Journal of Korea Multimedia Society Vol. 10, No. 11, November 2007(pp. 1407-1416)

# 피라미드 기반 광류 추정을 이용한 영상 내의 임의의 점 추적 알고리즘

이재광[†], 박창준[††]


## 요 약

본 논문에서는 피라미드 기반 광류 추정 방법과 이를 이용하여 영상 내 임의의 점 추적 알고리즘에 대해 설명한다. 본 논문에서는 Lucas-Kanade 광류 추정 방법을 기반으로 광류를 계산하였다. 작은 움직임에 민감하면서 큰 움직임까지 계산할 수 있도록 영상 피라미드를 사용하였고, 영상 피라미드와 Lucas-Kanade 광류 추정 방법을 혼합하는 과정에서 영상 피라미드의 하위수준으로 내려갈수록 증폭되는 광류 추정 오차를 줄이기 위한 정제방법을 제안하였다. 또한 광류의 제약조건과 부화소 보간을 이용하여 광류 추정의 정확도를 높였으며, 추적하고자 하는 영상내의 임의의 점 주변의 광류 값을 이용하여 테두리나 모퉁이 같은 특징이 없는 점들의 추적도 가능하도록 하였다. 본 논문에서는 웹 카메라를 이용하여 제안된 알고리즘의 광류 계산 결과와 임의의 점 추적 결과를 제시한다.


# Algorithm for Arbitrary Point Tracking using Pyramidal Optical Flow

Jae-kwang Lee[†], Chang-Joon Park[††]


## ABSTRACT

This paper describes an algorithm for arbitrary point tracking using pyramidal optical flow. The optical flow is calculated based on the Lucas-Kanade's optical flow estimation in this paper. The image pyramid is employed to calculate a big motion while being sensitive to a small motion. Furthermore, a rectification process is proposed to reduce the error which is increased as it goes down to the lower level of the image pyramid. The accuracy of the optical flow estimation was increased by using some constraints and sub-pixel interpolation of the optical flow and this makes our algorithm to track points in which they do not have features such as edges or corners. The proposed algorithm is implemented and primary results are shown in this paper.


**Key words:** Image Pyramid(영상 피라미드), Optical Flow(광류), Point Tracking(점 추적), Lucas-Kanade(Lucas-Kanade)

## 1. 서 론

컴퓨터 비전의 궁극적인 목표는 컴퓨터로 하여금 주어진 영상을 이해하도록 하는 것이다. 컴퓨터 비전에서 하나의 영상만으로도 많은 정보를 얻을 수 있지만 시간적으로 연속된 여러 영상들을 함께 고려한다면 하나의 영상에서 얻지 못하는 유용한 정보를 얻을 수 있다.

※ 교신저자(Corresponding Author) : 이재광, 주소 : 대전광역시 유성구 가정동 161번지(305-700), 전화 : 042)860-5345, FAX : 042)860-1050, E-mail : ljk64386@etri.re.kr
접수일 : 2007년 3월 29일, 완료일 : 2007년 9월 18일
[†] 준회원, 과학기술연합대학원대학교 컴퓨터 소프트웨어 및 공학과

[††] 정회원, 한국전자통신연구원 영상콘텐츠 연구그룹
(E-mail : chjpark@etri.re.kr)
※ 본 논문은 정보통신부 및 정보통신연구진흥원의 IT신성장동력핵심기술개발사업의 일환으로 수행하였음.[2006-S-044-02, 멀티코아 CPU 및 MPU 기반 크로스플랫폼 게임 기술]

카메라를 이용하여 시간적으로 연속된 영상을 얻을 때, 물체에 대해서 카메라의 위치를 이동시키거나 또는 카메라에 대해서 물체의 위치를 이동시키면 영상 내의 각 화소들의 밝기의 변화가 나타난다. 이러한 영상내의 명암의 이동을 광류(Optical Flow)라고 한다. 광류 정보를 이용하여 명암의 이동을 분석하여 물체의 3차원 구조를 추론할 수 있다. 이것은 우리가 2차원 영상의 연속적인 변화를 나타내는 영화를 통하여 3차원 구조를 추론하는 원리와 일치하며, 이와 같은 원리는 동역학 효과(Kinetic Effect)라 불리운다[1].

광류 추정 방법은 시공간 기울기(Spatiotemporal Gradient), 영역 기반(Region-based), 특징 기반(Feature-based) 방법 등으로 분류될 수 있다. 시공간 기울기법으로는 Horn과 Schunck[2], Negel과 Enkelmann[3], Subbarao[4] 등의 방법이 있다. 시공간 기울기법에 의하여 구해지는 광류는 기본적으로 영상의 밝기 값의 변화는 움직임에 의해서만 결정된다는 밝기 값 제한조건에 바탕을 두고 있다. 다시 말하면 화소의 밝기 값이 일정하다는 가정 하에 다음 프레임에서 같은 밝기 값을 갖는 화소를 찾는 방법이다. 광류를 안정적으로 구하기 위해서는 일반적으로 제약조건을 추가시키는 형태를 취하고 있어, 이 방법의 대부분이 편미분 방정식의 형태를 가지기 때문에 반복적인 해를 얻게 된다.

영역 기반 광류 추정 방법으로 Fuh와 Maragos[5]의 방법이 대표적이다. 이 방법은 오차를 최소화 하는 움직임을 찾는 방법으로 실제 물체의 움직임과는 다른 움직임을 찾는 경우가 생긴다. 특히 밝기 값이 일정한 부분과 직선 형태의 에지 부분에서는 좋은 결과를 주지 못한다. 최근에 P.-C. Chung et. al[6]에 의해 영역 기반으로 구해진 광류를 기울기 방향의 움직임 벡터의 평균 자승 오차를 최소화 하도록 역투영 시켜 광류를 추정하는 기법이 나오기도 하였다. 특징 기반 방법으로 Wang et al.[7]이 있는데 이러한 방법의 경우 특징으로 주어진 점에서는 정확한 움직임을 구할 수 있으나 모든 점에서의 움직임을 구하기 위해서는 보간이 필요하고, 특징점이 많이 나타나지 않거나 움직임이 불연속인 경우 보간이 어렵다. 이 방법은 카메라가 움직이는 경우에는, 영상내의 모든 움직임의 연속성이 크기 때문에 물체만 움직이는 경우보다 좋은 결과를 얻을 수 있다.

