동영상으로부터 3차원 물체의 모양과 움직임 복원

정 병오°* 이 준호° 김 형곤° 고 한석*
한국과학기술연구원 정보전자연구부°/고려대학교 전자공학과*

Abstract

본 논문에서는 SVD(Singular Value Decompsition)를 사용하여 동영상으로부터 효과적으로 잡음을 제거하며 카메라의 움직임과 3차원적인 물체의 모양을 추정해내는 순차적인(sequential) 방법을 제안한다. SVD를 사용하여 정확하고 안정성있게 동작하는 factorization방법은 Tomasi와 Kanade [11]에 의해서 처음으로 제안되었으나 일괄처리형(batch-type) 계산 구조로서 실시간 온라인 응용이 불가능하다. 본 연구에서는 제안한 준원근 투영 카메라 모델에 대한 순차적인(sequential) 해법은 카메라의 축 방향의 움직임이 큰 경우도 적용할 수 있고 매 프레임의 데이타가 들어올 때마다 그 프레임에서의 결과를 생성하며 실행 속도가 빠르므로 실시간 적용이 가능하다. 실험 결과는 일괄 처리 방법으로 구한 결과와 비교하여 볼 때 상당히 정확한 물체의 3차원 움직임과 모양이 재생됨을 보여 준다.

제 1 절 서론

본 논문에서는 SVD(singular value decomposition)를 사용하여 움직이는 카메라가 고정된 물체를 촬영한 동영상으로부터 카메라의 움직임과 물체의 3차원 모양을 추정해내는 방법을 제안한다. SVD를 사용하여 물체의 3차원 움직임과 모양을 추정하는 연구로서는 원근 투영 카메라 모델을 선형적으로 근사화한 준원근 투영 카메라 모델[12]에 대한 일괄처리(batch) 해법[7]과 정사투영 (orthographic projection) 카메라 모델에 대한 순차적(sequential) 처리 해법[6]이 있다. 그러나 [7]의 방법은 원근 효과를 고려한 카메라 모델을 사용하였으나 실시간 처리가 필요한 경우에는 적용이 불가능하고, [6]의 경우에는 정사투영 카메라 모델을 사용하였기 때문에 카메라의 축 방향의 움직임이 큰 경우 많은 오차를 수반한다. 본 연구에서는 준원근투영 카메라 모델에 대한, 실시간 적용이 가능한 순차적인 처리 방법을 제시하였다.

동영상에서 물체의 모양과 움직임을 추정하는 기법은 크게 특징점을 기반으로 하는 방법과 optical flow를 사용하는 방법으로 나눌 수가 있다. 특징점을 기반으로 하는 방법으로, Kalman 필터를 이용하는 방법[5][3][1]은 초기화를 잘못하면 발산하는 경우가 발생하여 초기화를 위하여 부가적인 과정이 필요하고, 움직임을 나타내는 E-행렬을 이용한 Huang[4]은 물체의 3차원적인 움직임을 구하기 위하여 미리 물체의 구조를 안다고 가정하였고 작은 움직임만을 고려하였으므로 응용에 제한이 있다. optical flow를 사용하는 방법으로서는 얼굴 부분의 움직임을 추정하기 위하여 평면(planar)모델을 사용하거나[2] 타원체(ellipsoidal)모델을 사용하여[8] optical flow regularization 기법을 적용였다. 하지만 이러한 방법은 프레임수가 증가할수록 누적된 오차때문에 오차가 더욱 커지는 문제를 일으킨다. 반면에 기존의 SVD factorization을 이용한 방법은 특징점을 기반으로하는 물체의 3차원 움직임과 모양을 추정하는 방법으로서 Kalman 필터를 이용한 방법의 단점인 초기화의 문제를 해결하며 실행 속도가 빠르다는 잇점이 있지만 메모리 증가와 실시간 적용이 어려운 일괄 처리의 단점이 있다.

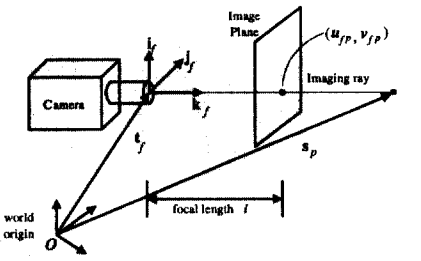

그림 1: 전체 좌표 시스템

본 연구에서는 그림 1에서 보듯이 물체는 정지하여 있고, 카메라가 움직인다고 가정하고 물체의 모양 ($\vec{s}_p$, 즉 물체표면의 특징점들의 좌표)과 카메라의 움직임을 나타내는 회전($\vec{i}_f, \vec{j}_f, \vec{j}_f$, 즉, 전체 좌표계에서의 카메라 축의 방향)과 평행이동(즉, $\vec{t}_f$)량을 매 프레임마다 계산한다. 구하여 진 카메라의 물체에 대한 상대적인 움직임을 카메라에 대한 물체의 움직임으로 전환하면 물체의 움직임과 모양을 계산할 수 있다. 본 연구에서 제안한 SVD factorization을 이용한 순차적인 해법은 일괄처리 해법의 문제점인 메모리 증가 문제가 없으며, Kalman 필터를 이용한 방법보다 실행 속도가 빠르고 실시간 적용이 가능하다는 잇점이 있다. 카메라 모델로서는 비선형적인 원근 투영(perspective projection)을 선형화한 준원근투영 카메라 모델을 사용하여 기존의 방법의 문제점인 카메라의 축 방향의 움직임이 큰 경우발생하는 오차를 해결하였고, SVD를 이용하여 측정 오차의 성분을 제거하였다. 실험 결과 준원근 투영 카메라 모델에 대한 일괄 처리 방법으로 구한 결과와 비교하여 볼 때 정확한 물체의 3차원 움직임과 모양이 재생됨을 알 수 있었다.

