 두 가지 정보를 이용하여 discontinuity value 를 계산한다. 움직임이 많은 액션 영화 3 편에 대해 실험한 결과 제안한 방법이 기존의 움직임 유사도와 픽셀 차이 값을 구하여 샷 경계 검출을 하는 방법보다 좀 더 정확한 샷 경계 검출을 하고 있다는 것을 보여준다.

1. 서론

최근 몇 년 사이 인터넷과 핸드폰, PDA 등 정보기기의 빠른 확산으로 우리가 접하는 정보의 양이 급격하게 늘어나게 되었고 이런 정보를 효율적으로 저장, 관리, 검색하는 방법의 중요성이 크게 부각되었다. 우리가 접할 수 있는 정보는 크게 텍스트 정보와 멀티미디어 정보로 나눌 수 있다. 현재 텍스트 정보의 검색은 어느 정도 이루어져 있지만 멀티미디어 정보의 검색은 초보적인 수준이다. 본 논문에서는 그 중요함이 커지고 있는 멀티미디어 정보의 관리와 검색, 특히 동영상 정보의 효율적인 관리, 검색을 위한 전처리 과정으로서의 샷 경계 검출 방법에 대해 연구하였다.

샷 경계 검출을 할 때 사용하는 영상의 특징들은 색깔, 모양, 움직임, 경계선, 히스토그램 등 여러 가지가 있다. Zeeshan 과 Shah. [1]는 histogram intersection 과 global motion 을 이용하여 샷 경계 검출을 하였고, Ngo 등 [2]은 spatio temporal slices 를 이용하여 카메라 움직임과 특수효과로 전환되는 샷을 인식하여 샷 경계 검출을 하는 방법을 제안하였다. 하지만 이전의 연구들은 대부분 움직임이 큰 영상이나 밝기가 빠르게 변하는 영상에서는 정확한 샷 경계 검출을 하지 못하고 있다.

본 논문에서는 블록 정합 알고리즘 (BMA: block matching algorithm) [6]를 사용하여 연속된 3 장의 프레임에서 움직임 정보를 얻은 뒤 블록 단위로 움직임을 비교하여 움직임 유사도를 계산하고, BMA 과정에서 얻어지는 연속된 두 프레임에서의 각 블록마다의 픽셀의 차이 값 중 motion blur 로 생기는 오차부분을 움직임 크기로 보상하여 움직임이 큰 영상에서 좀 더 정확하게 샷 경계 검출을 하기 위한 방법을 설명한다.

본 논문의 구성은 다음과 같다. 2 절에서는 제안하는 방법을 설명한다. 구체적으로 BMA 를 이용하여 움직임 벡터를 찾고 움직임 크기를 계산하는 방법을 설명하고, 연속된 3 장의 프레임에서 움직임 유사도를

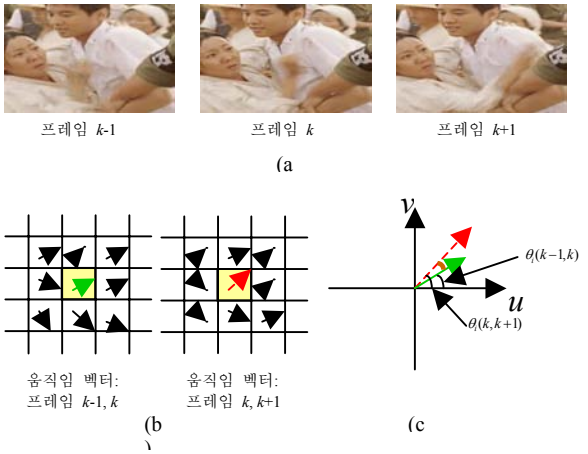

그림 1. 연속된 3 장의 프레임에서의 움직임 유사도를 찾는 방법: (a) 연속된 3 장의 프레임 *k*-1, *k* 그리고 *k*+1. (b) BMA 를 이용하여 (a)의 영상에서 찾은 움직임 벡터. (c) 움직임 벡터 중 정합되는 블록의 움직임 벡터의 방향의 유사도를 구하는 방법.