영역에 기반을 둔 방법은 실제의 움직임과는 다른 움직임을 찾는 단점이 있고, 특징에 의한 방법은 특징점에서는 정확한 움직임을 구할 수 있으나 모든 영역에서 움직임을 구할 수 없는 단점을 가진 반면, 시공간 기울기법은 특징점과 같이 영상 불연속에서는 움직임 추정이 평활화되나 실제의 움직임을 구하기에 적합하며, 영상전체에서 움직임을 구할 수 있어 영상해석에 중요하게 사용되는 방법이다.

본 논문에서는 영상 내의 임의의 점을 추적하기 위해서 기본적으로 영상 전체의 움직임을 구할 수 있는 시공간 기울기법을 사용하였다. 시공간 기울기법은 움직임 제한식과 연속성 제한식을 미분에 의해 구하므로 잡음에 민감한 특성이 있지만 본 논문에서는 시공간 기울기법을 최소자승법을 사용하여 고차항을 제거함으로서 잡음에 민감하지 않도록 광류 식을 재정의 하였으며, 작은 움직임에 민감하면서 큰 광류를 계산할 수 있도록 영상 피라미드를 사용하였다. 영상 피라미드 계산 시 증폭되는 광류의 오차를 줄이기 위해 피라미드의 각 수준에서 다음 수준의 광류를 계산할 때 다음 수준의 광류 값을 예측함으로서 보다 정확한 광류를 구할 수 있는 알고리즘을 제안하였다. 또한 광류의 제약 조건을 이용하여 기울기 기반 방법의 취약점을 개선시켰으며, 임의의 점을 추적하기 위해 주변 광류 값을 이용하여 양선형 보간을 사용하여 점 추적을 수행하였다.

본 논문의 구성은 2장에서 최소 자승법 기반 광류 추정 방법과 영상 피라미드 개념에 대해 설명을 하고 3장에서는 Lucas-Kanade 방법을 기반으로 한 영상 피라미드 정제 광류 방법에 대해 설명을 한다. 4장에서는 계산된 광류 결과를 제약조건과 보간법을 사용하여 임의의 점 추적에 관한 알고리즘에 대해 논의를 한 후 5, 6 장에서는 실험결과, 결론 및 향후 과제를 제시하였다.

## 2. 광류 추정(Optical Flow Estimation)

### 2.1 광류(Optical Flow)

광류를 계산하는 목적은 운동계를 찾기 위한 것인데 운동계란 3차원 물체를 2차원에 투영한 영상의 각 화소에 운동속도를 부여한 것이다[1]. 실세계인 3차원의 운동계는 시간에 따른 밝기 분포만으로는 계산 될 수 없다. 하지만 물체가 카메라에 대하여 상

대적으로 움직일 때에 관찰되는 밝기 패턴의 이동인 광류는 계산 될 수 있다.

Horn과 Schunck[2]는 광류를 수학적으로 정의하였다. 먼저 물체의 어떤 점에 대응하는 영상의 밝기 $E$는 물체가 이동하더라도 일정하다고 가정하였다. $E(x(t),y(t),t)$가 시간 $t$에서의 위치 $(x,y)$의 밝기를 나타낸다면,

$$\frac{dE(x(t),y(t),t)}{dt}=\frac{\partial E}{\partial x}\frac{dx}{dt}+\frac{\partial E}{\partial y}\frac{dy}{dt}+\frac{\partial E}{\partial t}=0 \qquad (1)$$

식 (1)을 다시 쓰면,

$$(\nabla E)^T v+E_t=0 \qquad (2)$$

와 같이 표현할 수 있다. 여기서, 영상에서의 공간 기울기 $\nabla E=\begin{bmatrix}\frac{\partial E}{\partial x}\\ \frac{\partial E}{\partial y}\end{bmatrix}$, 광류 $v=\begin{bmatrix}\frac{dx}{dt}\\ \frac{dy}{dt}\end{bmatrix}$, 시간에 따른 명도 변화량 $E_t=\frac{\partial E}{\partial t}$ 이다.

본 논문에서 사용된 시공간 기울기 방법은 인접한 영상에서 테두리와 같은 공간적 특징에 관련된 시간적 변화를 구해서 광류를 계산하는 방법으로서 식 (2)를 여러 점에 대하여 하나의 해를 구하는 방법이라 할 수 있겠다. 즉, 식 (2)는 $t^*E_x+u^*E_y+E_t=0$으로 쓸 수 있으며, 이 식의 해인 $(t,u)$는 점 $(E_x,E_y),E_t$를 지나는 직선의 방정식을 나타낸다고 볼 수 있다.

광류가 작은 탐색 영역 내에서 일정한 벡터 $v$로 표현될 수 있다고 가정하자. 작은 5x5 탐색 영역을 $Q$라 하고, 각 화소의 점을 $p_i$라고 하면, 식 (2)는 다음과 같이 표현할 수 있다.

$$(\nabla E(p_i))^T v+E_t(p_i)=0,\ \ i=1,\cdots,N^2 \qquad (3)$$

식 (3)을 최소자승법을 사용하여 해를 구하면 다음과 같다.

$$v=(A^TA)^{-1}A^Tb \qquad (4)$$

여기서, $A=\begin{bmatrix}E_x(p_1) & E_y(p_1)\\ E_x(p_2) & E_y(p_2)\\ \vdots & \vdots\\ E_x(p_{N^2}) & E_y(p_{N^2})\end{bmatrix}$, $b=\begin{bmatrix}-E_t(p_1)\\ -E_t(p_2)\\ \vdots\\ -E_t(p_{N^2})\end{bmatrix}$,

$v=\begin{bmatrix}v_x\\ v_y\end{bmatrix}$ 이다.

최소자승법은 영상 밝기의 편미분항의 고차항들을 필요로 하지 않기 때문에 반복적인 해를 구하지 않으며, 고차항이 없으므로 잡음에 민감하지 않다.