본 논문의 구성은 제 2 절에서는 준 원근 투영 카메라 모델에 대하여 살펴보고, 제 3 절에서는 각 프레임에서의 데이터를 다 취합한 후에 물체의 모양과 각 프레임에서 움직임을 측정하는 일괄 처리 방법에 대하여 설명하고 제 4 절에서는 실시간 적용을 위해 매 프레임의 데이타가 들어올 때마다 그 프레임에서의 결과를 생성하는 순차적인 해법에 대하여 기술한다. 제 5 절에서는 실험 결과를 논한다.

제 2 절 준 원근 투영 카메라 모델

원근 투영 카메라 모델의 비선형적인 특성을 선형

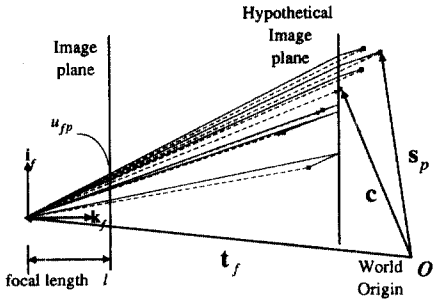

그림 2: 준 원근 투영 카메라 모델

적인 상태로 근사화한 준 원근 투영 카메라 모델(그림 2)은 *shape from texture* 문제를 해결하기 위하여 Ohta[12]에 의하여 제안되었다. 3차원 물체상의 한 점의 준 원근 투영 방식에 의한 영상 좌표는 다음과 같이 구한다.

1. 카메라의 촛점과 물체의 중심($\vec{c}$)을 직선으로 연결하고 이 직선에 평행하게 물체의 점($\vec{s}_p$)을 가상 영상면(hypothetical image plane)과의 교점을 구한다. 가상 영상면은 이미지 평면과 평행하고 물체의 중심($\vec{c}$)를 지나는 평면이다.

2. 구하여진 교점을 원근 투영 카메라 모델로 이미지 평면상에 투영한다.

이렇게 구한 좌표는 교점이 있는 평면과 이미지 평면은 평행하기 때문에 가상 평면과 영상 좌표를 단지 카메라의 촛점 거리와 두 평면 사이의 거리에 비례하여 선형적으로 축소하는 효과를 갖는다.

3차원 물체상의 임의의 점 $\vec{s}_p$ 를 $\vec{c}-\vec{t}_f$ 방향(즉, 카메라의 촛점에서 물체의 중심($\vec{c}$)으로의 방향)으로 투영한 가상 영상면에서의 좌표 $\vec{s}'_{fp}$ 는 다음과 같다.

$$\vec{s}'_{fp} = \vec{s}_p - \frac{\left(\vec{s}_p \cdot \vec{k}_f\right) - \left(\vec{c} \cdot \vec{k}_f\right)}{\left(\vec{c}-\vec{t}_f\right) \cdot \vec{k}_f} \left(\vec{c}-\vec{t}_f\right) \qquad (1)$$

이 점을 이미지 평면상에 원근 투영하여 얻은 좌표를 (u_{fp}, v_{fp})라고 하자. 식을 단순화하기 위해서 단위 촛점 거리를 가정(즉, $l=1$)하고, 전체 좌표계의 원점(world origin)을 물체의 무게중심(즉, $\vec{c} = \frac{1}{P}\sum_{p=1}^{P}\vec{s}_p = 0$)에 놓으면,

$$\begin{aligned} u_{fp} &= \frac{1}{z_f}\left\{\left[\vec{i}_f + \frac{\vec{i}_f \cdot \vec{i}_f}{z_f}\vec{k}_f\right] \cdot \vec{s}_p - \vec{t}_f \cdot \vec{i}_f\right\} \\ v_{fp} &= \frac{1}{z_f}\left\{\left[\vec{j}_f + \frac{\vec{j}_f \cdot \vec{i}_f}{z_f}\vec{k}_f\right] \cdot \vec{s}_p - \vec{t}_f \cdot \vec{j}_f\right\}. \end{aligned} \qquad (2)$$

이를 간략히 표현하면

$$u_{fp} = \vec{m}_f \cdot \vec{s}_p + x_f, \; v_{fp} = \vec{n}_f \cdot \vec{s}_p + y_f \qquad (3)$$

이다. 여기서,

$$z_f = -\vec{t}_f \cdot \vec{k}_f, \; x_f = -\frac{\vec{t}_f \cdot \vec{i}_f}{z_f}, \; y_f = -\frac{\vec{t}_f \cdot \vec{j}_f}{z_f} \qquad (4)$$

이고

$$\vec{m}_f = \frac{\vec{i}_f - x_f\vec{k}_f}{z_f}, \; \vec{n}_f = \frac{\vec{j}_f - x_f\vec{k}_f}{z_f}. \qquad (5)$$

식 (3)를 보면 움직임 정보를 나타내는 부분인 $\vec{m}_f, \vec{n}_f$와 모양 정보를 나타내는 부분인 $\vec{s}_p$가 분리되어 표현될 수 있음을 보여 준다.