계산하는 방법을 설명한다. 또 BMA 과정에서 얻어지는 연속된 두 프레임에서의 각 블록마다의 픽셀의 차이 값 중 motion blur 로 생기는 오차부분을 움직임 크기를 이용하여 보상해주는 방법을 설명한다. 2 절의 끝에서는 앞에서 구한 두 가지 정보 (움직임 유사도와 움직임 크기에 의해 보상된 픽셀의 차이 값)를 이용하여 discontinuity value 를 구하고 샷 경계를 검출하는 방법을 설명한다. 3 절에서는 실험 결과 및 토의를 보여주고, 4 절에서 본 논문의 결론을 보였다.

2. 제안하는 방법

1) 움직임 크기

본 논문에서는 움직임 벡터를 찾기 위해 BMA 를 사용한다. BMA 는 연속된 두 장의 프레임을 이용하여 전 프레임의 블록과 가장 정합이 잘 되는 블록을 현재의 프레임에서 찾음으로써 움직임을 찾는 방법이다. 이렇게 찾아진 움직임 벡터에서 각 블록의 움직임 크기 $M^i(k,k+1)$ 는

$$M^i(k,k+1)=\sqrt{u_i^2(k,k+1)+v_i^2(k,k+1)} \qquad (1)$$

로 나타낸다. 여기서 $u_i(k,k+1)$ 와 $v_i(k,k+1)$ 는 k 와 k+1 번째 프레임에서 구해진 움직임 벡터들 중 i 번째 블록의 움직임 벡터의 수평과 수직 성분을 나타낸다. 이 논문에서는 영상 크기 $X\times Y$ 인 영상에 대해 블록 크기 $M\times M$, 탐색영역 $3M\times 3M$ 인 전역 탐색 알고리즘 (full search algorithm)을 이용한다.

2) 움직임 유사도

이 절에서는 BMA 로 얻어진 움직임 벡터를 이용하여 연속된 프레임에서 매칭되는 각 블록의 움직임 방향을 비교 함으로서 두 프레임의 움직임 유사도를 계산하는 방법을 설명한다. 한 샷에서 물체와 배경의 움직임 방향은 거의 일정하기 때문에 하나의 샷 안에 있는 물체와 배경의 움직임 방향도 거의 일정하다 [3]. 또 각각의 샷은 서로 다른 물체와 배경을 갖는다. 따라서 물체와 배경의 움직임 방향이 갑작스럽게 변하는 부분이나 물체나 배경 자체가 갑자기 변하는 부분은 샷 경계로 가정할 수 있다. 그러나 프레임마다 이 둘을 나누는 일이 쉽지 않기 때문에 다른 방법으로 물체와 배경의 움직임 방향을 비교하여 프레임 간의 움직임 유사도를 계산하려고 한다.

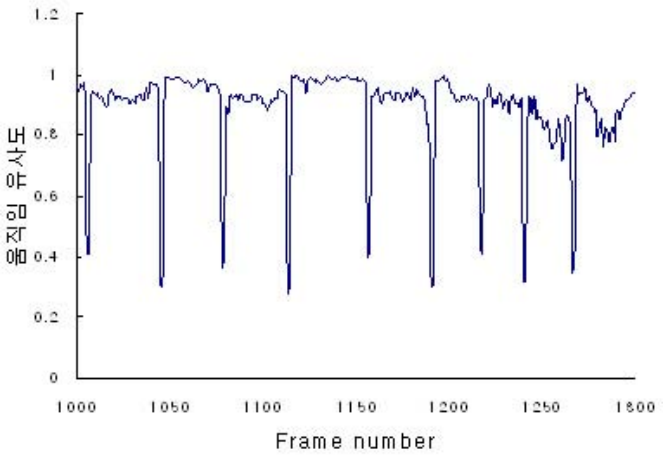

그림 2. 연속된 프레임의 움직임 유사도: 연속된 3 장의 프레임에서의 움직임 유사도 $M^s(k-1,k,k+1)$ 값을 프레임에 따라 표시한 그래프.