## 2.2 영상 피라미드(Image Pyramid)

영상 시퀀스에서 연속된 두 영상 $I,J$를 가정하면 광류는 영상 $I$의 한 점 $p$와 동일한 영상 값을 갖는 두 번째 영상 $J$에서의 동일한 점 $q$를 계산하는 것이다. 수식적으로는 $I(p)=J(q)$를 만족하는 $q=p+d$에서 이동 벡터 $d$를 구하는 것으로 표현할 수 있다. 시간 $t$에서의 화소 $p$와 시간 $t+1$에서의 $q$ 주위의 작은 탐색 영역을 고려하고, 화소 $p$는 $q$ 주위의 탐색 영역을 벗어날 수 없다고 가정한다. 그렇다면 화소의 움직임은 이 영역내의 영상 값들의 차이를 최소화하는 이동 벡터 $d$를 구하는 것으로 나타낼 수 있다. 이 관계를 수식적으로 나타내면,

$$\epsilon(d)=\sum_{x\in N(p)}[I(x)-J(x-d)]^2 \qquad (5)$$

와 같이 오차 함수로 나타낼 수 있으며 이 함수를 최소화하는 $d$를 구하면 된다. 식 (5)에서 점 $p$ 주위의 탐색 영역 크기를 $(2w_x+1)\times(2w_y+1)$이라고 하면 점 $p$의 이웃 점들은 다음과 같이 나타낼 수 있다.

$$N(p)=\left\{\begin{matrix}(x,y)|(x,y)\in[P_x-w_x,p_x+w_x]\times\\ [p_y-w_y,p_y+w_y]\end{matrix}\right\} \qquad (6)$$

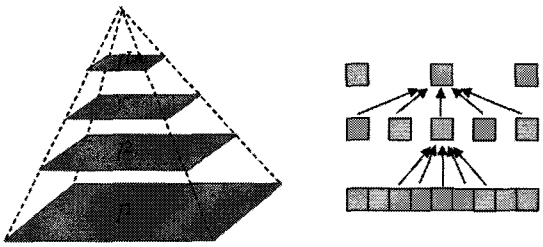
여기서, 탐색 영역 크기 $w$는 보통 2~7까지의 값이 사용된다.

광류를 계산할 때 계산의 정확도와 영상 특성에의 강건함을 고려해야 한다. 계산의 정확도란 계산된 움직임이 얼마나 정확한가를 나타낸다. 이는 첫 번째 영상과 두 번째 영상 사이에서 움직임을 찾는 영역이 작을 경우 영상 값의 작은 변화에도 민감하게 되며, 큰 영역보다 가려진 영역에 보다 적합하다. 하지만 작은 탐색 영역을 사용할 경우 움직임이 클 때에는 계산이 불가능해진다. 또한 영상 특성에의 강건함이란 알고리즘이 주어진 영상에서 움직임의 크기나 빛의 변화에 민감한지 등에 관련된 것으로서 큰 탐색 영역을 사용할 경우 큰 움직임을 검출할 수 있다. 따라서 움직임 검출에서의 영역 크기를 결정하는 것은 매우 중요한 문제이고, 두 요소들 사이의 상호 상충 조건이 생긴다고 할 수 있다. 본 논문에서는 이를 극

복하기 위해 영상 피라미드를 사용하였다.

영상 피라미드란 작은 탐색 영역을 사용하면서도 영상 값의 열화를 방지하고, 큰 움직임을 구할 수 있는 방법이다.

그림 1(a)에서 $\{I^L\}_{L=1,\cdots,L_m}$을 영상 피라미드 집합이라고 하고 $L$은 피라미드 높이(수준)라고 하자. $I^1$은 피라미드 영상에서 최하위 수준에 위치한 영상이고, 해상도가 가장 좋은 원 영상이 이에 해당한다. 피라미드 영상은 원 영상에 해당하는 최하위 수준 영상으로부터 부 표본화를 통해 영상의 크기를 두 배로 줄임으로서 반복적으로 만들어진다. 그림 1(b)는 화소 수준에서의 부 표본화를 예로 나타낸 것이다.



(a) 영상 피라미드 (b) 부 표본화

그림 1. 영상 피라미드와 부 표본화의 예

## 3. Lucas-Kanade 피라미드 광류 추정

Lucas-Kanade[8]는 반복적인 방법으로 광류를 계산하였다. 영상에서 임의의 점을 $p=(p_x,p_y)$라 하고 탐색 영역(Searching Window) 크기를 $(w_x,w_y)$라고 할 경우 SSD 오차 함수는

$$\epsilon(v_x,v_y)=\sum_{x=p_x-w_x}^{p_x+w_x}\sum_{y=p_y-w_y}^{p_y+w_y}(I(x,y)-J(x+v_x,y+v_y))^2 \tag{7}$$

으로 쓸 수 있다. 여기서 $(v_x,v_y)$는 광류를 나타내는 벡터이다.

광류 벡터를 구하기 위해서는 식 (7)의 오차함수가 최소가 되는 값을 구해야하며, 이는 식 (7)을 편미분해서 0이 되는 값을 찾으면 된다. 이를 식으로 나타내면

$$\frac{\partial\epsilon(v)}{\partial v}\Big|_{v=v^*}=-2\sum_{x\in N(p)}\frac{\partial J(x+v)}{\partial v}(I(x)-J(x+v))\Big|_{v=v^*}=0 \tag{8}$$

와 같고 $v$가 작을 경우에는 테일러 확장에 의해 근사화 시켜 다음과 같이 구할 수 있다.

$$v^*=(\sum_{x\in N(p)}\frac{\partial J(x)}{\partial x}\frac{\partial J(x)}{\partial x}^T)^{-1}(\sum_{x\in N(p)}\frac{\partial J(x)}{\partial x}(I(x)-J(x))) \tag{9}$$

하지만 식(9)는 움직임이 매우 작은 값일 때 정확한 해에 대한 근사치로서 구해질 수 있다. 움직임이 작지 않은 경우는 다음 그림 2에서 도시한 것과 식