제 3 절에서는 식 (3) 와 같이 나타내어지는 데이터를 모든 프레임에 대하여 다 취합한 후에 물체의 모양과 각 프레임에서 움직임을 측정하는 일괄 처리 방법에 대하여 기술한다.

제 3 절 일괄 계산 형태의 해법

1 SVD를 이용한 모양 공간과 움직임 공간의 분리

임의의 한 점에 대한 식 (2)를 P개의 특징점 $(\vec{s}_1, \vec{s}_2, \cdots, \vec{s}_p)$ 과 F개의 프레임 전부를 고려하여 행렬식으로 표현하면 다음과 같은 식을 얻는다.

$$\begin{bmatrix} u_{11} & \cdots & u_{1P} \\ \vdots & \vdots & \vdots \\ u_{F1} & \cdots & u_{FP} \\ v_{11} & \cdots & v_{1P} \\ \vdots & \vdots & \vdots \\ v_{F1} & \cdots & v_{FP} \end{bmatrix} = \begin{bmatrix} m_1 \\ \vdots \\ m_F \\ n_1 \\ \vdots \\ n_F \end{bmatrix} \begin{bmatrix} s_1 & \cdots & s_P \end{bmatrix} + \begin{bmatrix} x_1 \\ \vdots \\ x_F \\ y_1 \\ \vdots \\ y_F \end{bmatrix} \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}$$

또는 간단하게 ,

$$W = MS + T[1 \quad \cdots \quad 1] \qquad (6)$$

로 표현할 수 있는데, 여기서 $W(2F \times P)$는 입력 데이터의 행렬(measurement matrix), $M(2F \times 3)$은 움직임을 나타내는 행렬(motion matrix), $S(3 \times P)$은 모양을 나타내는 행렬(shape matrix), $T(2F \times 1)$은 평행이동을 나타내는 벡터(translation vector)이다.

평행이동 T의 원소인 x_f와 y_f 는 식 (2)와 $\vec{c} = \frac{1}{P}\sum_{p=1}^{P}\vec{s}_p = 0$을 이용하여 특징점들의 좌표로부터 다음과 같이 직접 구하여 진다.

$$x_f = \frac{1}{P}\sum_{p=1}^{P}u_{fp}, \; y_f = \frac{1}{P}\sum_{p=1}^{P}v_{fp} \qquad (7)$$

식 (6) 의 $T[1 \quad \cdots \quad 1]$을 좌변으로 이항하면

$$W^* = W - T[1 \quad \cdots \quad 1] = MS \qquad (8)$$

이고 여기서 W^*는 rank가 3인 두개의 행렬 M과 S의 곱으로서 나타내어지며 rank가 3보다 클 수 없으나 입력 영상에 존재하는 잡음에 의해 3보다 크게 된다. 입력 영상에 존재하는 잡음을 제거하고 M과 S를 분리하기 위해 SVD를 사용한다. 입력 행렬(measurement matrix) W^*를 SVD하면, 상대적으로 값이 매우 큰 3개의 singular value들과 나머지 0에 가까운 singular value들(잡음에 의한 singular value들)이 나오게 된다. 이 중 가장 큰 3개를 제외한 나머지 값들은 잡음 때문에 나타나는 값들이므로 잡음에 의한 영향을 제거하기 위해서 가장 큰 3개만을 채택하고 나머지는 0으로 한다. 그러므로 $W^* \in R^{2F \times P}$ 의 SVD 결과, W^*_{SVD}는 두

개의 orthogonal matrix $Q_1 \in R^{2F\times 3}$, $Q_2 \in R^{P\times 3}$ 와 대각행렬 $\Sigma \in R^{3\times 3}$로 표현할 수 있다.

$$W^*_{SVD} = Q_1 \Sigma Q_2^T \tag{9}$$

단, $\Sigma = diag\,(\sigma_1, \sigma_2, \sigma_3)$ 이고 $\sigma_1 \geq \sigma_2 \geq \sigma_3 > 0$.

Q_1는 W^*_{SVD}의 왼쪽 singular matrix의 열(column)중에서 가장 큰 3개의 singular value들에 대응하는 열로 이루어진 행렬이고, Q_2 는 W^*_{SVD}의 오른쪽 singular matrix의 열(column)중에서 가장 큰 3개의 singular value들에 대응하는 열로 이루어진 행렬이다. Σ는 singular value중에서 가장 큰 3개를 대각 성분으로 갖는 대각행렬이다. 따라서 $\hat{M}$ 과 $\hat{S}$ 를 다음과 같이 정의함으로써 W^*_{SVD}를 분리할 수 있다.

$$W^*_{SVD} = \hat{M}\hat{S} \tag{10}$$

$$\hat{M} = Q_1\Sigma,\ \hat{S} = Q_2^T \tag{11}$$

여기서

$$\hat{M} = \left[\,\hat{\vec{m}}_1 \quad \cdots \quad \hat{\vec{m}}_f \quad \hat{\vec{n}}_1 \quad \cdots \quad \hat{\vec{n}}_f\,\right]^T,$$

$$\hat{S} = \left[\,\hat{\vec{s}}_1 \quad \cdots \quad \hat{\vec{s}}_p\,\right].$$

$\hat{M} = Q_1\Sigma$ 에 의해서 전개(span)된 공간(space)은 움직임을 나타내는 공간이므로, $\hat{M}$을 움직임 공간(motion space)이라고 하고, $\hat{S} = Q_2$에 의해서 전개된 공간은 물체의 모양을 나타내는 공간이므로 $\hat{S}$ 을 모양 공간(shape space)이라고 한다. 이러한 공간의 차원 (dimension)은 rank 이론에 의해서 3이라는 것을 알 수가 있다. 결과적으로 보면 SVD를 사용한 factorization은 높은 차원을 갖는 입력 공간(input space)에서 차원이 3인 움직임을 나타내는 부분 공간(subspace) $\hat{M}$과 모양 공간을 나타내는 부분 공간 $\hat{S}$을 찾는 것이다.