움직임 유사도를 계산하는 방법은 여러 가지가 제안되었다. 제안된 방법들 중 대부분은 프레임의 특정 지역만은 고려하여 카메라의 움직임에 의해 생기는 대표되는 한 움직임을 찾아서 비교 [4]하거나 motion texture 를 사용하여 움직임 유사도를 계산 [5]하고 있다. 하지만 이런 방법들은 물체의 세세한 움직임은 고려하고 있지 않다.

본 논문에서는 BMA 를 통해 연속된 프레임에서의 정합되는 블록의 움직임을 비교함으로써 작은 움직임까지도 고려하여 좀 더 정확한 움직임 유사도를 구하려고 한다. 그림 1 에서 보는 것과 같이 연속된 두 프레임 *k*-1, *k* 그리고 *k*, *k*+1 로부터 구한 두 움직임 벡터에서 동일한 위치에 있는 블록 i 의 움직임의 방향을 각 $\theta_i(k-1,k)=\tan^{-1}(u_i(k-1,k)/v_i(k-1,k))$ 과 $\theta_i(k,k+1)$ 로 표시한다. 이 두 각의 사이 각이 기준 각 Th_θ 이하이면 두 블록의 움직임의 방향은 유사하다고 가정하고 블록마다 움직임 유사도 $M_i^s(k-1,k,k+1)$ 를

$$M_i^s(k-1,k,k+1)=\begin{cases}1, & \text{if } |\theta_i(k,k+1)-\theta_i(k-1,k)|\le Th_\theta \\ 0, & \text{otherwise}\end{cases} \qquad (2)$$

로 표현한다. 식 (2)를 모든 블록에 대해 계산한 후 움직임의 방향이 유사한 블록의 개수를 전체 블록의

수 $N_b = XY/M^2$ 로 나누어 연속된 프레임끼리의 움직임 유사도를 계산한다. 즉 움직임 유사도 $M^s(k-1,k,k+1)$ 는

$$M^s(k-1,k,k+1) = \frac{1}{N_b}\sum_{i=1}^{N_b} M_i^s(k-1,k,k+1) \quad (3)$$

과 같이 표현한다. 그림 2 에서 보는 것과 같이 움직임 유사도 $M^s(k-1,k,k+1)$ 는 k 번째 프레임에서 샷 경계가 있으면 급격하게 변한다. 이는 이 전 샷과 현재의 샷에서 나타나는 물체와 배경이 서로 다르고 움직임 방향도 다르기 때문이다.

3) 움직임 크기를 이용한 픽셀 차이 값 보상

연속된 두 프레임의 유사도를 계산할 때 쓰이는 가장 간단한 방법은 정합되는 각 픽셀의 크기 값의 차를 계산하는 것이다. 그러나 이 방법은 갑작스런 밝기의 변화가 있거나 배경과 물체의 움직임이 있을 때는 부정확한 결과를 보인다. 이 방법에 비해 한 단계 발전한 방법은 BMA 를 이용한 방법으로 픽셀의 차이 값 $d_i(k,k+1)$ 을

$$d_i(k,k+1) = \min_{1\le j\le(2M+1)^2} \frac{1}{M^2}\sum_{x=1}^{M}\sum_{y=1}^{M}(I_i^j(x,y,k+1) - I_i(x,y,k))^2 \quad (4)$$

으로 표현 할 수 있다. $I_i(x,y,k)$ 는 k 번째 프레임의 i 번째 블록에서의 픽셀 값이고, $I_i^j(x,y,k+1)$ 는 k+1 번째 프레임에서 k 번째 프레임의 i 번째 블록을 포함하고 있는 탐색영역의 블록 중 j 번째 블록의 픽셀 값을 나타낸다.

