論文99-36S-11-9

적응적 가중치에 의한 특징점 추적 알고리즘

(A Feature Tracking Algorithm Using Adaptive Weight Adjustment)

丁 鍾 勉 *, 文 泳 植 *

(Jong-Myeon Jeong and Young-Shik Moon)

요 약

본 논문에서는 동영상에서 특징점의 궤적을 추적하기 위한 알고리즘을 제안한다. 기존의 방법에서 사용된 대부분의 정합의 척도(matching measure)는 동영상의 움직임 특성을 정확히 반영하지 못하여 잘못된 궤적을 나타내는 경우가 있다. 본 논문에서는 특징점의 공간좌표, 이동방향과 이동거리 등 3가지 속성을 정합에 사용하는데 이들 속성에 대하여 가중치(weight)가 부여된 Euclidean 거리를 정합의 척도로 사용한다. 이때 3가지 속성에 대한 가중치를 움직임의 특성에 따라 적응적으로 변화시켜 줌으로써 강건하게 특징점을 추적할 수 있도록 한다. 제안하는 알고리즘은 매 프레임마다 특징점의 운동특성을 정확히 반영함으로써 기존의 방법에 비해 정확한 궤적을 찾을 수 있으며 이는 다양한 동영상에 대한 실험을 통해 확인되었다.

Abstract

A new algorithm for tracking feature points in an image sequence is presented. Most existing feature tracking algorithms often produce false trajectories, because the matching measures do not precisely reflect motion characteristics.

In this paper, three attributes including spatial coordinate, motion direction and motion magnitude are used to calculate the feature point correspondence. 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 of the attributes are updated reflecting the motion characteristics, so that the robust tracking of feature points is achieved. The proposed algorithm can find the trajectories correctly which has been shown by experimental results.

Ⅰ. 서 론

순차적으로 입력되는 영상열에서 특징점(feature points)의 대응점(correspondence)을 찾는 문제는 물체 추적, 자동 운항(auto-navigation), 로봇 공학, 비디오 인덱싱 등 컴퓨터 비전의 많은 분야에서 중요한 부분을 차지한다. 입력되는 영상열에서 추출한 특징점은 연속적인 프레임 사이의 대응점을 찾음으로써 그 궤적을 알 수 있고, 특징점의 궤적은 전체 영상열에서 이동하는 물체들의 움직임을 나타낼 수 있다. 그러나 특징점 간의 대응점을 찾는 문제는 프레임의 수가 증가하고 프레임내의 특징점의 수가 늘어나게 되면 대응 가능한 경우의 수가 기하급수적으로 증가하기 때문에 해결하기 어려운 문제로 알려져 있다[1]-[5]. 따라서 대부분의 경우에는 실제 동영상에서 나타날 수 있는 일반적인 경우를 고려하여 움직임에 대한 제약을 둔 정합 척도를 정의하여 대응점을 찾거나 가능한 대응점의 수를 줄이게 된다.

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

接受日字:1999年5月28日, 수정완료일:1999年8月30日

Sethi[1] 등은 인접한 특징점 사이의 대응관계를 알기 위해 특징점의 이동거리와 이동방향의 일관성을 고려한 경로 일관성 함수(path coherence function)를 정의하여 정합의 척도로 사용하였으며 주어진 영상열에 존재할 수 있는 궤적들을 greedy exchange 알고리즘을 이용하여 최적화 문제로 해결하였다. 그러나 경로 일관성 함수의 이동방향과 속도에 대한 가중치에 따라 궤적이 크게 달라지기 때문에 특징점의 운동 특성을 잘 반영할 수 있도록 가중치를 적절히 설정해야 올바른 궤적을 얻을 수 있으며 이를 위해서는 특징점의 운동 특성을 사전에 알아야 정확한 궤적을 찾을 수 있다는 단점이 있다. Rangarajan[2] 등은 인접 균일성 함수(proximal uniformity function)를 정의하여 인접한 프레임의 특징점 간의 정합 척도로 사용하였는데 인접한 프레임 사이의 대응 가능한 모든 대응쌍을 이용한 상대적 비율을 이용하였다. 그러나, 실제로 정합되지 않은 특징점들이 정합 척도에 영향을 줌으로써 오류가 발생할 수 있는 단점이 있다. 한편, Mehrotra[3]는 특징점의 운동 방향과 속도를 고려한 움직임 균일성 함수(motion uniformity function)를 정합의 척도로 사용하고, 궤적의 확장성을 검사하여 특징점의 궤적을 얻었다. 이 방법은 프레임의 중간에서 궤적이 시작되거나 끝날 수 있지만, 운동의 방향과 속도에 대한 가중치에 따라 다른 궤적이 나타날 수 있다는 단점이 있다. Chetverikov[4] 등은 Sethi 등이 사용했던 정합의 척도를 사용하였는데, 특징점의 최대 이동 거리를 사전에 알고 있다고 가정한 후, 최대 이동 거리를 검색 영역으로 하여 검색 영역 내의 모든 가능한 정합쌍에 대해 초기 가설(initial hypothesis)과 가설 검증(hypothesis test)을 거쳐 특징점의 궤적을 구하였다. 이 방법은 프레임 중간에서 특징점의 사라짐, 나타남, 가려짐 등 불완전한 궤적을 추적하는데 장점을 가지고 있지만 특징점의 이동방향과 속도에 대한 가중치에 따라 그 결과가 달라질 수 있으며, Sethi 등의 방법에 비해 추출된 궤적이 부정확하다. 또한, 이 방법은 검색 영역의 범위에 따라 그 결과가 달라지기도 한다. 특징점 추적을 위한 기존의 방법을 종합하여 살펴보면, 대부분의 방법들은 동영상마다 다르게 나타나는 특징점의 운동특성을 정합의 척도에 정확히 반영하지 못함으로써 인접한 프레임 사이의 대응점을 정확히 얻지 못하는 경우가 있고 그 결과, 잘못된 궤적을 추출하는 경우가 많다. 따라서 보다 정확한 궤적을 얻기 위해서는 특징점의 운동특성을 정합의 척도에 정확히 반영하기 위한 방안에 대한 연구가 필요하다고 할 수 있다.

