

論文2000-37SP-11-4

불완전한 궤적을 고려한 강건한 특징점 추적 알고리즘 (A Robust Algorithm for Tracking Feature Points with Incomplete Trajectories)

丁鍾勉*, 文泳植*
(Jong Myeon Jeong and Young Shik Moon)

요 약

특징점의 궤적은 인접한 프레임에 존재하는 특징점 사이의 대응관계로 정의할 수 있다. 실제 영상열에서 존재할 수 있는 잘못된 특징점(false positive, false negative)들은 특징점의 대응관계를 결정할 때 많은 문제를 야기하기 때문에 특징점의 대응관계를 찾는 문제는 어려운 문제로 알려져 있다.

본 논문에서는 새로운 궤적의 나타남, 사라짐 등 불완전한 궤적을 갖는 특징점들을 고려하는 특징점 추적 기법을 제안한다. 정합 척도로서 가중치가 부여된 유클리디언 거리를 사용하고 특징점의 운동 특성을 잘 반영할 수 있도록 그 가중치를 자동으로 조정한다. 대응점 탐색과정에서 치명적인 영향을 줄 수 있는 애매한 특징점이 존재하는 경우를 고려하여 인접한 프레임 사이의 정합점 결정을 그래프에 의한 최적 대응점 탐색 문제로 해결한다. 제안하는 대응점 탐색 알고리즘은 실제 영상열에서 나타날 수 있는 잘못된 특징점들이 대응관계를 결정할 때 주는 영향을 최소화하기 위하여 국부 최적(local optimal)을 찾게되며, 인접한 두 프레임에 m, n 개의 특징점이 주어졌을 경우, 최선의 경우 $O(mn)$, 최악의 경우 $O(m^2n)$ 의 계산량을 필요로 한다. 제안하는 알고리즘은 정합과정에서 잘못된 특징점을 고려하고, 특징점의 운동특성을 잘 반영함으로써 대량의 특징점을 추적하는데도 충분히 적용할 수 있음을 실험을 통해 확인하였다.

Abstract

The trajectories of feature points can be defined by the correspondences between points in consecutive frames. The correspondence problem is known to be difficult to solve because false positives and false negatives almost always exist in real image sequences.

In this paper, we propose a robust feature tracking algorithm considering incomplete trajectories such as entering and/or vanishing trajectories. The trajectories of feature points are determined by calculating the matching measure, which is defined as the minimum weighted Euclidean distance between two feature points. The weights are automatically updated in order to properly reflect the motion characteristics. We solve the correspondence problem as an optimal graph search problem, considering that the existence of false feature points may have serious effect on the correspondence search. The proposed algorithm finds a local optimal correspondence so that the effect of false feature point can be minimized in the decision process. The time complexity of the proposed graph search algorithm is given by $O(mn)$ in the best case and $O(m^2n)$ in the worst case, where m and n are the number of feature points in two consecutive frames.

By considering false feature points and by properly reflecting motion characteristics, the proposed algorithm can find trajectories correctly and robustly, which has been shown by experimental results.

* 正會員, 漢陽大學校 電子計算學科
(Dept. of Computer science & Engr., Hanyang University)

※ 본 연구는 한국과학재단 목적기초연구(2000-2-30300-005-3)지원으로 수행되었음.

接受日字:2000年3月6日, 수정완료일:2000年8月30日

I. 서 론

영상열에 존재하는 이동 물체를 추적하는 문제는 컴퓨터 비전의 가장 중요한 연구분야 중의 하나로써 로봇 공학, 비디오 인덱싱, 자동 운항 등 많은 응용 분야를 가지고 있다. 일반적으로 특징 추적 알고리즘에서 사용한 특징들은 영역(region), 형상(contour), 직선(line), 특징점(feature point) 등으로서, 각각의 프레임에서 특징들을 얻은 다음 서로 인접한 프레임에서 서로 같은 특징을 찾게 된다. 물체 추적에 적합한 특징은 처리하고자 하는 특징이 쉽고 정확하게 추출할 수 있어야 하며, 그 특징이 물체의 시공간적 특성을 잘 반영하여 정합하기에 용이해야 한다.

오래 전부터 동영상의 움직임 정보를 이용하여 특징을 추출하려는 노력이 이루어져 왔다. 그 대표적인 것은 광류(optical flow)를 계산한 다음 그 광류를 이용하여 영역을 추출하는 것인데, 광류는 움직임 경계(motion boundary)에 오류를 가지게 되므로 광류만으로는 영역을 정확하게 추출하기 힘들다^{[1]-[4]}. 최근 들어 활성 형상 모델(active contour model, snake)을 이용하여 형상을 추출하고, 그것을 affine model^[5]이나, Kalman filter^[6]를 이용하여 추적하는 방법에 대한 연구가 활발히 이루어지고 있다. 이 방법들은 특징 추출 단계에서 활성 형상 모델의 에너지 함수를 정의하는 것이 큰 이슈가 되며, 추적 단계에서는 가려짐 문제를 극복하기 위해 고려해야 할 사항이 많다. 이에 비해 특징점은 영역이나 형상 등 다른 특징에 비해 영상열에서 비교적 덜 왜곡되기 때문에 정합 단계에서도 특징 왜곡에 대해 고려해야 할 사항이 줄어드는 장점이 있다. 따라서 본 논문에서는 물체 추적을 위한 특징으로 특징점을 사용한다. 특징점의 추적은 연속으로 입력되는 영상열에 존재하는 특징점의 궤적을 찾아내는 것으로서, 연속적인 영상열에 존재하는 특징점 사이의 대응점을 찾음으로써 특징점들의 궤적을 알 수 있고, 특징점의 궤적은 전체 영상열에서 이동하는 물체들의 움직임을 나타낼 수 있다.

Sethi^[7] 등은 인접한 특징점 사이의 대응관계를 알기 위해 특징점의 이동거리와 이동방향의 일관성을 고려한 경로 일관성 함수(path coherence function)를 정의하고 greedy exchange 알고리즘을 이용하여 최적화 문제로 해결하였다. 그러나 이동방향과 속도에

