'96 대한전기학회 하계학술대회 논문집 (1996. 7. 22~24)

Optical flow를 이용한 motion estimation에 관한 연구

* 변 채웅, 김 재영, 정 진현
광운대학교 제어계측공학과

A study of motion estimation with optical flow

Chaeung Byun, Jaeyoung Kim, Chinhyun Chung
Control & Instrumentation Engineering, Kwangwoon Univ.

Abstract

The purpose of image sequence coding is to reduce the spatio-temporal redundancies. For the spatial redundancies, we can use the transform coding such as DCT. In this paper, the optical flow method is applied to solve the problem of temporal redundancies. There are several estimation methods like block matching method and pel-recursive method. Block matching method is easy for a hardware implementation because of the computational simplicity. So, it is now used as the estimation method in MPEG-1, MPEG-2, and H.261. We compared the merits and demerits of the optical flow method and the block matching method in this paper.

1. 서 론

영상압축의 목적은 영상이 가지는 spatial redundancy와 temporal redundancy를 제거하여 작은 정보량으로 그 영상을 저장하거나 전송하는 것이다. Spatial redundancy는 DCT와 같은 transform 코딩방식이나 waveform 코딩방식을 이용하여 줄이고, temporal redundancy는 motion compensation을 이용하여 줄일 수 있다. 현재, 표준으로 제정되어 있는 MPEG도 DCT와 motion compensation을 이용하고 있으며, block matching 방법을 이용하여 motion compensation을 한다. 그러나, block matching 방법은 추정한 움직임 벡터에 대한 신뢰성이 떨어지며, 블록화 현상과 같은 여러 가지 단점이 있다. 움직임 벡터를 예측하기 위한 방법은 gradient를 이용한 방법, pel-recursive 방법, block matching 방법, frequency-domain 방법으로 4가지로 나눌수 있다. 본 논문에서는 적용한 optical flow는 gradient를 이용한 방법으로 image sequence 분석과 같은 응용분야에 적용되어 왔다. 이 방법은 움직임궤적을 따라 image의 세기(휘도성분)가 변하지 않는다는 가정에 근거하고 있다. block matching 방법이 두 연속된 frame사이에서 블록 단위로 가장 matching이 잘되는 블록을 찾아 움직임 벡터를 구하는 것에 비해 optical flow를 이용한 방법은 각각의 픽셀에 대해 optical flow를 계산하기 때문에 각각의·픽셀에 대한 아주 세밀한 움직임 벡터가 구해진다. 따라서, block matching 방법에 비해 전송을 위한 정보량이 증가하게 되는 단점이 있다. 그리고, block matching 방법에 비해 계산과정이 복잡하기 때문에 하드웨어로 적용하기가 힘드나, 움직임 벡터를 더 정확하게 찾으며, 계산시간에 대한 부하가 줄어듬에 따라 많은 범위에 적용될 것으로 보인다. 본 논문에서는 Horn-schunck 방법과 Lucas-kanade 방법을 이용하여 block matching 방법과 비교하였다.

2. Optical Flow를 이용한 Motion Estimation

Image sequence s(X,t)의 변화에 근거하여, image좌표계 X의 시간 t에서 t'에서의 displacement를 correspondence vector라 하며, 식(1)로 정의한다.

$$d(X,t)=[d_1(X,t)\ \ d_2(X,t)]^T \qquad (1)$$

따라서, image 시간 t에서의 복원될 image는 식(2)와 같이 표현된다.

$$s(x_1+d_1(X,t),x_2+d_2(X,t),t+\Delta t) = s(x_1,x_2,t) \qquad (2)$$

식(2)는 image에 있어서 조명의 변화가 없이 세기의 변화가 단지 displacement(d_1, d_2)에만 영향을 받는다는 가정을 두고 있다. Optical flow벡터는 image 좌표계에서 임의의 픽셀 (X,t)에서의 순간적인 변화율로 정의한다. Optical flow를 이용한 움직임추정은 주어진 image에서의 속도성분벡터를 구하는 것이다. 따라서, 식(2)를 식(3)과 같이 쓸 수 있는데, 2 프레임에서 속도성분에 변화가 없이 일정하다는 가정에 근거한다. 즉, $d(X,t;\Delta t)=v(X,t)\Delta t$이다.