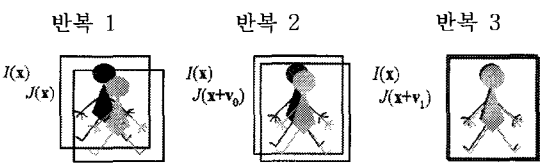



그림 2 반복적인 해

(10)~(12)와 같이 반복적인 계산을 통해 해를 구할 수 있다.

$$v_0\leftarrow(\sum_{x\in N(p)}\frac{\partial J(x)}{\partial x}\frac{\partial J(x)}{\partial x}^T)^{-1}(\sum_{x\in N(p)}\frac{\partial J(x)}{\partial x}(I(x)-J(x))) \tag{10}$$

$$v_1\leftarrow v_0+(\sum_{x\in N(p)}\frac{\partial J(x+v_0)}{\partial x}\frac{\partial J(x+v_0)}{\partial x}^T)^{-1}(\sum_{x\in N(p)}\frac{\partial J(x+v_0)}{\partial x}(I(x)-J(x+v_0))) \tag{11}$$

$$v_2\leftarrow v_1+(\sum_{x\in N(p)}\frac{\partial J(x+v_1)}{\partial x}\frac{\partial J(x+v_1)}{\partial x}^T)^{-1}(\sum_{x\in N(p)}\frac{\partial J(x+v_1)}{\partial x}(I(x)-J(x+v_1))) \tag{12}$$

식 (10)과 같이 첫 번째 계산에서 두 신호의 위치 차인 $v_0$를 구한다. 그 후 식 (11), 식 (12)와 같이 반복 계산을 하면서 $J$를 $v_0$만큼, $v_1$만큼 이동시킨 후 두 신호의 위치 차이를 구해 $v_0$을 더한 $v_1$을, $v_1$을 더한 $v_2$를 반복적으로 구하면 $v$가 큰 경우에도 해를 구할 수 있게 된다. 이렇게 반복적으로 해를 구하면 구하고자 하는 광류 $v$는 $q=p+d$를 만족하는 이동 벡터 $d$가 된다.

연속된 두 영상간의 한 점 $x$에서의 시간적인 차이와 공간적인 기울기는 다음 식 (13), 식 (14)와 같이

계산할 수 있다.

$$\delta I(x) = I(x) - J(x) \tag{13}$$

$$\nabla J(x) = \frac{\partial J(x)}{\partial x} \tag{14}$$

이며, $\nabla J(x,y) = \begin{bmatrix} J_x(x,y) \\ J_y(x,y) \end{bmatrix} = \begin{bmatrix} \dfrac{\partial J(x,y)}{\partial x} \\ \dfrac{\partial J(x,y)}{\partial y} \end{bmatrix}$ 이다. 공간적인 기울기 행렬 $G$와 영상 불일치 벡터 $b$의 변수를 사용하여 다음 식과 같이 간략화 시킬 수 있다. 움직임이 작은 경우의 해 $v^*$는 식 (15)와 같이 표현이 가능하며, 반복적인 해 $v_k$는 식 (16)과 같이 표현될 수 있다.

$$v^* = G^{-1} b \tag{15}$$

$$v_{k+1} \leftarrow v_k + G_k^{-1} b_k \tag{16}$$

여기서,

$$G = \sum_{x=p_x-w_x}^{p_x+w_x} \sum_{y=p_y-w_y}^{p_y+w_y} \begin{bmatrix} J_x^2(x,y) & J_x(x,y)J_y(x,y) \\ J_x(x,y)J_y(x,y) & J_y^2(x,y) \end{bmatrix},$$

$$b = \sum_{x=p_x-w_x}^{p_x+w_x} \sum_{y=p_y-w_y}^{p_y+w_y} \begin{bmatrix} J_x(x,y)\delta I(x,y) \\ J_y(x,y)\delta I(x,y) \end{bmatrix} \text{ 이며,}$$

$$G_k = \sum_{x \in N(p)} \nabla J_k(x+v_k) \nabla J_k(x+v_k)^T,$$

$$b_k = \sum_{x \in N(p)} \nabla J(x+v_k)(I(x) - J(x+v_k)) \text{ 이다.}$$

식 (13)에서 한 점 $x$에서의 시간적 차이는 시간에 따른 영상의 차이라고 할 수 있다. 이는 연속된 두 영상에 대한 뺄셈 연산을 통해 간단히 계산될 수 있으며, 식 (14)의 공간적인 기울기는 식 (17)과 같이 중간값 차이 연산자를 사용하여 간단히 계산할 수 있다.

$$\nabla J(x) = \begin{bmatrix} J_x(x,y) \\ J_y(x,y) \end{bmatrix} \Leftrightarrow \begin{cases} J * \frac{1}{2}[-1\ 0\ 1] \\ J * \frac{1}{2}[-1\ 0\ 1]^T \end{cases} \tag{17}$$

실제로 중간 값 차이 연산자를 통해 계산한 결과는 실제적으로 Sharr, Prewitt 또는 Sobel 연산자를 사용하는 것보다 좋은 결과를 나타낸다. 식 (15), 식 (16)에서 $v$는 $G$의 역행렬이 존재할 때 그 값이 존재하게 되며, 이는 영상이 $x, y$방향으로 기울기 정보를 포함해야 한다는 것을 의미한다.

A. Bruhn et. al.[9]은 Lucas-Kanade 방법과 Horn-Schunck 방법을 혼합하여 피라미드 기반으로 광류를 추정하였다. 각각의 방법에서 제시하는 비용함수들을 만족시키는 새로운 비용함수를 만들어 최소화 시키는 방법으로 광류 계산을 하였다. 또한 F. Lauze et. al.[10]도 피라미드를 사용하여 정제를 통해 광류를 추정하였지만 실험 영상들은 광류가 쉽게 뽑힐 수 있는 구조물을 포함하고 있어 본 논문에서는 Lucas-Kanade 방법에 피라미드를 적용시키면서 이에 알맞은 정제방법을 제시하였고 비교적 정확하고 깔끔한 광류를 계산할 수 있었다.