대한 가중치에 따라 궤적이 크게 달라지는 문제가 있고 긴 영상열에는 부적합하며 계산량이 방대하다는 단점이 있다. Salari^[8] 등은 Sethi 등의 알고리즘을 개선하여 궤적 추출 과정에서 불완전한 궤적에 대한 고려를 포함시켰으나, 특징점 추출 과정에 오류가 생겼을 경우에는 적용하기 어려우며, 영상열의 중간에 궤적이 사라지거나 나타나는 경우를 고려하지 않았다. Rangarajan^[9] 등은 인접 균일성 함수(proximal uniformity function)를 정의하고 인접한 프레임 사이의 대응 가능한 모든 대응쌍을 이용한 상대적 비율을 이용하였는데, 이 방법은 특징점의 정합 순서가 달라질 경우에는 그 결과가 달라질 수도 있으며, 이것은 대응점 탐색 결과에 심각한 오류를 야기할 수 있다. Krishnan^[10] 등은 비행 모의실험에서 카메라의 FOE (Focus of Expansion)와 충돌시간(time-to-contact)을 계산하기 위하여 특징점을 추적하였다. 이 방법은 특징 블록의 밝기 비교를 위한 많은 계산량을 줄이기 위해 특징점의 최대 이동 거리를 3화소 이하로 가정하였고, 영상 내에는 특징블록으로 이용할 수 있는 충분한 양의 텍스처가 존재해야 적용할 수 있다는 단점이 있다. 정재영^[11] 등은 불완전한 궤적을 고려할 수 있는 특징점 추적 알고리즘을 제안하였는데, 특징점 추출, 정합 과정에 많은 임계치를 포함하고 있어서, 적절한 임계치를 찾는 것이 어렵다. Mehrotra^[12]는 특징점의 운동 방향과 속도를 고려한 움직임 균일성 함수(motion uniformity function)를 정합의 척도로 사용하고, 궤적의 확장성을 검사하여 특징점의 궤적을 얻었다. 이 방법은 운동의 방향과 속도에 대한 가중치에 따라 다른 궤적이 나타날 수 있다는 단점이 있다. Chetverikov^[13] 등은 Sethi 등이 사용했던 정합의 척도를 사용하여 특징점의 궤적을 구하였는데, 불완전한 궤적을 추적하는데 장점을 가지고 있지만 특징점의 이동방향과 속도에 대한 가중치에 따라 그 결과가 달라질 수 있으며, Sethi 등의 방법에 비해 추출된 궤적이 부정확하기 때문에 추적해야 하는 특징점의 수가 증가하였을 경우에는 적용하기 힘들다. 정종면^[14] 등은 가중치가 부여된 유클리디언 거리를 정합의 척도로 사용하였는데, Kottke 등이 군집 정합(cluster matching)에 사용하였던 가중치 조정 기법을 특징점의 궤적을 추적하는 적용함으로써 강건한 특징점 추적이 가능하도록 하였다. 그러나 모든 프레임에 동일한 개수의 특징점이 주어진다고 가정함으로써 불완전한 궤적은 추

적할 수 없다는 단점이 있으며, 한 프레임에 운동특성이 서로 다른 여러 물체가 동시에 존재할 경우에는 가중치 조절 기법에 오류가 있을 수 있다.

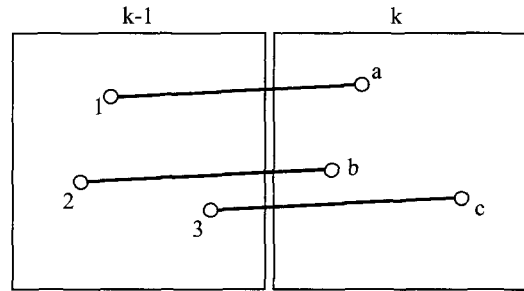
기존의 방법들^{[7], [14]}을 종합하여 살펴보면 대부분의 경우 궤적의 초기화에 문제가 있고 불완전한 궤적을 추적하는데 한계가 있으며, 프레임에 특징점의 개수가 많아지게 되면 고려해야하는 궤적의 경우의 수가 기하급수적으로 증가함으로써 잘못된 궤적을 나타내는 경우가 많다. 본 논문에서는 잡음을 포함하는 대량의 특징점이 존재하는 영상열에서 궤적의 초기화를 포함한 특징점 추적을 수행할 수 있는 대응점 탐색 알고리즘을 제안한다. 제안하는 방법은 대응점 탐색 문제를 그래프 탐색 문제로 해결함으로써 불완전한 궤적, 잡음에 의한 특징점의 왜곡 문제를 극복하고 강건하게 특징점을 추적할 수 있다. 본 논문의 II장에서는 일반적인 대응점 탐색문제를 제기하고 제안하는 알고리즘을 설명한 후, III장에서 실험 결과를 보인 다음, IV장에서 결론을 맺는다.

II. 특징점 추적 알고리즘

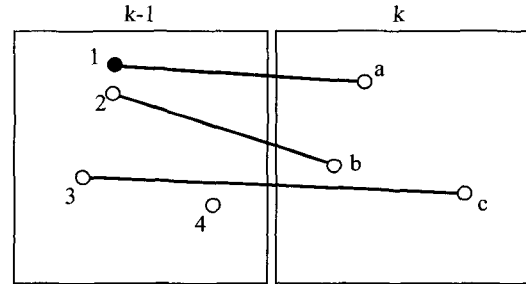
1. 대응점 탐색 문제

입력 영상열을 구성하는 특징점들의 궤적을 얻기 위해서는 먼저 인접한 프레임 사이의 대응점을 구해야 한다. 동영상을 구성하는 각 프레임에서 추출된 특징점들은 여러 가지 잡음과 특징점 추출 알고리즘의 문제로 인해 특징점이 아닌 점이 특징점으로 추출되는 경우와 존재하는 특징점을 정확히 추출하지 못하는 경우를 포함한다. 이런 오류는 대응쌍 결정과정에 영향을 주게 되는데, 잘못된 특징점으로 인한 잘못된 대응이 있을 경우 전체 대응쌍에 그 오류가 파급되므로 심각한 문제를 야기하게 된다. 그림 1은 잘못된 특징점에 의한 잘못된 대응쌍이 있을 경우 전체 대응쌍에 그 오류가 파급되는 것을 보이고 있는데, 그림 1(a)는 잘못된 특징점이 존재하지 않는 경우의 정합쌍들을 보이고 있으며, 그림 1(b)는 잘못된 특징점들이 존재하는 경우의 정합쌍들을 보이고 있다. 그림 1(b)에서 검게 채워진 원은 잘못된 특징점을 의미하는데 "1"로 표시된 잘못된 특징점이 "a"로 표시된 특징점에 대응되면 그 오류가 이후의 대응점 선택과정에 연쇄적으로 영향을 주게 돼서 최종적으로 결정된 정합쌍은 실제 정합쌍과 전혀 다르게 된다. 따라서 일반적인 환경

에서 적용하기 위한 특징점 추적 알고리즘은 이런 문제를 반드시 해결해야 하는데, 기존의 특징점 추적 알고리즘에서는 이런 문제에 대한 고려가 없거나^{[7], [9], [14]}, 미흡하여^{[10]-[13]} 강건한 특징점 추적이 어려웠다. 본 논문에서는 여러가지 이유로 인하여 잘못 추출된 특징점들을 고려하는 강건한 대응점 탐색 알고리즘을 제안한다.



(a) 잘못된 특징점이 존재하지 않은 경우



(b) 잘못된 특징점이 존재하는 경우

그림 1. 대응점 탐색 문제

Fig. 1. Correspondence search problem.

2. 특징점 추출

물체 추적을 위한 특징점이 갖춰야 될 조건은 영상을 구성하는 물체의 구조를 잘 반영하고, 물체의 움직임에 그 속성이 크게 변화하지 않아야 한다. 본 논문에서는 영상의 코너를 추출하여 특징점으로 사용하였는데 코너는 그림 2와 같이 한정된 범위이내에서 에지의 꼭지각이 임계치 이하이면서 국부 최소인 값을 의미하는 것으로서, 코너의 요건을 검사하는 알고리즘은 다음과 같다.

i) 에지(e)를 중심으로 한 일정 영역 R의 평균 밝기 값을 구한다.

ii) 평균 밝기값을 임계치로 하여 영역 R을 이진화

하여 물체영역과 배경영역으로 분리한다.

iii) 에지 e가 물체 영역과 배경 영역의 경계선에 있지 않으면 e는 코너가 아니라고 판정하고 다음 에지 화소에 대하여 i)부터 다시 시작한다.

iv) 에지 e를 시작점으로 하여 경계선 추적(boundary following)을 양방향으로 수행하여 물체영역과 배경영역의 경계선이 영역 R의 경계선과 만나는 두 점을 찾는다(그림 2의 A, B).

v) 에지 e에서 교차하는 두 직선의 내각을 측정하여 임계치보다 크면 e는 코너가 아니라고 판정한 다음 에지 화소에 대하여 i)부터 다시 시작하고, 임계치보다 작으면 e를 후보 코너점으로 판정한 후 다음 에지 화소에 대하여 i)부터 다시 시작한다.

vi) 모든 에지 화소에 대하여 i)-v) 과정을 수행한다.

vii) 후보코너 중 일정 영역이내에서 국부최소인 후보코너를 최종 코너로 판정한다.