즉 연속된 두 프레임 에서 N_b 개로 나눠진 블록들을 각각에 대해 그 블록의 탐색 영역에서 가장 유사한 블록을 찾는 것이기 때문에 움직임이 있는 영상에서도 좀 더 정확한 픽셀의 차이 값을 계산할 수 있다.

그러나 이 방법도 움직임이 큰 영상에서는 정확도가 떨어진다. 즉 움직임이 큰 영상의 경우 motion blur 가 심하게 일어나기 때문에 정확한 블록 정합을 하지 못하고 두 프레임에서의 픽셀의 차이 값에 오차가 생기게 된다. 이런 현상은 움직임이 클수록 더욱 심하게 나타나 샷 경계가 아닌 곳을 샷 경계로 인식하게 만들기도 한다. 이런 부분이 액션이 많은 영상의 샷 경계 검출을 어렵게 하는 부분이기도 하다.

이 절에서는 두 프레임에서의 픽셀의 차이 값 중 motion blur 로 인해 생기는 오차를 움직임 크기로 보상하여 움직임이 큰 영상에서 좀 더 정확한 픽셀의 차이 값을 계산하는 방법을 설명한다. Motion blur 로 인해 생기는 오차는 움직임 벡터의 크기에 비례하여 나타난다. 그래서 식 (4)에 의해 계산된 각 블록마다의 픽셀의 차이 값 $d_i(k,k+1)$ 에 블록의 움직임 벡터의 크기에 반비례하는 가중치를 곱하여 motion blur 로 생기는 오차를 보상한다. 가중치 ω 는

$$\omega = e^{-\left\{\frac{u_i^2(k,k+1)}{2\sigma_u^2(k,k+1)} + \frac{v_i^2(k,k+1)}{2\sigma_v^2(k,k+1)}\right\}} \quad (5)$$

로 나타낸다. $\sigma_u(k,k+1)$ 와 $\sigma_v(k,k+1)$ 는 한 프레임에서의 $u_i(k,k+1)$ 과 $v_i(k,k+1)$ 의 표준편차를 말한다. 이 때 각 축마다 움직임 크기가 0 인 부분은 motion blur 에 아무런 영향을 주지 않기 때문에 제외하고 계산한다. 각 블록에서의 움직임 크기를 이용하여 보상된 두 프레임에서의 픽셀의 차이 값 $D_i(k,k+1)$ 는

$$D_i(k,k+1) = \begin{cases} d_i(k,k+1), & \text{if } u_i(k,k+1) = v_i(k,k+1) = 0 \\ d_i(k,k+1)\cdot\omega, & \text{otherwise} \end{cases} \quad (6)$$

로 나타낸다. 즉 가중치 ω 와 $d_i(k,k+1)$ 를 서로 곱하여 각 블록에서 큰 움직임으로 생긴 픽셀 차이 값의 오차를 보상하게 된다. 최종적으로 프레임의 모든 블록에 대해 식 (6)을 계산을 하여 얻은 연속된 프레임간의 픽셀의 차이 값 $D(k,k+1)$ 는

$$D(k,k+1) = \sum_{i=1}^{n_b} D_i(k,k+1) \quad (7)$$

로 나타낸다.