제 4 절 실시간 적용을 위한 순차적 해법

이 절에서는 실시간 적용을 위해 매 프레임의 데이타가 들어올 때마다 그 프레임에서의 결과를 생성하는 순차적인 해법에 대하여 기술한다.

1 매 프레임에서 입력 처리

매 프레임에서 시스템에 들어오는 데이타는 이미지 평면에서의 특징점의 좌표로 이루어진 다음의 두개의 벡터 $\vec{U}_f$와 $\vec{V}_f$이다.

$$\vec{U}_f = [u_{f1}, u_{f2}, \cdots, u_{fP}]^T,\ \vec{V}_f = [v_{f1}, v_{f2}, \cdots, v_{fP}]^T$$

여기서 u_{fi}와 v_{fi}는 f번째 프레임에서의 i번째 특징점의 이미지 평면에서 x와 y좌표를 각각 나타낸다. 실시간 적용을 위한 순차적 해법은 매 프레임마다 $(\vec{U}_f, \vec{V}_f)$를 입력하여 그 프레임(f 번째)에 해당하는 움직임$(\vec{i}_f, \vec{j}_f, \vec{k}_f)$과 모양 정보$(S_f)$를 계산하고, 다음 프레임($f+1$ 번째)에 대한 입력$(\vec{U}_{f_1}, \vec{V}_{f+1}$)을 받고 새로운 카메라 움직임$(\vec{i}_{f+1}, \vec{j}_{f+1}, \vec{k}_{f+1})$과 더 정확해진 모양정보$(S_{f+1})$을 계산한다. 여기서 새로운 matrix time series $Z_f \in R^{P\times P}$를 다음과 같이 정의하여

$$\begin{aligned} Z_f &= Z_{f-1} + U_f U_f^T + V_f V_f^T \\ &= W_f^T W_f \end{aligned} \quad 1 \tag{12}$$

입력 행렬로 사용하는데 그 당위성은 제 2 절의 서두에 기술되어 있다. 입력 행렬, Z_f(식 (12))는 프레임 수가 계속 증가하여도 메모리 사이즈가 증가하지 않는 이점이 있다.

2 순차적으로 모양 공간과 움직임 공간 구하기

W_f 의 rank는 3이므로 Z_f의 rank도 3이고 W_f 의 SVD가 다음과 같다면,

$$W_f = Q_{1f}\Sigma_f Q_{2f}^T \tag{13}$$

여기서, $Q_{1f} \in R^{2f\times 3}$, $Q_{2f} \in R^{P\times 3}$이고 $\Sigma_f = diag\,(\sigma_{f1}, \sigma_{f2}, \sigma_{f3}),\ \ \sigma_{f1} \geq \sigma_{f2} \geq \sigma_{f3} > 0$,

$$\begin{aligned} Z_f &= (Q_{1f}\Sigma_f Q_{2f})^T (Q_{1f}\Sigma_f Q_{2f}) \\ &= Q_{2f}\Sigma_f^2 Q_{2f}^T \end{aligned} \tag{14}$$

Z_f의 eigenvector들은 W_f의 오른쪽 singular vector들인 Q_{2f}와 같다는 것을 의미하므로 Z_f의 eigenvector들을 구함으로서 모양 공간(shape space)을 나타내는 $\hat{S}$를 구할 수 있다. singular value 행렬인 Σ_f는 Z_f의 eigenvalue들의 제곱근에 해당한다. 모양 공간(shape space)을 알면 식 (13)에 의해서 현재 (f번째 프레임) 움직임 공간(motion space)을 나타내는 $\hat{\vec{m}}_f, \hat{\vec{n}}_f$는 다음과 같이 구할 수 있다.

$$\hat{\vec{m}}_f = U_f^T Q_{2f},\ \hat{\vec{n}}_f = V_f^T Q_{2f} \tag{15}$$

3 유일한 해를 구하기 위한 제한 조건

하나의 행렬을 두개 행렬의 곱으로 나타내는 것은 유일하지 않으므로 위의 식 (9)을 식 (11)와 같이 두개의 행렬로 분리하는 것은 유일하게 결정되지 않는다. 이의 물리적인 의미는 움직임 공간과 모양 공간을 구하였지만 현재의 정확한 움직임과 물체의 정확한 모양을 알 수는 없고 단지 정확한 값이 들어 있는 공간만을 알고 있을 뿐이다. 따라서 정확한 값을 찾기 위해서 선형 변환을 나타내는 non-singular matrix A를 이용하여 식 (10)에 다음과 같이 삽입하고 일정한 제한 조건을 부과하여 A 행렬을 계산하여 움직임 공간 $\hat{M}$ 에서 정확한 움직임 M 과 모양 공간 $\hat{S}$ 에서 정확한 모양 S를 구한다.

$$\begin{aligned} W^*_{SVD} &= \hat{M}AA^{-1}\hat{S} = MS \\ M &= \hat{M}A,\ S = A^{-1}\hat{S} \end{aligned} \tag{16}$$

A 를 구하기 위해서 다음의 조건들을 이용한다. 식(5)로부터

$$|\vec{m}_f|^2 = \frac{1+x_f^2}{z_f^2},\ \ |\vec{n}_f|^2 = \frac{1+y_f^2}{z_f^2} \tag{17}$$

이고 다음의 두 제한 조건을 만족하여야 한다.