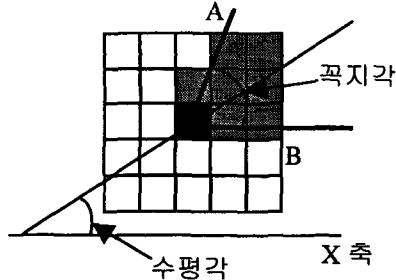


그림 2. 특징점의 정의

Fig. 2. Definition of feature point.

3. 특징 벡터의 구성

각 프레임에서 얻어진 특징점은 인접한 프레임 사이의 대응점을 찾음으로써 그 궤적을 추적할 수 있다. 이를 위해 본 논문에서는 인접한 프레임에서 특징점의 이동방향과 이동 거리는 크게 변하지 않는다는 작은 속도변화 제약(small velocity change constraint)과 최대속도 제약(maximum motion constraint), 그리고 일관된 정합 제약(consistent match constraint)을 이용하여 인접한 프레임 사이의 대응점을 찾는다. 특징점 간의 대응점을 찾기 위해 사용한 특징점의 속성은 특징점의 공간좌표, 꼭지각, 수평각, 이동거리, 이동 방향 등 5가지로서 식 1과 같은 6차원의 벡터로 나타낼 수 있다.

$$X_k^i = [x_k^i, y_k^i, \theta_k^i, h_k^i, d_k^i, m_k^i]^T \quad (1)$$

여기에서 X_k^i 는 k 번째 프레임의 i 번째 특징점을 의미하며, (x_k^i, y_k^i) 는 k 번째 프레임에 있는 i 번째 특징점의 공간 좌표를, θ_k^i 와 h_k^i 는 k 번째 프레임에 있는 i 번째 특징점의 꼭지각과 수평각을, d_k^i 와 m_k^i 는 k 번째 프레임에 있는 i 번째 특징점의 이동방향과 이동거리를 각각 의미한다. 특징점의 꼭지각과 수평각은 그림 2와 같으며 물체의 움직임에 그 값의 변화가 둔감하기 때문에 특징점 추적에 적합하다. 특징점의 속성 중 꼭지각을 제외한 나머지 모든 속성들은 특징점의 이동이나 회전에 따라 그 값이 변화하는데, 인접한 프레임에서는 특징점의 속성의 변화가 작을수록 서로 대응될 가능성이 일반적으로 높다. 또한 처리하고자 하는 동영상에 따라 속성들이 상대적으로 다르게 변화하기 때문에 정합 척도를 정의할 때 이에 대한 고려가 필요하다.

4. 정합 척도

본 논문에서는 식 1과 같은 6차원 벡터들에 대해서 가중치가 부여된 Euclidean 거리를 정합의 척도로 사용하는 데 이는 식 2와 같이 나타낼 수 있다.

$$D_k^{ij} = (X_{k-1}^i - X_k^j)^T W (X_{k-1}^i - X_k^j) \quad (2)$$

여기에서 X_k^i 는 $k-1$ 번째 프레임의 i 번째 특징점과 가장 잘 일치되는 k 번째 프레임 상의 한 특징점이며 D_k^{ij} 는 $k-1$ 번째 프레임의 i 번째 특징점과 가장 잘 일치되는 k 번째 프레임 상의 특징점 X_k^j 사이의 Euclidean 거리를 나타낸다. W 는 식 3, 식 4와 같은 가중치 행렬로 주어진다.

$$W = \begin{bmatrix} w_x & 0 & 0 & 0 & 0 & 0 \\ 0 & w_y & 0 & 0 & 0 & 0 \\ 0 & 0 & w_\theta & 0 & 0 & 0 \\ 0 & 0 & 0 & w_h & 0 & 0 \\ 0 & 0 & 0 & 0 & w_d & 0 \\ 0 & 0 & 0 & 0 & 0 & w_m \end{bmatrix} \quad (3)$$

$$w_x w_y w_\theta w_h w_d w_m = 1 \quad (4)$$

신뢰성 있는 특징점 추적을 위해서는 식 3의 가중치를 적절히 조절하는 것이 필수적인데 본 논문에서는 Kottke 등이 군집 정합을 위해 제안하였던 가중치 조절 방법을 확장 적용하였다^[15]. Kottke 등이 사용하였던 방법을 변형하면, 일련의 대응쌍이 주어졌을

경우 그 대응쌍의 Euclidean 거리의 합을 최소화시킬 수 있는 최적의 가중치는 다음과 같이 얻을 수 있다^[14].

$$w_x = \left[\sum_{i=1}^N (y_{k-1}^i - y_k^i)^2 \right]^{1/6} \left[\sum_{i=1}^N (d_{k-1}^i - d_k^i)^2 \right]^{1/6} \left[\sum_{i=1}^N (b_{k-1}^i - b_k^i)^2 \right]^{1/6} \left[\sum_{i=1}^N (h_{k-1}^i - h_k^i)^2 \right]^{1/6} \quad (5)$$

$$\left[\sum_{i=1}^N (m_{k-1}^i - m_k^i)^2 \right]^{1/6} \left[\sum_{i=1}^N (x_{k-1}^i - x_k^i)^2 \right]^{-5/6}$$

$$w_y = w_x \frac{\sum_{i=1}^N (x_{k-1}^i - x_k^i)^2}{\sum_{i=1}^N (y_{k-1}^i - y_k^i)^2} \quad (6)$$

$$w_b = w_x \frac{\sum_{i=1}^N (x_{k-1}^i - x_k^i)^2}{\sum_{i=1}^N (b_{k-1}^i - b_k^i)^2} \quad (7)$$

$$w_h = w_x \frac{\sum_{i=1}^N (x_{k-1}^i - x_k^i)^2}{\sum_{i=1}^N (h_{k-1}^i - h_k^i)^2} \quad (8)$$

$$w_d = w_x \frac{\sum_{i=1}^N (x_{k-1}^i - x_k^i)^2}{\sum_{i=1}^N (d_{k-1}^i - d_k^i)^2} \quad (9)$$

$$w_m = w_x \frac{\sum_{i=1}^N (x_{k-1}^i - x_k^i)^2}{\sum_{i=1}^N (m_{k-1}^i - m_k^i)^2} \quad (10)$$

즉, 주어진 대응쌍에서 변화량이 적은 속성에 대한 가중치는 크게 하고, 변화량이 큰 속성에 대한 가중치는 작게 하여 동영상의 운동 특성을 자동으로 적절히 반영할 수 있도록 한다.

5. 특징점 추적 알고리즘

대부분의 특징점 추적 알고리즘은 동영상의 두 프레임에 존재하는 특징점 사이의 대응점을 결정할 때, 한 프레임의 각 특징점에 대해 정합 척도를 최소화, 혹은 최대화하는 나머지 프레임의 특징점을 대응점으로 선택한다. 그러나 1절에서 언급한 바와 같이 단순히 정합척도를 최소화, 최대화하는 특징점을 대응점으로 결정하는 것은 실제 환경에서는 적절하지 못하다.