피라미드 영상은 원 영상에 해당하는 최하위 수준 영상으로부터 부 표본화를 통해 반복적으로 만들어진다. 최하위 첫 번째 수준의 영상은 원영상이고 두 번째 수준의 영상은 원 영상에서 가우시안 함수를 사용하여 사이즈를 1/2로 줄인 영상이다. 본 논문에서는 부 표본화에 따른 위신호(Aliasing)를 없애기 위해 3×3 크기의 저역 통과 필터를 사용했다.

앞서 설명한 반복적인 움직임 검출 방법에 영상 피라미드를 적용하여 구현하게 되면 광류의 계산속도 향상과 작은 움직임에 민감하면서도 큰 움직임 계산이 가능하지만 그림 3에서 도식화 한 것과 같이 피라미드의 아래 수준으로 내려갈수록 광류의 추정 오차가 두 배로 증폭되는 단점이 있다. 증폭되는 추정 오차를 줄이기 위해 본 논문에서는 영상 피라미드와 반복적인 Lucas-Kanade 광류 추정 방법을 혼합하는 방법에 있어서 영상 피라미드의 최상위 영상으로부터 차례대로 움직임을 구해 나가되, 좀 더 정확하고 세밀한 값으로 구해 나가는 방법을 사용하였다.

그림 4(a)는 광류 추정 알고리즘 전체 흐름도를 나타낸다. 먼저 두 연속된 영상이 입력되면 각 영상에서 영상 피라미드를 생성한다. 광류는 제일 작은 영상인 최상위 수준에서 계산이 시작된다. 반복적인 계산을 위해 광류 벡터의 초기화가 필요하다. 최상위 수준에서의 광류 벡터는 0으로 초기화한다. 피라미드의 최상위 수준 영상에서 두 연속된 영상에서 광류는 그림 4(b)에서와 같이 식 (13)부터 식 (16)의 과정

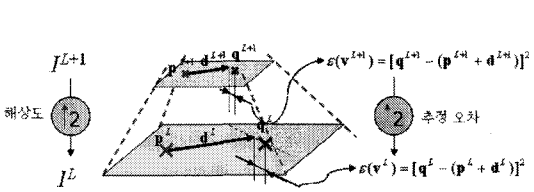
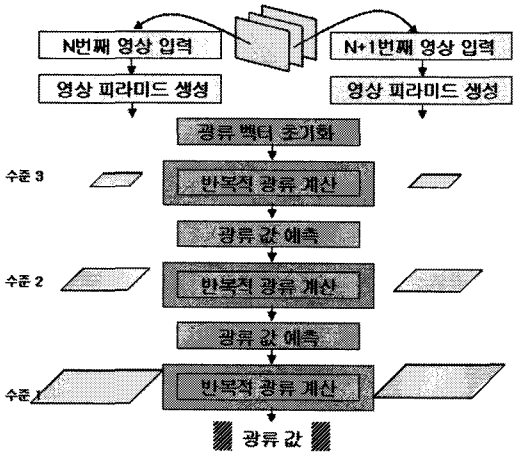
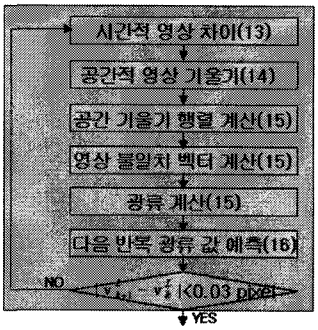



그림 3. 영상 피라미드에서의 광류 추정 오차




(a) 광류 추정 알고리즘 흐름도




(b) 반복적 광류 계산 알고리즘

그림 4. 정제된 광류 추정 알고리즘 흐름도

을 거쳐 계산하게 된다. 실제 광류에 가까운 광류 값 계산을 위해 반복을 하게 되는데 다음 반복을 위한 광류 값을 예측하고 현재 구해진 광류 값과 광류의 초기 값과의 차이를 계산하여 0.03 화소 이상인 경우 0.03 화소 이하가 될 때까지 반복 계산하게 되며, 0.03 화소 이하인 경우 다음 하위 수준에서의 광류 초기 값을 위해 광류 값을 식 (18)과 같이 계산하게 된다. 만약 현재 수준($L$)에서의 광류 초기 값을 $d^L$이라 하고 구해진 광류 값을 $v_k^L$이라고 하면, 다음 하위 수준에서의 광류 초기 값은

$$d^{L-1} = 2(d^L + v_k^L) \tag{18}$$

로 예측하게 된다. 이렇게 광류 값을 예측함으로서 광류 벡터를 보다 세밀하게 정제시킨다. 이러한 과정을 최하위 수준의 영상까지 수행하면 추정오차가 적으면서도 큰 광류를 얻을 수 있다. 최종 광류 값은 $d$ 가 아닌 마지막 수준에서 계산된 $v$가 되며, $d$의 값은 큰 반면 $v$는 매우 작기 때문에 광류는 작은 탐색 영역을 사용하더라도 큰 광류를 검출할 수 있다. 결과적으로 본 논문에서는 $L_m = 3$의 피라미드 영상을 사용하였고, 우리가 정한 최대 이동 벡터 $d_{\max}$의 15배에 해당하는 광류를 검출할 수 있게 된다.

## 4. 영상 내 임의의 점 추적

식(15), (16)에서 두 프레임 사이의 움직임을 추정할 경우 $x$는 정수 값이 된다. 하지만 연속된 여러 장의 프레임들 사이에서 임의의 점 추적을 수행할 경우에는 $x$가 정수 값이 아니다. 즉, $x, v_k$가 정수 값이 아니므로 보다 정확한 해를 구하기 위해서 부 화소 단위의 계산이 필요하며 본 논문에서는 양선형 보간 방법[11]을 이용하였다.