[제한 조건 1]:

$$\frac{|\vec{m}_f|^2}{1+x_f^2} = \frac{|\vec{n}_f|^2}{1+y_f^2} \quad \left(= \frac{1}{z_f^2}\right) \tag{18}$$

[제한 조건 2]:

$$\vec{m}_f \cdot \vec{n}_f = \frac{x_f y_f}{z_f^2} = x_f y_f \frac{1}{2}\left(\frac{|\vec{m}_f|^2}{1+\vec{x_f^2}} + \frac{|\vec{n}_f|^2}{1+y_f^2}\right) \tag{19}$$

[제한 조건 2]를 보면 식 (18)에서 $\frac{1}{x_f^2}$ 에 대한 값을 식 (18)에 있는 두 개중 하나를 사용할 수 있지만 잡음이 섞인 입력 때문에 두개의 값은 정확하게 일치하지는 않을 수 있기 때문에 두 값의 평균을 취한다.

대칭행렬 $L \in R^{3\times 3}$을 다음과 같이 두면,

$$L = AA^T = \begin{bmatrix} l_1 & l_2 & l_3 \\ l_2 & l_4 & l_5 \\ l_3 & l_5 & l_6 \end{bmatrix}$$

$$\vec{m}_f \cdot \vec{n}_f = \hat{\vec{m}}_f A A^T \hat{\vec{n}}_f^T = \hat{\vec{m}}_f L \hat{\vec{n}}_f^T \qquad (20)$$

이고 식 (20)에서 구하려는 L만을 분리해서 분리해서 표현하기 위해 L을 $\vec{l} = [l_1 \quad l_2 \quad l_3 \quad l_4 \quad l_5 \quad l_6\ ] \neq \vec{0}$로 두면, 식 (20)는 다음과 같이 나타낼 수 있다.

$$\vec{m}_f \cdot \vec{n}_f = g\left(\hat{\vec{m}}_f, \hat{\vec{n}}_f\right)\vec{l}$$

여기서,

$$\vec{m}_f = (m_{f1}\ m_{f2}\ m_{f3}),\ \vec{n}_f = (n_{f1}\ n_{f2}\ n_{f3})$$

이고

$$g(\vec{m}_f, \vec{n}_f) = [m_{f1}n_{f1}\ 2m_{f1}n_{f2}\ 2m_{f1}n_{f3}\ m_{f2}n_{f2}\ 2m_{f2}n_{f3}\ m_{f3}n_{f3}]$$

이다. 따라서 식 (18), (19)은 행렬 형태로 다음과 같이 나타낼 수 있다.

[제한 조건1]

$$\vec{\alpha}_f \vec{l} = 0,$$

$$\vec{\alpha}_f = \frac{g\left(\hat{\vec{m}}_f, \hat{\vec{m}}_f\right)}{1+{x_f}^2} - \frac{g\left(\hat{\vec{n}}_f, \hat{\vec{n}}_f\right)}{1+{y_f}^2}.$$

[제한 조건2]

$$\vec{\beta}_f \vec{l} = 0,$$

$$\vec{\beta}_f = g\left(\hat{\vec{m}}_f, \hat{\vec{n}}_f\right) - x_f y_f \frac{1}{2}\left(\frac{g\left(\hat{\vec{m}}_f, \hat{\vec{m}}_f\right)}{1+x_f^2} + \frac{g\left(\hat{\vec{n}}_f, \hat{\vec{n}}_f\right)}{1+y_f^2}\right).$$

제한 조건을 f번째 프레임까지 적용하면

$$G_f \vec{l} = \vec{0} \qquad (21)$$

$G_f = [\,\vec{\alpha}_1 \quad \vec{\beta}_1 \quad \cdots \quad \vec{\alpha}_f \quad \vec{\beta}_f\,]^T$ 이다.

식 (21)가 $\vec{l} \neq \vec{0}$ (trivial solution 이 아닌) $\vec{l}(\| \vec{l} \| = 1)$이 존재하려면 nullity($G_f$)≥1 이어야 하지만 잡음의 영향때문에 G_f는 full rank(즉, Nullity가 0)를 갖는 over-constrained system이므로 $\vec{0}$이 아닌 $\vec{l}$은 존재하지 않는다. 잡음의 영향을 고려하고 $\vec{l} \neq \vec{0}$인 $\vec{l}$을 구하기위해 에러 $\| G_f \vec{l} - \vec{0} \|$를 최소로 하는 LMS(Least Mean Square)해인 $\vec{l}$을 구하여야 한다. LMS 해를 구하기 위해 식 (21)의 양변에 G_f^T를 곱하면

$$G_f^T G_f \vec{l} = \vec{0} \qquad (22)$$

이고 $G_f^T G_f$를 eigendecomposition을 하면 6개의 eigenvalue들과 eigenvector들이 각각 존재한다. f 프레임에서의 $G_f^T G_f$는 다음과 같이 구한다.