본 논문에서는 잡음에 의한 특징점, 불완전한 궤적 등을 고려함으로써 실제 영상의 특징점을 강건하게 추적하도록 하였다. 이를 위하여 불완전한 궤적을 고려하지 않아서 실제 영상열에는 적용하기 힘들었던 참고 문헌 [14]에서 제안된 특징점 추적 방법을 실제 영상열에 적용할 수 있도록 확장하였다. 참고 문헌 [14]의 알고리즘을 요약하면 다음과 같다.

1) 주어진 가중치를 이용하여 초기 정합쌍을 구하고 이를 이용하여 정합쌍의 유클리디언 거리의 총합 D_k^T 를 계산한다.

2) 1)에서 얻어진 결과에 따라 새로운 가중치를 계산

한다.

3) 정합쌍의 유클리디언 거리의 총합 D_k^T 가 이전 반복과 같으면 1)에서 얻은 정합쌍을 최종 정합쌍으로 결정하고 반복을 멈춘다.

4) 1)-3) 과정을 반복한다.

위의 알고리즘에서 1)은 $k-1$ 번째 프레임의 각 특징점들로부터 가장 가까운 k 번째 프레임상의 점을 순차적으로 선택하는 것이다. 이때 모든 프레임에 동일한 개수의 특징점이 주어지면서 동시에 모든 궤적은 모든 프레임에 존재할 경우에는 대응점 탐색이 효과적으로 이루어지지만, 오류에 의한 특징점이나, 궤적의 나타남, 사라짐 등 불완전한 궤적이 존재할 경우에는 대응점 탐색이 효과적으로 이루어 질 수 없다.

본 논문에서는 불완전한 궤적이 존재하는 경우에 효율적인 대응점 탐색을 위하여 그래프를 이용하였다. 그래프를 이용하여 정합(matching)문제를 해결하는 것은 오래 전부터 다양한 응용분야에서 사용되어 왔는데 정합문제를 해결하기 위한 기존의 그래프 탐색 알고리즘은 전역 최적(global optimal)을 찾는 것으로서 불완전한 궤적이 존재하는 특징점을 추적하는데 적용하기에는 적절하지 않다^[16].

본 논문에서는 잡음에 의하여 잘못 추출된 특징점이 존재하는 경우에 두 프레임 사이의 대응점을 찾기 위해서 두 프레임에 존재하는 특징점을 노드로 하는 bipartite 그래프 $G(N, E)$ 를 구성한다. 그래프의 에지 E 는 가중치가 부여된 에지로서, 서로 다른 프레임에 존재하는 특징점 사이의 정합척도에 의한 정합 정도를 의미한다. 그림 3은 그림 1(b)에 있는 두 프레임에 가중치가 있는 그래프로 나타낸 것으로서, 왼쪽의 노드들은 $k-1$ 번째 프레임의 특징점들을, 오른쪽의 노드들은 k 번째 프레임의 특징점들을 각각 나타낸다. 본 논문에서는 Euclidean 거리를 정합 척도로 하였기 때문에 에지의 가중치는 Euclidean 거리를 의미하며, 그림 3의 에지의 가중치는 그림 1(b)의 두 특징점 사이의 Euclidean 거리를 가정한 것이다. 따라서 두 프레임에 존재하는 특징점들의 최적의 대응쌍을 찾는 문제는 주어진 그래프에서 최소 가중치를 갖는 노드쌍을 찾는 문제가 되며 실선으로 그려진 에지들이 찾고자 하는 최적의 대응쌍에 의한 에지들이 된다. 제안하는 방법을 의사코드(pseudo code)로 나타내면 다음과 같다.

```

WHILE (there exists a node in the graph) BEGIN
  Step 1. 초기화
    1.1 eg ← 임의의 값
    1.2 pe ← eg 값과 다른 임의의 값
    1.3 nd ← 두 프레임 중 노드의 개수가 적은 프레임의 임의의 노드 번호
  Step 2. 최적 정합쌍 선택
    WHILE ((pe ≠ eg) and (there exists an edge in the graph)) BEGIN
      2.1 pe ← eg
      2.2 eg ← nd에 연결된 에지 중 최소 가중치를 갖는 에지의 가중치
      2.3 nd ← 2.2의 최소 가중치를 갖는 에지에 연결된 노드 중 nd와 반대쪽에 있는 노드 번호
    END WHILE;
  Step 3. 선택된 정합쌍을 그래프에서 제거
    IF (pe = eg) THEN BEGIN
      3.1 eg가 연결하는 두 노드를 최적 정합쌍으로 선택
      3.2 eg가 연결하는 두 노드를 그래프에서 제거
      3.3 eg가 연결하는 두 노드에 관련된 모든 에지를 그래프에서 제거
    END
    ELSE BEGIN
      3.4 최소 가중치쌍 탐색 과정 종료
    END IF;
  END WHILE;

```

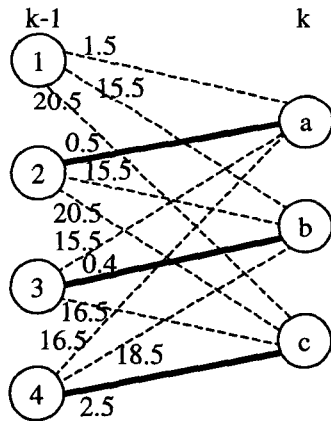


그림 3. 그래프를 이용한 대응점 탐색 기법
Fig. 3. A correspondence search algorithm using weighted graph.

즉, bipartite 그래프에서 적은 수의 노드를 갖는 부노드(sub node) 집합-특징점의 수가 적은 프레임-에 있는 임의의 노드에서 그래프 탐색을 시작하여, 그 노드에 연결된 에지들중 최소의 가중치를 갖는 에지를 타고 계속 이동하다, 방금 지나왔던 에지로 다시 돌아가게 되는 경우가 생기면 그에 해당하는 에지와 그 에지가 연결하는 두 노드를 최종적인 대응쌍으로 결정한다. 이후의 그래프 탐색에서 그 노드에 관련된 모든 것을 제외하고, 다시 임의의 노드에서 그래프 탐색을 계속하게 된다. 제안하는 그래프 탐색 알고리즘에 의

해 찾아진 최소의 가중치를 갖는 에지의 집합은 전역 최적을 보장하지 못한다.

그래프의 대응쌍 탐색 문제에서 전역 최적의 의미는 그래프를 구성하는 모든 노드의 가중치 총합을 최소화 혹은 최대화시키도록 대응쌍을 결정한다는 것을 의미한다. 따라서 잘못된 특징점으로 인한 불완전한 제거가 존재하는 환경에서 특징점 추적을 위해 전역 최적을 찾는 것은 적절하지 못하다. 즉, 특징점 추적을 위한 그래프 탐색기법에서 전역 최적을 찾게 되면 오류에 의한 특징점(false positive)이 최적점을 찾기 위한 과정에 많은 영향을 주기 때문에, 특징점 추적을 위해 전역 최적을 찾는 것은 적당하지 않으며, 제안하는 방법과 같이 국부 최적(local optimal)을 찾는 것이 타당하다고 할 수 있다. 제안된 그래프 탐색 알고리즘은 어떤 노드에서 탐색을 시작하여도 그 탐색 결과는 차이가 없다. 다만, 시작 노드에 따라 탐색을 위한 계산 시간이 달라질 수는 있다.