4) Discontinuity value 의 계산 및 샷 경계 검출

이 절에서는 앞에서 구한 두 가지 특징을 이용하여 움직임이 많은 영상에서 샷 경계 검출을 할 때 사용하는 threshold 에 좀 덜 민감한 discontinuity value 를 찾는 방법을 알아본다. 우리는 앞에서 움직임 유사도 $M^s(k-1,k,k+1)$ 값이 샷 경계부분에서 샷 경계가 아닌 부분보다 작은 값을 갖는 것을 확인하였다. 또 움직임 크기에 의해 보상된 두 프레임간의 픽셀의 차이 값 $D(k,k+1)$ 은 샷 경계에서 그 이외의 부분보다 더 큰 값을 갖는다는 것을 알았다. 하지만 움직임이 큰 부분에서는 샷 경계의 값보다는 작지만 샷 경계가 아닌 부분보다는 여전히 상대적으로 큰 값을 보이고 있다. 이런 값이 움직임이 많은 영상에서 적합한 threshold 를 결정하는데 어려움을 주고 잘못된 샷 경계 검출결과를 발생하게 하는 부분이다. 따라서 샷 경계에서의 큰 값은 그대로 유지하면서 샷이 아닌 부분에서 큰 움직임 때문에 생기는 값을 움직임 유사도 값을 이용하여 줄여주면 좀 더 움직임에 둔감한 discontinuity value 를 찾을 수 있을 것이다. Discontinuity value $Z(k,k+1)$ 는

$$Z(k,k+1) = D(k,k+1)\exp\left(-\frac{C_1 * (M^s(k-1,k,k+1))}{\underset{-5<i<1}{Avg}(M^s(k+i-1,k+i,k+i+1))}\right) \quad (8)$$

로 나타낸다.

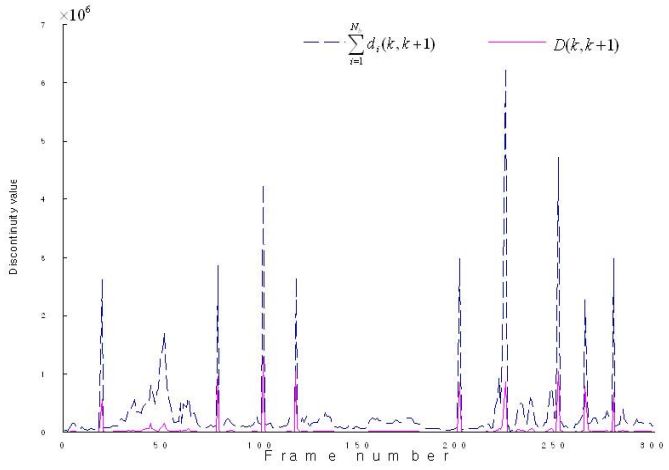

그림 3. Discontinuity value.

한 샷 안에서 큰 움직임이 발생하더라도 그 움직임 방향은 일정하기 때문에 움직임 유사도는 크게 나타난다. 이 값을 exponent 를 취하면 아주 작은 값이 되기 때문에 큰 움직임으로 인해 생긴 픽셀 차이 값의 오차를 줄여줄 수 있다. 이렇게 구해진 discontinuity value 는 그림 3 에서 확인할 수 있다. 샷 경계 검출을 위한 기준은

$$Z(k,k+1) = \max_{-5<i<1}(Z(k+i,k+i+1)) \quad \& \; Z(k,k+1) \geq C_2 * \underset{-5<i<1}{Avg}(Z(k+i,k+i+1)) \tag{9}$$

와 같이 나타낸다. Discontinuity value 와 움직임 유사도 값의 특징을 고려한 식 (9)를 만족하는 부분을 샷 경계로 검출을 하게 된다.

3. 실험 결과 및 토의

본 논문에서는 움직임이 많은 액션 영화 3 편을 실험 영상으로 사용하였다. Ground truth 는 사람이 직접 보면서 확인하였고 특수효과로 전환되는 샷 경계를 제외한 hard cut 에 대한 부분만을 고려하였다. 표 1 은 실험 결과를 나타내고 있다. Ground truth 는 실제 샷의 수를 나타내고, 정확하게 검출한 샷의 개수는 correct (N_c)로, 샷인 곳을 검출하지 못하고 지나간 개수는 miss (N_m)로, 샷이 아닌 곳을 잘못 검출한 개수는 false (N_f)로 표시하였다. 그리고 샷 경계 검출의 정확도를 Precision = $N_c / N_c + N_f$ 과 Recall = $N_c / N_c + N_m$ 로 나타내었다.