$$G_f^T G_f \in R^{6\times 6} = G_{f-1}^T G_{f-1} + \alpha_f \alpha_f^T + \vec{\beta}_f \vec{\beta}_f^T$$

eigenvalue들중에서 가장 작은 값에 해당하는 eigenvector를 법선(normal) 벡터로 갖는 평면은 G_f 의 행(row) 벡터들과의 오차를 최소로 갖는 평면이 된다. 그러므로 D_f의 eigenvalue중에서 가장 작은 값에 해당하는 eigenvector를 $\vec{l}$로 정함으로써 $\| G_f \vec{l} - \vec{0} \|$를 최소화 할 수 있다. 이렇게 구한 L이 positive definite인한 cholesky decomposition[9]을 사용하면 L 에서 선형 변환 행렬 A를 구할 수 있다.

4 물체의 모양과 카메라의 움직임 구하기

위와 같이 계산하여 얻은 행렬 A를 식 (16)에 대입하여 모양 공간 $\hat{S}$ 에서 물체의 모양을 나타내는 S_f를 추정할 수 있고 움직임 공간 $\hat{M}$에서 움직임을 나타내는 $\vec{m}_f = \hat{\vec{m}}_f A$와 $\vec{n}_f = \hat{\vec{n}}_f A$를 구할 수가 있다. 우선, 식 (5)으로부터 다음 식들을 얻을 수 있다.

$$\vec{i}_f = x_f \vec{m}_f + x_f \vec{k}_f,\ \vec{j}_f = z_f \vec{n}_f + y_f \vec{k}_f. \qquad (23)$$

실제적인 카메라의 움직임을 나타내는 $\vec{i}_f, \vec{j}_f, \vec{k}_f$ (즉, f번째 프레임에서 전체 좌표계에서의 카메라의 방향)들이 서로 직교하고 크기가 1이라는 사실과 식 (23)을 이용하면 다음 식이 성립된다.

$$\begin{aligned} \vec{i}_f \times \vec{j}_f &= \left(z_f \vec{m}_f + x_f \vec{k}_f\right) \times \left(z_f \vec{n}_f + y_f \vec{k}_f\right) \\ &= \vec{k}_f \end{aligned} \qquad (24)$$

$$\left|\vec{i}_f\right| = \left|z_f \vec{m}_f + x_f \vec{k}_f\right| = 1 \qquad (25)$$

$$\left|\vec{j}_f\right| = \left|z_f \vec{n}_f + y_f \vec{k}_f\right| = 1 \qquad (26)$$

위식에서 z_f를 알지 못하지만 식 (18)을 적용하면 다음과 같이 식을 얻을 수 있다.

$$E_f \vec{k}_f = H_f \qquad (27)$$

여기서,

$$E_f = \begin{bmatrix} \left(\tilde{\vec{m}}_f \times \tilde{\vec{n}}_f\right) \\ \tilde{\vec{m}}_f \\ \tilde{\vec{n}}_f \end{bmatrix},\quad H_f = \begin{bmatrix} 1 \\ -x_f \\ -y_f \end{bmatrix},$$

$$\tilde{\vec{m}}_f = \sqrt{1+x_f^2}\frac{\vec{m}_f}{|\vec{m}_f|},\quad \tilde{\vec{n}}_f = \sqrt{1+y_f^2}\frac{\vec{n}_f}{|\vec{n}_f|}.$$

그러므로 카메라의 움직임을 나타내는 $(\vec{i}_f, \vec{j}_f, \vec{k}_f)$는 다음과 같이 구해진다.

$$\vec{k}_f = E_f^{-1} H_f,\ \vec{i}_f = \tilde{\vec{n}} \times \vec{k}_f,\ \vec{j}_f = \tilde{\vec{m}} \times \vec{k}_f \qquad (28)$$

식 (28)로 부터 $\vec{i}_f, \vec{j}_f, \vec{k}_f$ 를 구할 수 있고 식 (23) 으로 부터 깊이 정보인 z_f를 계산할 수 있다. $\vec{i}_f, \vec{j}_f, \vec{k}_f, x_f, y_f, z_f$를 알고 있기 때문에 식 (4)를 이용하여 t_f도 구하여 진다.

제 5 절 실험 결과

실험에 사용한 물체는 그림 3에서 보듯이 피라미드 형태로 특징점으로 피라미드의 모서리에 위치한 36개의 점을 사용하였다. 그림 3과 같은 물체에 그림 4에 나

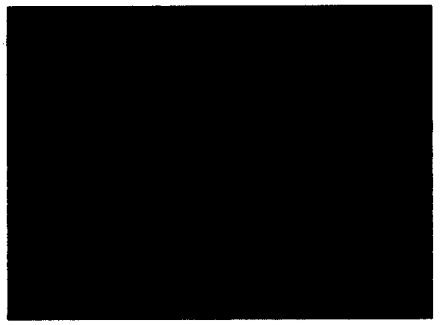

그림 3: 실험에 사용한 피라미드 물체(피라미드 위에서 꼭지점 쪽으로 내려본 영상)