한편, 특징점의 제거는 영상열의 중간에 사라지거나 새롭게 나타날 수 있기 때문에 현재 프레임에서 제거에 속하지 않은 특징점들은 다음 프레임에 대한 대응점을 찾을 때 새로운 제거를 만들어 낼 수 있다. 반대로 현재 프레임에서 제거에 포함되었던 특징점들은 다음 프레임에서는 그 제거가 사라져서 대응점을 찾지 못할 수도 있다. 이러한 고려사항을 포함하여 선택된

대응쌍을 이용한 궤적의 판별은 다음과 같이 할 수 있다.

- 1) 기존 궤적의 확장 : $k-1$ 번째 프레임의 특징점 중 궤적에 포함돼 있는 특징점에 대한 대응점을 k 번째 프레임에서 찾아 궤적을 확장한 후 이후의 연산에서 제외한다.
- 2) 사라지는 궤적 : 1)과정에서 대응점을 찾지 못한 $k-1$ 번째 프레임의 궤적에 포함된 특징점은 사라지는 궤적을 의미한다.
- 3) 새로운 궤적의 탐색 : $k-1$ 번째 프레임의 특징점 중 궤적에 포함돼 있지 않은 특징점에 대한 대응점을 k 번째 프레임에서 찾는다. 그 결과 찾아진 대응쌍은 새로운 궤적의 시작을 의미한다.

6. 계산의 복잡도

주어진 그래프가 $k-1, k$ 번째 프레임에 있는 m, n 개씩의 특징점들을 표현할 때, 두 프레임에 존재하는 대응쌍의 집합을 찾기 위한 계산량 $T_n(m, n)$ 을 최선의 경우와 최악의 경우로 제시한다.

i) 최악의 경우의 계산량(worst case time complexity)

한 개의 대응쌍을 결정하기 위한 최악의 계산량 $T_n(m, n)$ 은 두 프레임 중 특징점의 개수가 적은 프레임의 노드부터 그래프 탐색이 시작된다고 가정하면 다음과 같다.

- 1) $m = 1 \leq n$ 인 경우

$$T_n(m, n) = n + 1 \quad (11)$$

- 2) $m = 2 \leq n$ 인 경우

$$T_n(m, n) = n + 2 + n + 1 = 2n + 2 + 1 \quad (12)$$

- 3) $m = 3 \leq n$ 인 경우

$$T_n(m, n) = n + 3 + n + 2 + (n - 1) + 1 = 3n + 3 + 2 \quad (13)$$

- 4) $m = k \leq n$ 인 경우(단, $k \geq 2$)

$$\begin{aligned} T_n(m, n) &= n + k + n + (k-1) + (n-1) + (k-2) + \\ &\quad (n-2) + (k-3) + (n-3) + \dots + 3 + (n-(k-3)) + \\ &\quad 2 + (n-(k-2)) + 1 = kn + k + k - 1 = kn + 2k - 1 \end{aligned} \quad (14)$$

그러므로 대응쌍을 찾고자 하는 두 프레임에 각각 m, n 개의 특징점이 존재할 경우, 제안하는 알고리즘을

이용하여 한 개의 대응쌍을 찾기 위한 계산량은 다음과 같다.

$$T_n(m, n) = \begin{cases} n + 1 & \text{if } m = 1 \leq n \\ mn + 2m - 1 & \text{if } 2 \leq m \leq n \end{cases} \quad (15)$$

따라서 주어진 두 프레임에 존재하는 모든 특징점에 대한 대응쌍의 집합을 얻기 위한 계산량 $T_g(m, n)$ 은 식 16과 같이 쓸 수 있다.

$$\begin{aligned} T_g(m, n) &= T_n(m, n) + T_g(m-1, n-1) \\ &= T_n(m, n) + T_n(m-1, n-1) + \dots \\ &\quad + T_n(2, n-(m-2)) + T_n(1, n-(m-1)) \end{aligned} \quad (16)$$

식 15를 이용하여 식 16의 $T_n(\cdot, \cdot)$ 을 풀어쓰면 식 17, 18, 19, 20과 같다.

$$T_n(1, n-(m-1)) = n - (m-1) + 1 = n - m + 2 \quad (17)$$

$$T_n(2, n-(m-2)) = 2(n-(m-2)) + 2 \cdot 2 - 1 \quad (18)$$

$$T_n(m-1, n-1) = (m-1)(n-1) + 2(m-1) - 1 \quad (19)$$

$$T_n(m, n) = mn + 2m - 1 \quad (20)$$

따라서 두 프레임에 존재하는 모든 특징점에 대한 대응쌍의 집합을 얻기 위한 최악의 계산량 $T_g(m, n)$ 은 다음과 같다.

$$\begin{aligned} T_g(m, n) &= n - m + 2 + \sum_{i=2}^m (i(n-(m-i)) + 2i - 1) \\ &= n - m + 2 + n \sum_{i=2}^m i + m \sum_{i=2}^m i + \sum_{i=2}^m i^2 + 2 \sum_{i=2}^m i + \sum_{i=2}^m 1 \end{aligned} \quad (21)$$

$$\therefore T_g(m, n) = O(m^2 n) \quad \text{단, } 1 \leq m \leq n \quad (22)$$

ii) 최선의 경우의 계산량(best case time complexity)

한 개의 대응쌍의 결정하기 위한 최선의 경우의 계산량 $T_n(m, n)$ 은 두 프레임 중 특징점의 개수가 적은 프레임의 노드부터 그래프 탐색이 시작된다고 가정하면 식 23과 같다.

$$T_n(m, n) = m + n \quad \text{if } m \geq 1 \quad (23)$$

그러므로 주어진 두 프레임에 존재하는 모든 특징점에 대한 대응쌍의 집합을 얻기 위한 계산량 $T_g(m, n)$ 은 다음과 같다.

$$\begin{aligned}
 T_g(m, n) &= T_n(m, n) + T_g(m-1, n-1) \\
 &= (m+n) + ((m-1) + (n-1)) + \dots \\
 &\quad + (2+n-(m-2)) + (1+n-(m-1)) \\
 &= \sum_{k=0}^{m-1} (m+n-2(m-k)) = \sum_{k=0}^{m-1} (m+n-2k) \\
 &= m^2 + mn - (m-1)(m-2) = mn + 3m - 2 \quad (24) \\
 \therefore T_g(m, n) &= O(mn) \quad \text{단, } 1 \leq m \leq n \quad (25)
 \end{aligned}$$

즉, 그래프 탐색을 위한 계산량은 최선의 경우 $O(mn)$, 최악의 경우 $O(m^2n)$ 으로 주어진다.

III. 실험 결과

제안하는 알고리즘의 타당성을 입증하기 위하여 제안하는 알고리즘을 기존의 방법들과 비교 실험하였다.

이를 위하여 Pentium II MMX 300 MHz PC, Windows 98 환경에서 Delphi로 알고리즘을 구현하여 특징점이 주어졌을 경우와, 주어지지 않은 경우로 나누어 실험하였다.