$$s(x_1+v_1(X,t),x_2+v_2(X,t),t+\Delta t) = s(x_1,x_2,t) \qquad (3)$$

$v_1(X,t), v_2(X,t)$는 각각 $\frac{dx_1}{dt}$, $\frac{dx_2}{dt}$로서 image 좌표계에서의 좌표계 각각의 성분의 속도벡터를 나타낸다. 식(3)을 다시 정리하면, 시간의 변화에 따라 image $s_c(x_1,x_2,t)$의 각각의 픽셀점은 움직임궤적을 따라 일정하게 되는 점을 찾아가는데 식(4)와 같이 된다.

$$\frac{ds_c(x_1,x_2,t)}{dt}=0 \qquad (4)$$

식(4)는 시간의 변화에 따른 image-plane 좌표계의 변화율로서, 움직임 궤적에 따른 세기의 변화율을 나타낸다. 식(4)에 chain rule을 적용하면 식(5)와 같이 표현될 수 있다. 그림 1은 식(5)를 그래프를 이용하여 표현한 것으로 시간t의 변화에 따른 image의 displacement D(d_1, d_2)를 나타내고 있다. 식(3)을 Taylor's expansion을 이용하여 고차항은 버리고, 나머지 초기 몇 개의 항만을 선택하여 식(5)를 유도 할 수도 있다.

$$\frac{\partial s_c(X;t)}{\partial x_1}v_1(X,t)+\frac{\partial s_c(X;t)}{\partial x_2}v_2(X,t)+\frac{\partial s_c(X;t)}{\partial t}=0 \qquad (5)$$

식(5)를 optical flow equation 또는 optical flow constraint라 한다. 식(5)를 gradient operator를 이용하여 다시 쓰면 식(6)과 같이 된다.

$$\nabla s_c(X;t)\cdot v(X,t)+\frac{\partial s_c(X;t)}{\partial t}=0 \qquad (6)$$

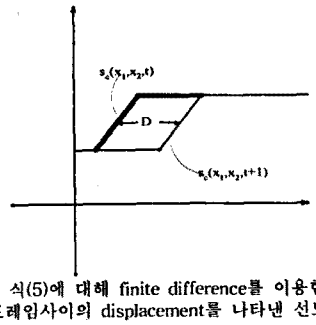


그림 1. 식(5)에 대해 finite difference를 이용한 현재 프레임과 이전 프레임사이의 displacement를 나타낸 선도.

$\nabla s_c(X;t)$는 x_1,x_2에 대한 gradient를 나타낸다. Image-plane에서 식(6)을 이용하면, 임의의 한점에서 움직임으로 인한 새

기의 변화에 따른 하나의 값만을 구할 수 있다. 반면에, 그점에서의 움직임 벡터는 2변수를 가지기 때문에 식(6)만으로는 움직임 벡터를 구할 수 없는데, aperture problem이라 한다. 즉, image-plane에서의 gradient 방향에 있는 움직임만을 알 수 있다. 따라서, 이러한 문제를 해결하기 위한 또다른 조건이 필요하게 되었다. Horn-Schunck는 smoothness 조건을 주어 optical flow gradient 제곱값이 최소가 되는값을 찾는 방식으로, 픽셀와 픽셀사이의 속도벡터의 변화에 대한 조건을 다음 식(7)과 같이 주어서 optical flow를 계산한다..

$$\varepsilon^2{}_s(v(X,t)) = (\frac{\partial v_1}{\partial x_1})^2 + (\frac{\partial v_1}{\partial x_2})^2 + (\frac{\partial v_2}{\partial x_1})^2 + (\frac{\partial v_2}{\partial x_2})^2 \quad (7)$$

따라서, 속도벡터가 더 smooth하면, 에러값은 더 작아지게 된다. 식(7)를 이용하여 optical flow equation은 다음 식(8)를 최소가 되게 함으로서 구할 수 있다.

$$\int\int\{(v\cdot\nabla s_c(X;t))^2 + \alpha^2\cdot\varepsilon^2{}_s(v(X,t))\}dxdy \quad (8)$$

또 다른 방법으로는, 움직임 벡터가 특정한 블럭내에서 일정하다는 것으로 Lucas-Kanade에 의해 제안되었다. 즉, 특정 블럭 B에서 각각의 픽셀점에서의 속도성분은 모두 같은 방향의 속도성분을 가진다는 가정으로서, 속도성분은 다음 식(9)와 같이 표현될 수 있다.