그림 3 의 점선과 실선의 그래프를 비교하면 frame number 50 근처에서의 점선의 값은 큰 움직임에 의해 샷이 아닌 부분에서도 값이 크게 나타나지만 최종적으로 계산된 discontinuity value 는 움직임 유사도까지 고려함으로써 큰 움직임이 있는 영상에서도 좀 더 정확한 값을 찾을 수 있었다. 이는 global motion 이나 카메라 움직임을 이용하여 영상에서 대표되는 움직임만을 고려하지 않고 물체와 배경의 작은 움직임까지 세세하게 고려하였기 때문이다. 표 1 의 precision 과 recall 의 결과도 좋은 결과를 보여주고 있다.

표 1. 샷 경계 검출 결과.

Test Sequences	Ground Truth	Correct (N_c)	Miss (N_m)	False (N_f)	Recall	Precision
Movie 1	172	164	7	1	95.9%	99.4%
Movie 2	100	96	3	1	96.9%	98.9%
Movie 3	127	123	1	3	99.2%	97.6%
합계	399	383	11	5	97.3%	98.7%

4. 결론

본 논문에서는 BMA 를 통해 얻어진 움직임 정보를 이용하여 연속된 세 장의 프레임에서 움직임 유사도를 계산하고, 연속된 두 프레임에서 각 블록마다의 픽셀의 차이 값 중 motion blur 로 인해 생기는 오차부분을 움직임 크기로 보상하여 움직임이 많은 영상에서 정확하게 샷 경계 검출을 할 수 있는 방법을 제안하였다. 실험 결과와 같이 움직임이 큰 영상에서 제안한 방법이 기존의 샷 경계 검출방법보다 효과적인 것으로 나타났다. 하지만 폭발이나 번개와 같이 갑작스런 밝기의 변화가 있는 영상과 hard cut 이외의 다른 종류의 샷 경계에 대해서는 아직 정확한 검출을 하지 못하고 있다. 앞으로 좀 더 정확한 움직임 유사도의 계산방법 연구와 더불어 여러 종류의 샷 경계에서 discontinuity value 가 어떤 특징을 갖는 지 고찰하고 그 특징을 이용하여 샷 경계 검출을 하는 방법에 관한 연구가 필요하다.

참고문헌

[1] Z. Rasheed and M. Shah, "Scene boundary detection in hollywood movies and TV shows," in *Proc. IEEE Computer Vision and Pattern Recognition 2003*, vol. 2, pp. 343-348, Madison, Wisconsin, USA, June, 2003.

[2] C.W. Ngo, T.C. Pong, and H.J. Zhang, "Motion analysis and segmentation through spatio-temporal slices processing," in *Proc. International Conference on Image Processing 2003*, vol. 12, no. 3, pp. 341-355 Barcelona, Spain, Sep. 2003.

[3] D. Arijon, *Grammar of the Film Language*, Los Angeles, CA: Silman-James Press, 1976.

[4] D.J. Lan, Y.F. Ma, and H.J. Zhang, "A novel motion-based representation for video mining," in *Proc. International Conference on Multimedia and Expo 2003*, vol. 3, pp. 469-472, Baltimore, Maryland, USA, July 2003.

[5] Y.F. Ma, and H.J. Zhang, "Motion texture: a new motion based video representation," in *Proc. International Conference on Pattern Recognition 2002*, vol. 2, pp. 548-551, Quebec, Canada, Aug., 2002.

[6] F.Moschetti, M.Kunt, and E.Debes, "A statistical adaptive block-matching motion estimation," *IEEE Trans. Circuits and Systems for Video Technology 2003*, vol. 13, no. 5, pp. 417-431, May 2003.