기존의 방법 중 비교적 정확한 제적을 얻을 수 있는 방법으로는 Sethi 등의 방법^[7]과 Rangarajan 등의 방법^[9] 그리고 참고 문헌 [14]에서 제안된 방법 등이 있는데, 이들 방법은 각 프레임에 존재하는 특징점의 수가 항상 같으며, 잘못된 특징점이 존재하지 않는다는 가정을 필요로 한다. 본 논문에서 제안된 방법은 잡음이나 오류에 의한 잘못된 특징점을 고려하여 특징점의 대응관계를 결정하므로, 이 방법들에 비해 그 적용 범위가 훨씬 넓다. 한편, 본 논문에서 사용하는 정합의 척도는 참고문헌 [14]에서 사용한 것을 일부 수



그림 4. Ettliger-Tor sequence
Fig. 4. Ettliger-Tor sequence

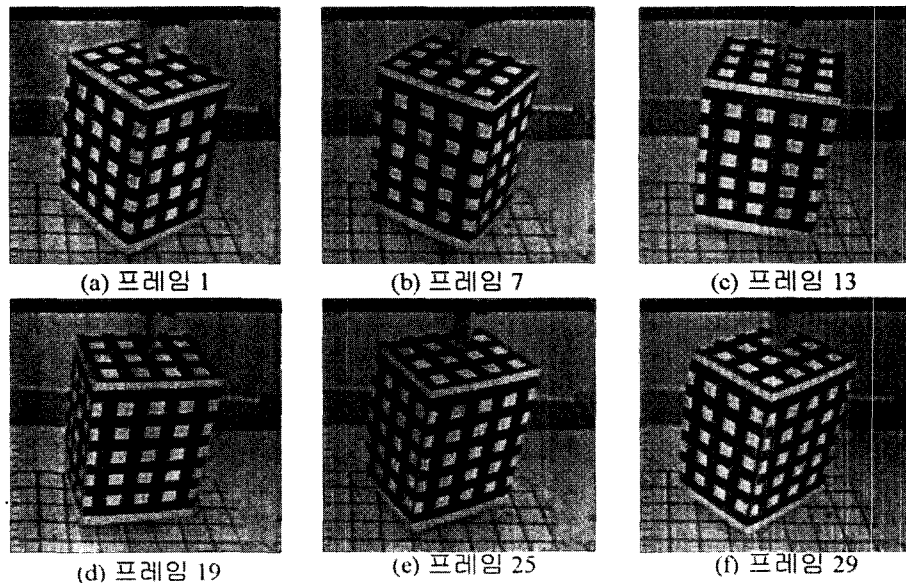


그림 5. Box sequence
Fig. 5. Box sequence.

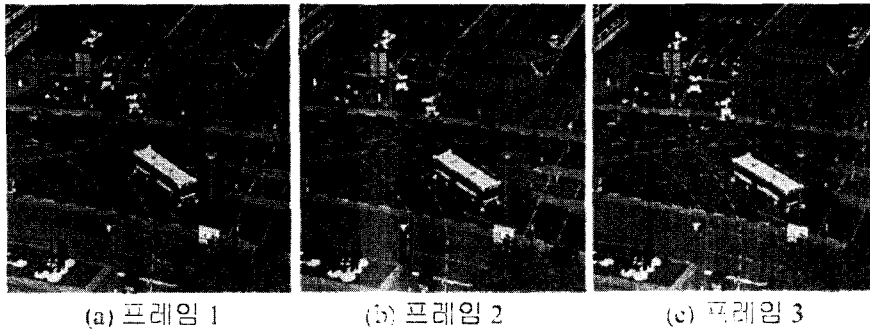


그림 6. Ettliger-Tor에 주어진 특징점
Fig. 6. Ettliger-Tor sequence with the given feature points.

정하여 사용하고 있기 때문에 각 프레임에 존재하는 특징점의 개수가 같고 특징점에 오류가 없을 경우에는 추출된 궤적의 정확도는 참고문헌 [14]와 같으며, Sethi 등의 방법, Rangarajan 등의 방법에 비해 더 정확한 결과를 보인다. 한편, 특징점이 주어지지 않은 경우 즉, 불완전한 궤적이 존재하는 경우에 대해서는 이들 방법을 적용할 수 없기 때문에 본 논문에서는 불완전한 궤적에 대한 고려가 충분한 Chetverikov 등의 방법^[13]을 제안된 방법과 비교·실험하였다. 실험에 사용한 영상은 256*256 해상도의 Ettliger-Tor sequence^[17]와 box sequence^[18]이다. 그림 4와 그림 5는 실험에 사용한 영상열들을 보이고 있다.

그림 6과 그림 7은 Ettliger-Tor sequence에 대해 특징점들이 주어진 경우에 대한 실험 결과를 보이고 있는데, 그림 6은 주어진 특징점들을 나타내고 있으며 그림 7은 주어진 특징점을 이용한 궤적의 추출 결과를 보이고 있다. 그림 7(a)는 제안하는 방법에 의한 궤적인데, 주어진 특징점에 대해 정확하게 궤적을 찾아내는 것을 볼 수 있다. 그림 7(b)는 Chetverikov

등의 방법을 Ettliger-Tor sequence의 처음 4프레임에 대해 적용한 것인데 궤적에 오류가 존재함을 볼 수 있으며, 이 오류는 이후의 프레임에 대한 특징점을 추적할 때 직접적인 영향을 끼치기 때문에 이후의 특징점 추적은 사실상 힘들게 된다. 그림 8과 그림 9는 box sequence의 처음 3프레임에 대해 특징점들이 주어진 경우에 대한 실험 결과를 보이고 있는데, 그림 8은 주어진 특징점들을 나타내고 있으며 그림 9는 주어진 특징점을 이용한 궤적의 추출 결과를 보이고 있다. 이 경우에도 Chetverikov 등의 방법에 의한 궤적에 오류가 존재하는 것을 볼 수 있다.

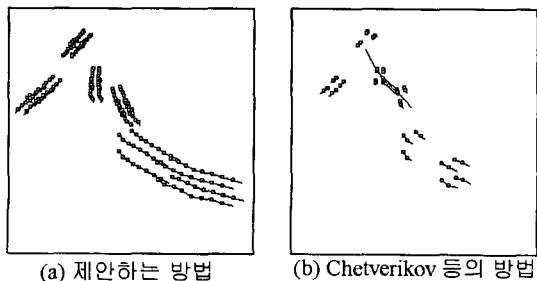


그림 7. Ettliger-Tor sequence에서 궤적 추출 결과
Fig. 7. Extracted trajectories in the Ettliger-Tor sequence.

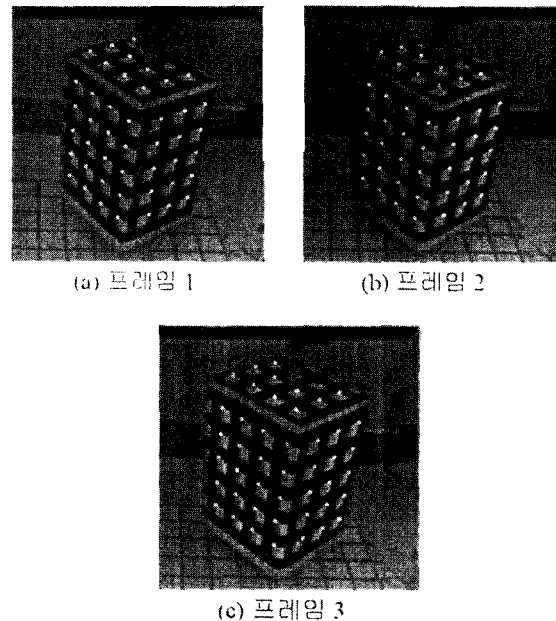


그림 8. Box sequence에 주어진 특징점
Fig. 8. Box sequence with the given feature points.