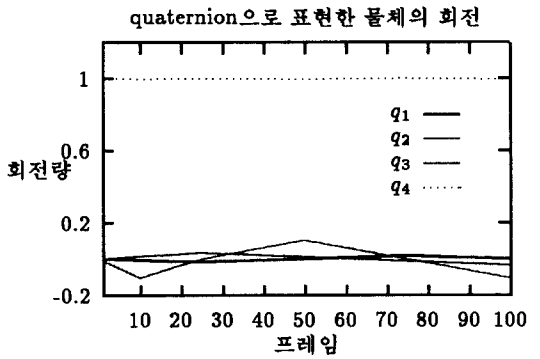

그림 4: 실험에 사용한 실제의 움직임

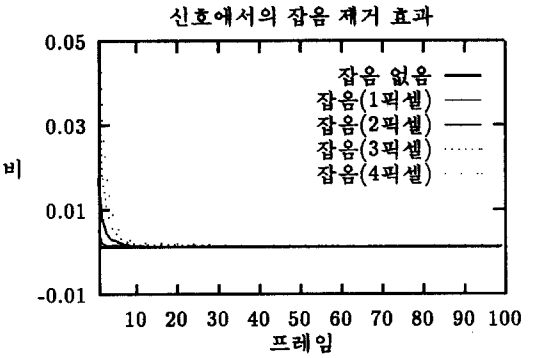

그림 5: singular value의 비: $\frac{\sigma_4}{\sigma_3}$

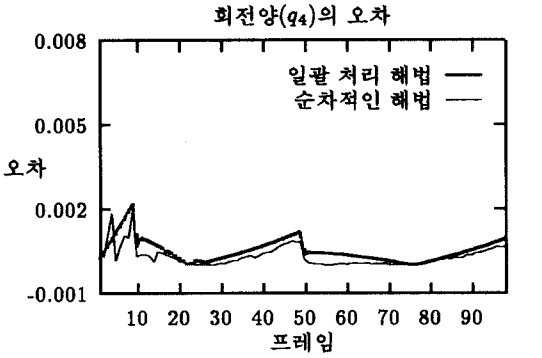

그림 6: 순차적인 해법과 일괄 처리 해법에서의 실제 회전 각도와 복원된 회전 각도의 차이

타나 있는 회전을 주어 만든 100개의 프레임을 사용하여 실험하였다. 잡음이 섞인 데이타의 생성을 위해서는 평균이 0이고 표준편차가 σ인 가우스 분포, $N(0,\sigma^2)$을 이용하여 $(u_{fp}+N(0,\sigma), v_{fp}+N(0,\sigma))$를 이미지 평면에서의 특징점의 좌표로 사용하였다. 물체의 회전은 움직임을 quaternion, (q_1,q_2,q_3,q_4)으로 표현하였다. quaternion에서는 회전축을 나타내는 벡터, $\vec{\alpha}$는 (q_1,q_2,q_3)로, 회전축에 대한 회전각, η는 q_4로 표현한다. 특징점들의 검출과 추적(tracking)은 [10]의 방법을 사용하였다. 그림 5는 입력 행렬(measurement matrix)의 3번째와 4번째로 큰 singular value의 크기를 비교한 것으로 잡음이 없으면 초기의 프레임부터 입력 행렬의 rank가 3임을 나타내고, 잡음이 증가할 수록 초기 프레임에서는 rank가 3으로 수렴하지 않지만 10번째 프레임부터는 rank가 3으로 수렴한다. 이것은 10번째 프레임부터는 잡음이 들어간 신호에서 잡음을 충분히 제거하여 원래의 신호(물체의 모양과 3차원 움직임)를 복구할 수 있음을 나타낸다. 그림 6는 일괄 처리 해법과 순차적인 해법에서 구한 회전에 대한 차이를 나타낸다. 그림 6에서 보면 회전량을 나타내는 q_4의 차이이고, 그림 7는 회전축의 차이를 나타내는데 대략 15 프레임에서 같아지는 것을 볼 수 있다. 그림 8는 물체의 모양(즉, 특징점들의 3차원 좌표)을 매 프레임마다 구하여 실제의 좌표들과 비교하여 차이를 구한 값이다. 잡음이 없는 상황에서는 초기 프레임부터 정확히 모양을 재생함을 나타내지만, 잡음이 섞였을 때 대략 15 프레임부터 정확한 모양을 재생함을 알 수가 있다. 그림 9과 10은 실제의 움직임(그림 4)과 매 프레임마다 구한 움직임을 비교하여 그 차이를 구한 것이다. 그림 9 은 회전 각도를 나타내는 q_4값의 차이를 표시한 것이고 그림 10 은 회전 축을 나타내는 (q_1,q_2,q_3) 값의 차이를 표시한 것이다. 움직임에 대한 오차는 대략 15프레임에서 잡음이 없는 경우와 비슷하게 수렴함을 알 수가 있다. 그림 9 에서 10프레임, 50 프레임 그리고 100프레임 등에서 오차가 큰 이유는 그림 4 에서 보듯이 해당 프레임에서의 움직임이 크기 때문이다.

제 6 절 결론

본 논문에서는 SVD를 사용하여 동영상으로부터 효과적으로 잡음을 제거하며 카메라의 움직임과 물체의 3차원 모양을 추정해내는 순차적인 방법을 제안하였다. 제안된 순차적인 방법은 준원근 투영 카메라 모델을 사용하였고 매 프레임의 데이타가 들어올 때마다 그 프레임에서의 결과를 생성하며 실행 속도가 빠르므로 실시간 적용이 가능하다. 실험 결과에서 알수 있듯이 일괄 처리 방법으로 구한 결과와 비교하여 볼 때 정확한 물체의 3차원 움직임과 모양이 재생됨을 알 수 있었다. 물체의 3차원 움직임과 모양 추정 기술이 핵심으로 사용되는 모든 응용 분야에서의 중요한 기여가 기대된다.