양선형 보간은 1차 보간 방법이고 2×2 이웃점들 사이에서 선형적으로 보간 한다. 보간 방법에는 여러 가지가 있지만, 계산량과 성능 면에서 광류 추정 시 가장 적당한 보간 방법이다. 양선형 보간의 목적은 $x, y$가 정수형이 아닌 경우에 영상 값 $I(x,y)$을 구하는 것이다. 좌표 $x, y$를 $x = x_0 + \alpha_x, y = y_0 + \alpha_y$로 각각 정수 값과 나머지로 분리해서 나타내면 그림 4(a)와 같이 $I(x_0,y)$는 $I(x_0,y_0), I(x_0,y_0+1)$로부터 선형적으로 구할 수 있고 $I(x_0+1,y_0)$는 $I(x_0+1,y_0), I(x_0+1,y_0+1)$로부터 선형적으로 계산이 가능하며 이를 이용하여 $I(x,y)$값도 다음 식과 같이 계산할 수 있다.

$$I(x_0,y) = (1-\alpha_y)I(x_0,y_0) + \alpha_y I(x_0,y_0+1) \tag{19}$$

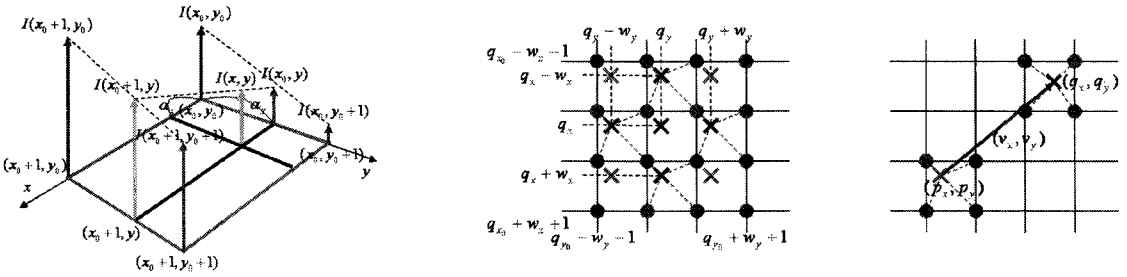
$$I(x_0+1,y) = (1-\alpha_y)I(x_0+1,y_0) + \alpha_y I(x_0+1,y_0+1) \tag{20}$$

$$I(x,y) = (1-\alpha_x)I(x_0,y) + \alpha_x I(x_0+1,y) \tag{21}$$

위의 세 수식을 연립하면 $I(x,y)$는 다음과 같이 구할 수 있다.

$$\begin{aligned} I(x,y) = {} & (1-\alpha_x)(1-\alpha_y)I(x_0,y_0) + (1-\alpha_x)\alpha_y I(x_0,y_0+1) \\ & + \alpha_x(1-\alpha_y)I(x_0+1,y_0) + \alpha_x\alpha_y I(x_0+1,y_0+1) \end{aligned} \tag{22}$$

영상 $I$에서의 점 $I(p)$와 영상 $J$에서의 점 $J(q)$가 대응되는 광류라고 가정할 경우 공간적인 기울기는



(a) 양선형 보간 방법 (b) 공간적인 영상 기울기 (c) 시간적인 영상 차이

그림 5. 광류 추정에서의 양선형 보간 방법

$$\nabla J(x+v_k)=\frac{\partial J(x+v_k)}{\partial x}\Rightarrow\nabla J(q)=\frac{\partial J(q)}{\partial x} \tag{23}$$

라고 쓸 수 있으며 중간 값 차이 연산자를 이용하여 구했다면, 3×3 이웃 점들은 그림 5(b)처럼

$$N(p)\in\{(x,y)|(x,y)\in[q_x-w_x,q_x+w_x]\times[q_y-w_y,q_y+w_y]\} \tag{24}$$

에서 계산을 하게 된다. 또한 시간적인 영상 차이는 그림 5(c)와 같이 표현을 할 수 있고, 이를 식으로 나타내면 다음과 같다.

$$\delta I_k(x)=I(x)-J(x+v_k)\Rightarrow\delta I(p)=I(p)-J(p+v) \tag{25}$$

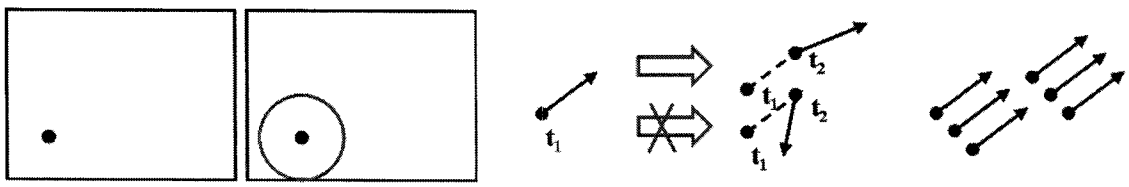
여기서 $p$와 $v$는 정수 값이 아니다.

보통 광류 계산 시 계산량을 줄이기 위해 입력 영상 내의 모든 화소에 대해 계산하지 않고 일정한 간격을 가지는 점들에서만 계산을 한다. 기존의 특징점 기반 추적 알고리즘들은 그 점이 사라지거나 다른 물체에 의해 가려졌을 경우 추적이 불가능하다. 하지만 광류의 특징 중 하나는 그림 6과 같이 광류의 제약조건을 이용할 수 있다는 것이다.

광류의 제약 조건은 첫 번째 영상과 두 번째 영상의 시간차가 적다는 가정 하에 성립이 되며, 그림 6(a)를 이용하여 광류의 탐색 영역의 크기를 결정하는데 도움이 되는 제약 조건으로서 첫 번째 영상과 두 번째 영상의 시간차가 적다면 화소는 아무리 빨리 움직여도 어느 정도의 이동범위를 벗어날 수 없다는 것이며, 그림 6(b)는 $t_2$에서의 광류의 방향이 $t_1$에서의 광류 방향과 급격한 변화가 없다는 것을 나타낸다. 또한 그림 6(c)는 주변의 광류는 같은 움직임을 갖는다는 것이다. 본 논문에서는 기본적으로 그림 6(a), (b), (c)의 제약조건을 이용하여 임의의 점에 대한 추적을 주변에서 계산된 광류 값을 이용하여 구하였다.