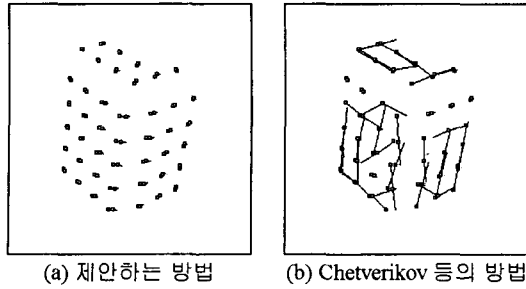


그림 9. Ettliger-Tor sequence에서 궤적 추출 결과
 Fig. 9. Extracted trajectories in the Ettliger-Tor sequence.

그림 10-12는 특징점이 주어지지 않은 경우에 대한 실험 결과이다. 그림 10은 box sequence의 앞부분을 구성하는 8개의 프레임에 대해서 II-2절에서 설명한 방법으로 코너를 추출한 결과를 보이고 있는데, 그림 10(b)-(h)는 인접한 프레임 사이의 차영상을 이용하여 움직임이 있다고 생각되는 곳에 대해서만 특징점 추출 알고리즘을 적용한 것이다. 그림 10(a)는 영상열의 첫 번째 프레임이기 때문에 움직임이 없는 영역의 코너도 나타남을 볼 수 있다. 그림 10의 각 프레임에 존재하는 특징점은 다수의 잘못된 특징점(false positive,

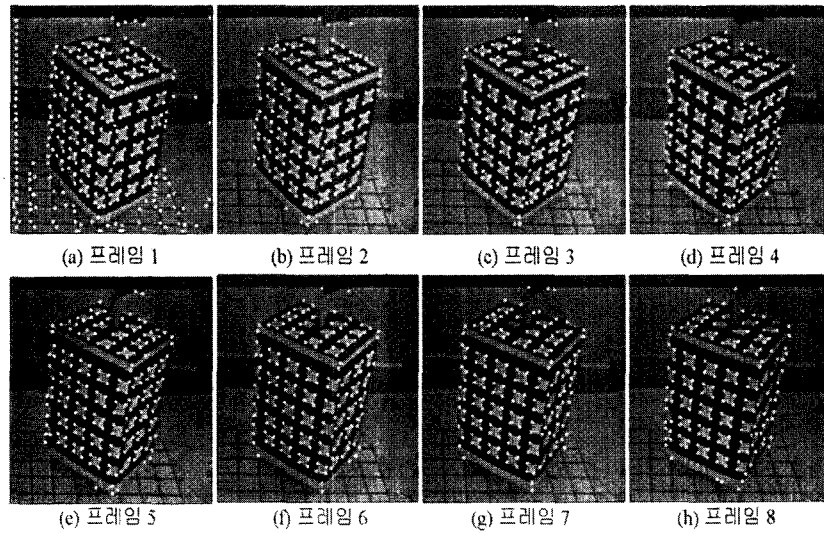


그림 10. Box sequence에 대한 특징점 추출 결과
 Fig. 10. Extracted feature points in the box sequence.

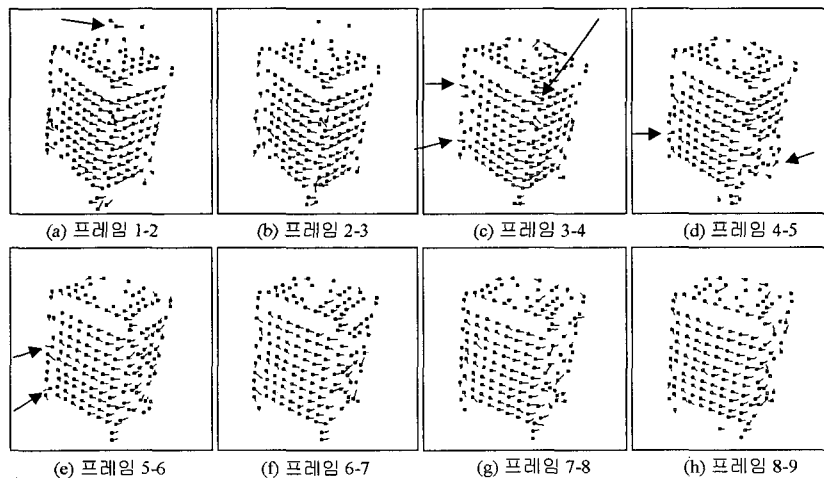


그림 11. 인접한 프레임의 특징점에 대한 대응관계
 Fig. 11. The correspondences of the feature points in the consecutive frames.

false negative)들을 포함하고 있는데, 이러한 잘못된 특징점들은 대응점 결정에 심각한 영향을 줄 수 있다.

그림 11은 그림 10에서 얻은 특징점들에 대하여 제안된 방법에 의해 결정된 특징점의 대응쌍을 보이고 있다. 그림 11(a), (c), (d), (e)의 화살표 표시가 되어 있는 곳은 대응점이 잘못 결정된 것들 중 일부를 보이고 있는데, 주목할 것은 현재 프레임의 잘못된 대응쌍이 다음 프레임으로 전파되지 않는다는 것이다. 이는 본 논문에서 사용한 특징점 정합 척도가 특징점의 운동 특성을 반영할 때 이전 프레임의 정합 결과는 현재 프레임의 정합 수행시 초기 가중치만 결정할 뿐이며 이후의 반복적인 연산에 의해 현재 프레임의 최적 가중치가 결정되므로 한 프레임에서의 오류가 이후의 프레임에 영향을 끼치지 않기 때문이다. 그러나 기존의 방법^{[7] [13]}에서는 이전 프레임의 정합 결과를 이용하여 현재 프레임의 대응점을 결정하기 때문에 이전 프레임의 정합 결과에 오류가 존재한다면 이후의 대응점 탐색에 그 오류는 직접적인 영향을 주게 된다.

그림 12는 box sequence의 궤적을 프레임별로 한 평면에 나타내었다. 그림 12(a)의 화살표로 표시된 것 중 "┐"은 잘못된 궤적이고 "┌"은 정확히 추출된 궤적이다. 이후의 궤적 추적 과정에서 잘못된 궤적은 더 이상 궤적의 확장이 일어나지 않고 소멸되는 것을 볼 수 있으며 올바른 궤적은 이후의 궤적이 정상적으로 확장, 추적되는 것을 볼 수 있다. 그림 12(c)에서

화살표 "┌"으로 표시되어 있는 궤적은 그림 12(b)에 존재하지 않았던 궤적이 새로 나타난 것이기 때문에 새로운 궤적이 나타남을 의미하며 이 궤적은 그림 12(d)에서도 확장, 추적된다. 그림 12(e)에서 "┌"로 표시된 화살표는 사라지는 궤적으로서, 그림 12(d)까지 계속 확장되던 궤적이 그림 12(f)이후에서는 확장이 되지 않음을 볼 수 있다.

IV. 결론

본 논문에서는 특징점의 운동 특성을 고려하는 정합 척도와 이를 이용하여 불완전한 궤적을 고려하여 특징점을 강건하게 추적할 수 있는 알고리즘 제안하였다. 정합 척도는 가중치가 부여된 유클리디언 거리를 사용하였는데, 가중치를 특징점의 운동특성에 따라 적응적으로 조절하였다.