$$v(X,t) = v(t) = [v_1(t)\,v_2(t)]^T, \text{ for } X\in B \quad (9)$$

이 방법은 rotation과 같은 움직임은 해결 할 수 없지만, 단지 translation만 있을 경우에 적용할 수 있는데, 픽셀에서의 세기 성분의 변화가 충분해야 된다. 위 두 방법의 각각의 좌표성분에 대한 gradient 계산은 다음과 같이 2가지 방식을 사용하였다. 첫 번째 방식은 finite differences를 이용한 것으로 중심픽셀 ±1 주위의 픽셀과 다음 프레임에서의 같은 픽셀에 값들의 합을 평균하여 근사화된 값을 사용하였다. 두 번째 방식은 polynomial fitting을 이용한 방식으로 image $s_c(x_1, x_2, t)$를 다음 식(10)과 같이 다항식으로 근사화하여 계산한다.

$$s_c(x_1, x_2, t) \approx \sum_{i=0}^{N-1} a_i \varnothing_i(x_1, x_2, t) \quad (10)$$

a는 각각의 다항식에 대한 계수값, ∅는 기본다항식, N은 다항식 개수를 나타낸다. 위 근사치를 이용한 optical flow계산은 원 픽셀값과 근사화된 값의 차이가 최소가 되는 계수값을 찾아냄으로써 구해지는데, 다음 식(11)과 같이 least square error를 이용하여 계수값 a를 계산한다.

$$e^2 = \sum_{n_1=0}\sum_{n_2=0}\sum_{k=0}\left(s(n_1,n_2,k) - \sum_{i=0}^{N-1} a_i \varnothing_i(x_1,x_2,t)\Big|_{[x_1\,x_2\,t]^T = V[n_1\,n_2\,t]^T}\right) \quad (11)$$

위 식(2-11)을 다시 계수 a에 대해 계산하면 각각의 좌표값에 대한 gradient를 계산 할 수 있다.

3. 모의 실험

실험에서는 시간상의 연속된 2프레임을 이용하여 Lucas-Kanade 방법과 Horn-Schunck 방법에 대해 움직임 벡터를 구하였다. IBM-PC486 DX2-66시스템을 이용하여 프로그램을 수행하였으며, Borland C를 이용하여 코딩하였다. 전처리를 하지 않은 상태의 raw image sequence에 대해 움직임 벡터를 구하였으며, 복원된 영상에 대해서도 프레임사이의 에러 신호에 대한 보상을 하지 않았다. Horn-Schunck방법에서는 α^2값을 100으로 하였으며, Gauss-Seidel 방식을 이용하여 25번 반복하여 각각의 좌표값에 대한 속도성분을 구하였다. Lucas-Kanade 방법에서는 블록을 5×5로하여 계산을 하였다. 또, 각각의 방법에 대한 gradient 계산을 위한 방식으로 finite difference와 polynomial fitting을 이용하였는데, finite difference는 여러 가지 방법이 있지만, 계산상의 편의를 위해 픽셀 ±1주위의 픽셀값들만을 고려하여 계산하였다. Polynomial fitting은 기본다항식의 차수를 2차로 하여 9개의 기본항을 가지며, 픽셀 주위 5×5블록에 대해 윈도우를 씌워 각각의 기본항에 대한 계수값 a를 구하였다. 그리고, 각각의 반복계산을 수행할 때, 속도성분의 초기치는 0으로 하여 계산하였으며, 2번째 반복부터는 주위 8-direction의 평균을 계산하여 현재 좌표값으로 취하였다. 각방법에 대한 성능비교를 위해 PSNR값을 이용하였는데, 구하는 식은 다음과 같다.

$$PSNR = 10\log_{10}\left[\frac{255^2}{\sum_{m=0}^{M}\sum_{n=0}^{N}[U(m,n) - U_E(m,n)]^2}\right] \quad (12)$$

U(m,n)은 원 영상, U_E(m,n)은 추정한 영상을 나타내며, 255는 픽셀값의 최대값을 나타낸다. 결과비교를 위해 사용한 block matching 방법은 full search방식을 사용하였다. 블록비교를 위해서는 다음식과 같이 MAE(mean absolute error)를 이용하였다.

$$MAE = \frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}[U(m,n) - U_R(m+i, n+j)]$$
$$-p \le i, j \le p \quad (13)$$