## 5. 실험 결과

본 실험에서는 Yosemite 시퀀스 영상을 가지고 Lucas-Kanade 방법과 본 논문에서 제안하는 정제 방법과 제약조건, 부 화소 보간을 사용한 피라미드 기반 광류 추정 방법을 Matlab을 이용하여 비교하였으며, 또한 일반영상에서 웹 카메라에서 캡쳐되는



(a) 광류의 최대 이동 범위 (b) 이동 방향의 작은 변화 (c) 공통 움직임

그림 6. 광류의 제약 조건

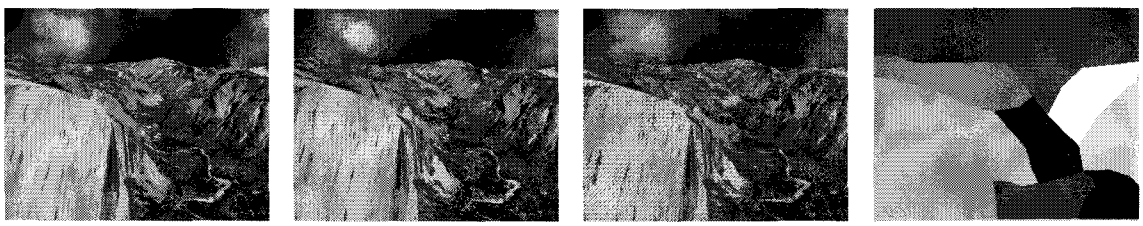
실시간 영상을 가지고 Visual C++를 이용하여 광류 추정 및 임의의 점 추적 결과를 나타내었다. 본 실험은 듀얼 CPU 3.2GHz, 2GB RAM PC에서 작동하였고, 카메라는 로지텍 사의 퀵 캠 프로 4000을 사용하였다.

그림 7(a)는 Yosemite 시퀀스 영상 원본이며, 이때 광류는 5화소 간격의 점에서 계산하였다. 그림 7(b)는 Lucas-Kanade 광류 추정 결과이다. Lucas-Kanade는 미분을 사용하여 계산하였기 때문에 잡음에 매우 민감한 광류 결과를 볼 수 있다. 반면 그림 7(c)에서 보는 바와 같이 제안된 광류 추정 결과는 최소자승법을 사용하였기 때문에 잡음에 보다 강한 결과를 나타내었으며, 특히 그림 6의 제약 조건을 추가하여 그림 7 (d)와 같이 영상 분할까지 가능한 깔끔한 광류 값을 얻을 수 있었다.

그림 8은 광류 제약 조건을 추가하기 전과 후의 실험 결과이다. 그림 8(a)는 제약조건 추가 전 움직이는 물체가 있을 경우의 실험 결과이다. 광류의 최대 이동범위를 지정하지 않았기 때문에 크게 튀는 모습이 보인다. 그림 8(b)는 카메라가 움직이는 경우이다. 중간 중간에 광류가 튀는 모습이 보인다. 그림 8(c)의 경우는 그림 6에서 설명한 광류의 제약조건을 추가한 실험 결과이다. 광류의 최대 이동범위 제약 조건으로 인해 크게 튀는 광류 값은 제거가 되었으며, 주변 광류 값과 공통의 움직임을 갖는다는 제약 조건으로 인해 크게 튀는 경우나 주변과 크게 움직임이 다른 광류를 걸러내어 깔끔한 광류 결과가 나왔다. 그림 8(d)의 경우는 움직이는 물체가 없이 카메라만 움직이는 경우이다. 이 또한 광류 값이 제약조건으로 인해 깔끔하게 나타나는 모습을 볼 수 있다.

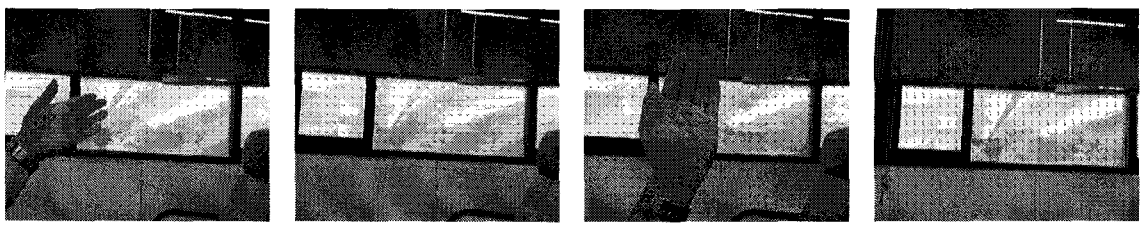
다음 그림 9는 연속된 영상에서 제안된 알고리즘 기반 임의의 점 추적 결과를 나타낸다. 점은 마우스를 이용하여 테두리나 모퉁이 같은 특징이 없는 부분에 표시하였다. 카메라나 물체가 움직임으로서 나타나는 주변의 광류 정보를 이용하여 임의의 점들이 잘 추적 되는 결과를 볼 수 있다. 또한 오른쪽에 있던 점들은 카메라가 움직임으로서 영상 내에서 없어지게 되며, 점들도 함께 사라짐을 볼 수 있다.

표 1은 화소 간격별, 계산 방법별 광류의 추정 속도와 정확도를 나타내며, 추정 속도는 1분간 추정 속도를 평균하여 계산한 결과이고, 정확도는 1분 동안 계산된 결과를 20 프레임 샘플을 뽑아 계산 화소 개



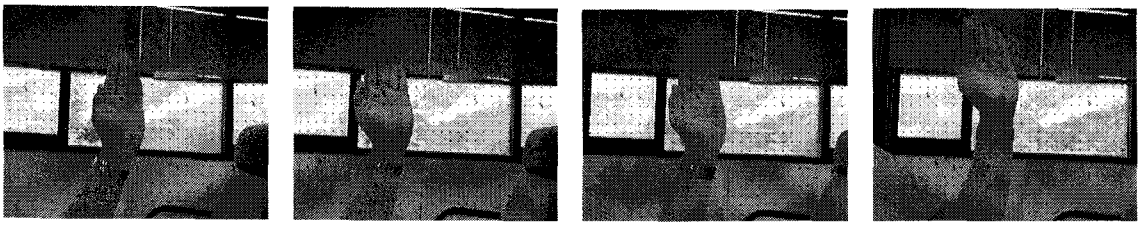
(a) Yosemite 영상 (b) Lucas-Kanade 광류 추정 결과 (c) 제안된 피라미드 기반 정제 광류 추정 결과 (d) 제안된 결과 기반 영상 분할