영상열의 중간에 특징점의 나타남, 사라짐 등을 고려하기 위하여 불완전한 궤적을 고려하는 강건한 특징점 추적 알고리즘 제안하였는데 서로 인접한 프레임의 특징점들을 가중치가 있는 그래프로 표현한 다음, 두 프레임의 특징점간의 대응쌍 결정 문제를 그래프의 최소 가중치 탐색문제로 해결하였다. 제안하는 방법은 인접한 두 프레임에 m, n 개의 특징점이 존재할 경우, 최선의 경우 $O(mn)$, 최악의 경우 $O(m^2n)$ 의 계산량을 필요로 하며, 실제 영상열에 적용해 본 결과 제안하는

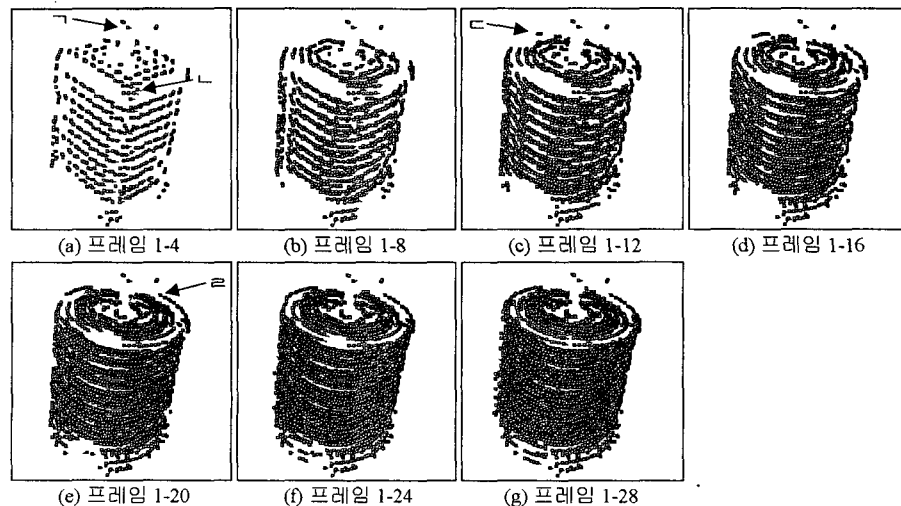


그림 12. Box sequence의 특징점 궤적
Fig. 12. The trajectories of the feature points in the box sequence.

방법이 효과적으로 특징점을 추적함을 확인할 수 있었다. 향후연구과제로는 보다 개선된 특징점 추출 알고리즘의 개발이 필요하며, 제안하는 방법을 비디오 검색 등에 적용하기 위하여 보다 복잡한 자연 영상에 대한 실험이 요구된다.

참 고 문 헌

- [1] Alexander Strehl, J. K. Aggarwal, "A New Bayesian Relaxation Framework for the Estimation and Segmentation of Multiple Motions," *Proc. of 4th IEEE Southwest Symposium on Image Analysis and Interpretation*, pp. 21-25, 2000.
- [2] E. Memem, P. Perez, "Dense Estimation and Object-Based Segmentation of the Optical Flow with Robust Techniques," *IEEE Trans. Image Processing*, vol. 7, no. 5, pp. 703-719, 1998.
- [3] A. Giachetti, M. Campani, V. Torre, "The Use Optical Flow for Road Navigation," *IEEE Trans. on Robotics & Automation*, vol. 14, no. 1, pp. 34-48, 1998.
- [4] J. Weber, J. Malik, "Rigid Body Segmentation and Shape Description from Dense Optical Flow under Weak Perspective," *IEEE Trans. on PAMI.*, vol. 19, no. 2, pp. 139-143, 1997.
- [5] Shoichi Araki, Takashi Matsuoka, Haruo Takemura, Naokazu Yokoya, "Real-time Tracking of Multiple Moving Objects in Moving Camera Image Sequences Using Robust Statistics," *Proc. ICPR.*, vol. 2, pp. 1433-1435, 1998.
- [6] N. Peterfreund, "Robust Tracking of Position and Velocity with Kalman Snakes," *IEEE Trans. PAMI.*, vol. 21, no. 6, pp. 564-569, 1999.
- [7] I. K. Sethi and R. Jain, "Finding Trajectories of Feature Points in an Monocular Image Sequence," *IEEE Trans. PAMI.*, vol. 9, no. 1, pp. 56-73, 1987.
- [8] V. Salari and I. K. Sethi, "Feature Point Correspondence in the Present of Occlusion," *IEEE Trans. PAMI.*, vol. 12, no. 1, pp. 87-91, 1990.
- [9] K. Rangarajan and M. Shah, "Establishing Motion Correspondence," *CVGIP: Image Understanding*, vol. 54, no. 1, pp. 56-73, 1991.
- [10] S. Krishnan and D. Raviv, "2D Feature Tracking Algorithm for Motion Analysis," *Pattern Recognition*, vol. 28, no. 8, pp. 1103-1126, 1995.
- [11] 정재영, 김문현, "다중 물체가 움직이는 동영상에서 자동적인 특징점 추출 및 추적," *정보과학회 논문지(B)*, 제25권, 제3호, pp. 562-573, 1998
- [12] R. Mehrotra, "Establishing Motion-based Feature Point Correspondence," *Pattern Recognition*, vol. 31, no. 1, pp. 23-30, 1998.
- [13] D. Chetverikov and J. Verestoy, "Tracking Feature Points : A New Algorithm," *Proc. ICPR 98*, pp. 1436-1438, 1998.
- [14] 정중면, 문영식, "적응적 가중치에 의한 특징점 추적 알고리즘," *전자공학회논문지*, 제36권 S편, 제11호, pp. 68-78, 1999
- [15] D. P. Kottke and Y. Sun, "Motion Estimation Via Cluster Matching," *IEEE Trans. PAMI.*, vol. 16, no. 11, pp. 1128-1132, 1994.
- [16] Simon Even, *Graph Algorithms*, Computer Science Press, 1979.
- [17] http://i21www.ira.uka.de/image_sequences/
- [18] <http://periscope.cs.umass.edu/~vislib/Motion/box/images.html>

 저 자 소 개

丁 鍾 勉(正會員)

1992년 한양대학교 전자계산학과 졸업(학사). 1994년 한양대학교 전자계산학과 졸업(석사). 1995년-현재 한양대학교 전자계산학과 박사과정. 1996년 3월 2000년 2월 한양대학교 컴퓨터 교육위원회 강사. 관심분야 : 컴퓨터비전, 패턴인식, 멀티미디어 응용. 전자공학논문지 제33권 B편 제5호 참조

文 泳 植(正會員)

1980년 서울대학교 전자공학과 졸업(학사). 1982년 한국과학기술원 전기 및 전자공학과 졸업(석사). 1990년 The University of California at Irvine 전기 및 컴퓨터공학과 졸업(박사). 1982년-1985년 한국전자통신연구소 연구원. 1989년-1990년 InnoVision Medical (미국) 선임 연구원. 1990년-1992년 생산기술연구원 선임연구원. 1992년-현재 한양대학교 전자계산학과 부교수. 관심분야 : 컴퓨터비전, 패턴인식, 멀티미디어 응용. 전자공학논문지 제33권 B편 제5호 참조