참고 서적

[1] A. Azarbayejani and A. Pentland. Recursive estimation of motion, structure, and focal length.

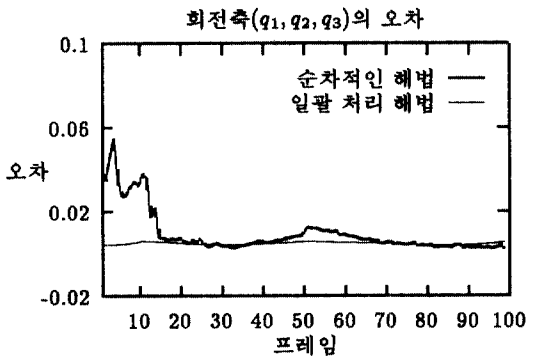

그림 7: 순차적인 해법과 일괄 처리 해법에서의 실제 회전축과 복원된 회전축의 차이

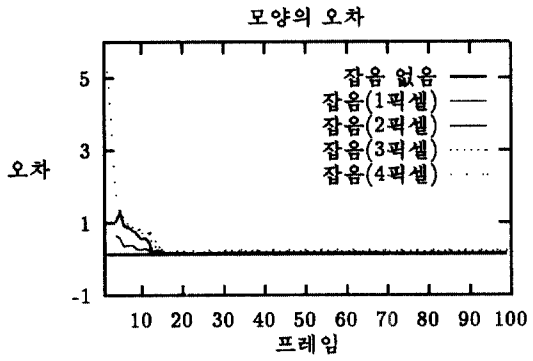

그림 8: 복원된 모양과 실제 모양의 오차

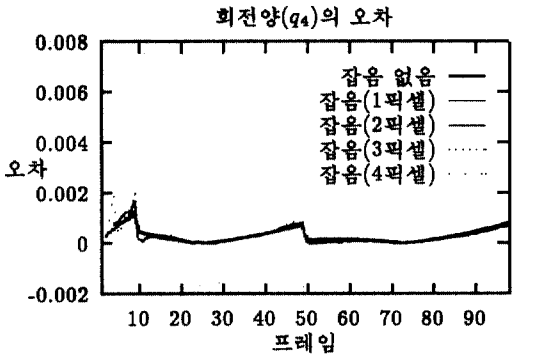

그림 9: 회전 각도에 대한 실제 값과 복원된 값의 차이

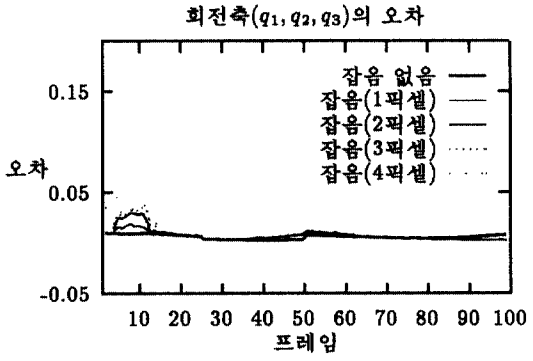

그림 10: 실제 회전축과 복원된 회전축의 차이

IEEE Trans. Pattern Analysis and Machine Intelligence, 7(6):562–575, June 1995.

[2] M. J. Black and Y. Yacoob. Tracking and recognizing rigid and non-rigid facial motions using local parametric model of image motion. *Proceedings of the International Conference on Computer Vision*, pages 374–381, 1995.

[3] T. J. Broida and R. Chellappa. Recursive 3-d motion estimation from a monocular image sequence. *IEEE Trans. Aerospace and Electronic systems*, 26(4):639–656, July 1990.

[4] T. S. Huang and O. D. Faugeras. Some properties of the e-matrix in two-view motion estimation. *IEEE Trans. Anal. Mach. Intel.*, 11(12):1310–1312, December 1989.

[5] N. Ahuja J. Weng and T. S. Huang. Optimal motion and structure estimation. *IEEE Trans. Patt. Anal. Mach. Intel.*, 15(9), September 1993.

[6] T. Morita and T. Kanade. A sequential factorization method for recovering shape and motion from image streams. In *Prec. 1994 ARPA Image Understanding Workshop*, volume 2, pages 1177–1188, November 1994.

[7] C. J. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. Technical Report CMU-CS-93-219, Carnegie Mellon University, 1993.

[8] I. Essa S. Basu and A. Pentland. Motion reguarization for model-based head tracking. In *13th International Conference on Pattern Recognition*, August 1996.

[9] G. Strang. *Linear Algebra and Its Application*, page 334. Harcourt Brace Jovanovich Colledge Publishers, 3rd edition, 1988.

[10] C. Tomas and T. Kanade. Detection and tracking of point features. Technical Report CMI-CS-91-132, Carnegie Mellon University, 1991.

[11] C. Tomasi and T. Kanade. Shape from motion from image streams under orhography: A factorizaion method. *International Journal of Computer Vision*, 9(2):137–1529, November 1992.

[12] Kiyoshi Maenobu Yu-ichi Ohta and Toshiyuki Sakai. Obtaining surface orientation from texels under perspective projection. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, pages 746–751, August 1981.