다음 표는 각방법들에 대해서, ping pong sequence와 mobile and calendar sequence에 대해 PSNR값을 비교한 것이다. 각각의 sequence에 대한 움직임 벡터와 복원된 영상을 나타내었는데, 그림에서 알 수 있는바와 같이 ping pong sequence에 대해서는 estimation을 잘하는 것을 알 수 있다. 그러나, mobile and calendar sequence에 대해서는 estimation을 잘 하지 못하는데, 중심픽셀 주위의 픽셀값들을 전처리 과정을 통하여 방향성에대해 어느정도 정확한 정보를 주어야 될 것같다. 각각의 좌표값에 대한 미분값은 Finite differences가 polynomial fitting에 비해 약 1.5~2dB이 높았다. 그리고, polynomial fitting방식을 이용한 Horn-Schunck방법은 상당히 많은 계산시간이 걸렸는데, 약 22분정도 걸렸다.

표 1. 'ping pong sequence'를 이용하였을 때의 비교

Method		PSNR(dB)
Block matching	Full Search	30.95
Lucas_Kanade	Differences	28.96
	Polynomial fitting	26.18
Horn-Schunck	Differences	29.06
	Polynomial fitting	25.07

표 2. 'mobile and calendar sequence'를 이용하였을때의 비교

Method		PSNR(dB)
Block matching	Full Search	22.80
Lucas-Kanade	Differences	19.87
	Polynomial fitting	17.49
Horn-Schunck	Differences	18.70
	Polynomial fitting	17.66

4. 결 론

그림 2, 3, 4, 5에서 알 수는바와 같이 ping pong sequence에 대해서는 움직임 벡터를 아주 정확하게 잘 추정함을 알 수 있다. 그러나, mobile and calendar sequence에 대해서는 움직임 벡터를 그다지 정확하게 추정하지 않았는데, 상당히 움직임과 픽셀세기의 분포가 복잡하기 때문에 전처리를 한 번 거친후에 각각의 픽셀좌표에대한 gradient를 더 정확하게 구하여야 하겠다. 그리고, image sequence에 대해서 전처리를 하지 않았는데, 움직임 벡터를 추정할 때 Gaussian Kernel를 이용하여 image sequence를 미리 smoothing을 하여 속도성분에 대한 더 정확한 정보를 주어야 한다. 에지부분은 많은 부분이 block matching 방법에 비해 블록화 현상이 줄어듬을 알 수있으며, ping pong sequence에 대해서는 확연히 알 수 있다. Lucas-Kanade 방법에 비해 Horn-Schunck 방법은 화질상의 차이가 거의 없는데 비해, 계산시간은 상당히 오래 걸렸다. 이 실험에서는 표시하지 않았는데, PSNR값의 차이는 Lucas-Kanade 방법과 거의 차이가 없지만 bit-rate는 Horn-Schunck 방법이 좋았다. 실험에서는 이용하지 않았지만 3-step search 방법과 같은 block matching 방법에 비해 optical flow를 이용한 방법은 정확한 움직임을 추정할 수 있지만, 시간이 많이 걸리는 단점이 있다. 그리고, mobile and calendar sequence같은 경우에는 정확한 움직임 벡터를 구할 수 없었는데, 조건식에 대한 개선된 알고리즘이 필요하다. Lucas-Kanade방법에 쓰인 블록을 가변적으로 변화시킴으로서 좀더 개선된 움직임 벡터를 구할 수 있을 것으로 본다.

(a) (b)

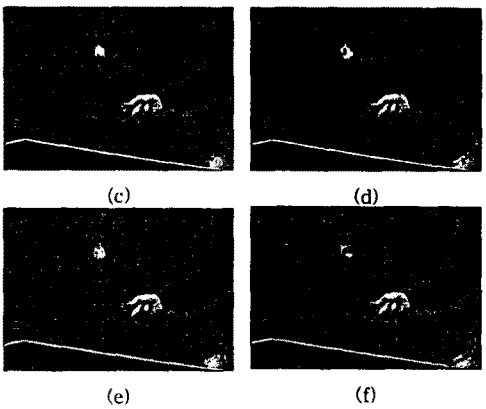


그림 2. 각각의 방식에 대한 복원영상 ; (a):Original image,(b):Fullsearch,(c):Lucas-Kanade(differences), (d):Lucas-Kanade(polynomialfitting),(e):Horn-Schunck(differences),(f):Horn-Schunck(polynomial fitting).