그림 7. Yosemite 영상을 이용한 광류 알고리즘 비교



(a) 움직이는 물체가 있을 경우 (b) 카메라가 움직이는 경우 (c) 움직이는 물체가 있는 경우 (d) 카메라가 움직이는 경우

그림 8. 광류 제약 조건 추가 전과 후의 실험 결과



(a) 1번째 프레임 (b) 4번째 프레임 (c) 7번째 프레임 (d) 10번째 프레임

그림 9. 연속된 영상에서의 임의의 점 추적 결과

표 1. 광류의 추정 속도 및 점 추적 속도

| 화소 간격 (화소) | 계산방법 | 광류 추정 | | 임의의 점 추적 | |
|---|---|---|---|---|---|
| | | 속도 (fps) | 정확도 (%) | 속도 (fps) | 정확도 (%) |
| 5 | L-K 광류 | 2.4 | 72 | 2 | 60 |
| | 제안된 광류 | 5.2 | 92 | 4.9 | 88 |
| 10 | L-K 광류 | 0.9 | 77 | 0.5 | 62 |
| | 제안된 광류 | 2 | 92 | 1.8 | 90 |

수와 정확히 추정된 화소의 개수를 이용하여 계산한 결과이다. L-K 광류 계산 방법은 영상 피라미드를 사용하지 않은 순수 Lucas-Kanade 광류 방법에 의한 결과이다.

## 6. 결론 및 향후 과제

본 논문에서는 광류를 계산하기 위해 시공간 기울기법을 사용하였다. 시공간 기울기법 중 많이 사용되는 Lucas-Kanade 광류 추정 방법을 기반으로 잡음에 강인하도록 최소자승법을 사용하여 광류 식을 간단하게 정리하였으며, 작은 움직임에 민감하면서도 큰 움직임도 계산할 수 있도록 영상 피라미드를 사용하였다. 영상 피라미드 계산 시 피라미드의 각 수준에서 다음 수준의 광류 값을 예측하여 다음 수준의 초기 값으로 설정 후 반복 광류 계산을 수행하여 증폭되는 오차를 줄일 수 있었다. 또한 광류의 제약조건을 이용하여 기울기 기반 방법 사용 시 발생할 수 있는 튀는 광류를 제거함으로서 정확도를 높일 수 있었다. 그리고 주변의 광류 값과 양선형 보간법을 이용하여 정수가 아닌 광류 벡터를 세밀하게 계산함으로서 테두리나 모퉁이가 아닌 점들의 추적을 가능하게 하였다.

광류 추정은 다양한 컴퓨터 비전문제의 기본 정보를 제공할 수 있다는 장점에도 불구하고 문제 자체가 ill-posed 문제이어서 현재 많은 연구가 진행되고 있으며, 추후에는 광류의 제약조건으로 야기되는 물체의 외곽부분에서 발생하는 오차를 줄일 수 있는 알고리즘 개발이 필요하며, 회전 등의 움직임에 보다 정확한 알고리즘 개선이 필요하다. 또한 광류 정보를 이용하여 카메라의 자세 또는 움직임을 계산할 수 있는 알고리즘을 개발하여 카메라 트래킹 등 여러 분야에 활용이 가능할 것으로 보인다.

## 참 고 문 헌

[1] B.K.P. Horn. *Robot Vision*. MIT Press, Cambridge, MA, 1986.

[2] B.K.P. Horn and B.G. Schunck, "Determining Optical Flow," *Artificial Intelligent*, Vol.17, pp. 185-203, 1981.

[3] H.H. Negel and W. Enkelmann, "An Investigation of Smoothness Constraints for the Estimation of Displacement Vector Fields from Image Sequences," *IEEE Trans. Pattern Anal. Machine Intell.*, Vol.8, No.5, pp, 565-593, 1986.

[4] M. Subbarao, "Interpretation of Image Flow: A Spatio-Temporal Approach," *Pattern Analysis and Machine Intelligence*, Vol.11, pp. 266-278, 1989.

[5] C.S. Fuh and P. Maragos, "Region based Optical Flow Estimation," *Computer Vision and Pattern Recognition*, pp. 130-135, 1989.

[6] P.-C.Chung, C.-L. Huang, and E.-L. Chen, "A Region-based Selective Optical Flow Back-

projection for Genuine Motion Vector Estimation," *Pattern Recognition*, Vol.40, No.3, pp. 1066-1077, 2007.

[7] C.Y. Wang, H. Sun, S. Yada, and A. Rosenfeld, "Some Experiments in Relaxation Image Matching using Corner Features," *Pattern Recognition*, Vol.16, No.2, pp. 167-182, 1983.

[8] B. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," *In International Joint Conference on Artificial Intelligence*, pp. 674-679, 1981.

[9] A. Bruhn, J. Weickert, and C. Schnorr, "Lucas/Kanade Meets Horn/Schunck: Combining Local and Global Optic Flow Methods," *International Journal of Computer Vision*, Vol.31, No.3, pp. 221-231, 2005.

[10] F. Lauze, P. Kornprobst, and E. Memin, "A Coarse To Fine Multiscale Approach For Linear Least Squares Optical Flow Estimation," *British Machine Vision Conference*, Vol.2, pp. 767-776, 2004.

[11] X.R. Gerhard and N.W. Joseph, *Handbook of Computer Vision Algorithms in Image Algebra 2nd ed.* CRC Press, 2001.

**이 재 광**

2003년 홍익대학교 전자전기공학부 졸업(공학사)
2005년 인하대학교 전기공학과 졸업(공학석사)
2005년~현재 과학기술연합대학원대학교 박사과정 재학중
관심분야 : 영상처리, 컴퓨터 비전, 3차원 복원, 카메라 트래킹

**박 창 준**

1994년 경북대학교 전자공학과 졸업(공학사)
1996년 경북대학교 전자공학과 졸업(공학석사)
2000년 경북대학교 전자공학과 졸업(공학박사)
1998년~현재 한국전자통신연구원 재직
2005년~현재 과학기술연합대학원대학교 겸임교수
한국전자통신연구원 HD 게임 연구팀 팀장
관심분야 : 컴퓨터 비전, 3차원 복원, 카메라 트래킹, 게임 엔진