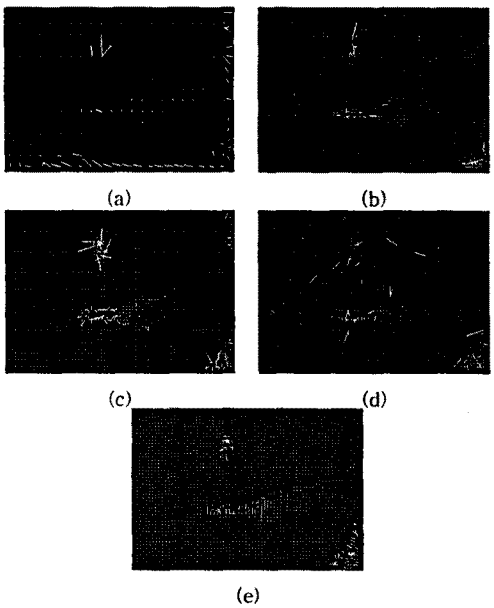
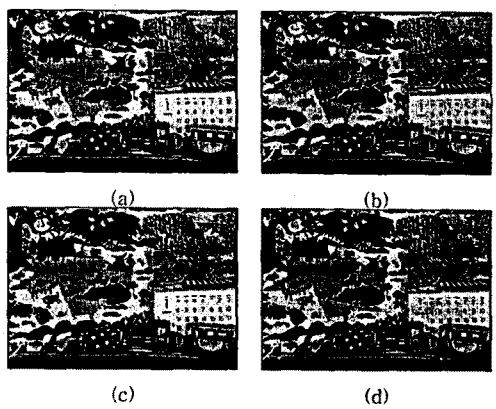


그림 3. 각각의 방식에 대한 motion vector; (a):Fullsearch,(b):Lucas-Kanade(differences), (c):Lucas-Kanade(polynomialfitting),(d):Horn-Schunck(differences), (e):Horn-Schunck(polynomial fitting).



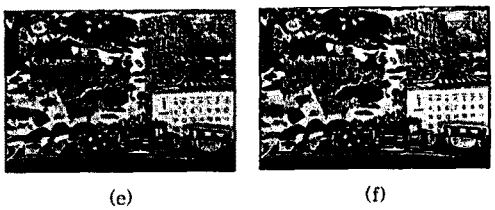


그림 4. 각각의 방식에 대한 복원영상 ; (a):Original image, (b):Fullsearch,(c):Lucas-Kanade(differences), (d):Lucas-Kanade(polynomial fitting),(e):Horn-Schunck(differences),(f):Horn-Schunck(polynomial fitting).

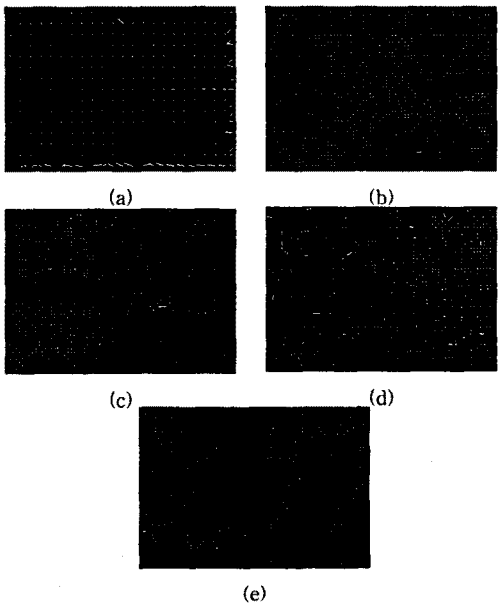


그림 5. 각각의 방식에 대한 motion vector; (a):Fullsearch,(b):Lucas-Kanade(differences), (c):Lucas-Kanade(polynomial fitting), (d):Horn-Schunck(differences), (e):Horn-Schunck(polynomial fitting).

참고 문헌

[1] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," Proc. DARPA Image Understanding Workshop, pp. 121-130, 1981.

[2] B. K. P. Horn and B. G. Schunck, "Determining optical flow," Artif. Intell., vol. 17, pp. 185-203, 1981.

[3] J. K. Aggarwal and N. Nandhakumar, "On the Computation of Motion from Sequences of Imgaes-A Review," Proc. IEEE, vol. 76, pp. 917-935, Aug. 1988.

[4] 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.

[5] A. murat tekalp, "Digital video processing," Prentice Hall Signal Processing Series, 1995.

[6] A.K. Jain, "Fundamentals of digital image processing," Prentice Hall, 1989.