한편, Kottke[6] 등은 각 프레임을 변형된 K-means 군집화(clustering) 알고리즘으로 공간상에서 군집화 한 후, 서로 인접한 프레임간의 영역 정합(cluster matching) 방법을 사용하여 동영상 분석에 적용하였다. 정합의 척도는 가중치가 있는 Euclidean 거리를 사용하였고 인접한 프레임에서 서로 정합된 영역의 중심 좌표를 이용하여 영역의 이동방향과 거리를 계산하였다. 이 방법은 최종 정합 결과를 얻기 위해, 정합되는 영역의 운동 특성에 맞게 가중치를 반복적으로 조절하여 Euclidean 거리가 최소가 되도록 가중치를 조절함으로써 영역의 운동 특징에 관계없이 최적의 정합쌍을 얻을 수 있다. 그러나 각 프레임을 모두 군집화하기 위한 계산량이 대단히 방대하고 K-means 군집화 알고리즘에서 최적의 K를 결정하는 것이 어렵다. 또한 물체의 이동에 의해 필연적으로 발생하는 배경 영역의 변화는 군집화와 정합과정에서 많은 문제를 야기한다.

본 논문에서는 동일한 개수의 특징점이 각각의 프레임에 주어진다고 가정하고 Kottke 등이 사용하였던 가중치 조절방법을 특징점의 궤적을 추적하는데 적용한다. 제안된 방법은 특징점의 운동 특성에 따라 자동으로 정합의 가중치를 조절하기 때문에 입력되는 영상열에 대한 사전 지식이 없이 궤적의 추적이 가능하고 기존의 방법에 비해 정확한 궤적을 찾을 수 있다. 본 논문의 Ⅱ장에서는 특징점 추적 기법의 일반적 문제에 관해 설명하고, Ⅲ장에서 제안하는 방법에 대해 설명한 후, Ⅳ장에서는 기존의 방법과 제안하는 방법을 비교 실험한 결과를 보이며, Ⅴ장에서 결론과 향후 연구과제를 제시한다.

Ⅱ. 특징점 추적

특징점의 대응점을 찾는 문제는 컴퓨터 비전의 여러 분야에서 적용할 수 있기 때문에 그동안 많은 연구가 이루어져 왔다. 스테레오 비전은 대응점을 찾는 문제의 한 예로서, 두 대의 카메라로부터 입력받은 두 영상에서 왼쪽 영상에 있는 특징점의 대응점을 오른쪽 영상에 있는 특징점에서 찾는 것인데, 두 영상간에는 서로 불일치(disparity)가 존재하기 때문에 각 특징점의 대응점을 찾는 것은 해결하기 쉽지 않은 문제이다[7]. 따라서 스테레오 비전에서는 에피폴라(epipolar) 제약 등을

이용해 대응점을 찾는 문제를 단순화시키게 된다. 연속적으로 입력되는 프레임에서 특징점을 추적하는 것은 인접한 프레임에 존재하는 특징점들 사이의 대응점을 찾는 문제로서 특징점의 수가 증가하고 프레임의 수가 증가함에 따라 대응 가능한 정합쌍의 수가 기하급수적으로 증가하기 때문에 문제 해결이 어려워진다. 따라서 스테레오 비전에서 에피폴라 제약을 이용하듯이 특징점 추적 알고리즘에서도 특징점의 대응점을 찾기 위해 실제 환경에서 일어날 수 있는 상황을 고려하여 제약 조건을 부가함으로써 문제를 해결한다.

동영상 분석에서 일반적으로 사용되고 있는 운동의 제약은 작은 속도 변화(small velocity change), 공통 운동(common motion), 강체 운동(rigidity), 일관된 정합(consistent match), 최대 속도(maximum velocity) 등으로서, 이러한 제약 조건을 이용하여 대응 가능한 정합쌍의 수를 크게 줄여 잘못된 정합을 수행할 가능성을 줄여준다[2][8]. 작은 속도 변화 제약은 특징점이 움직이는 방향과 속도가 인접한 프레임 - 즉, 짧은 시간 간격동안 -에서는 크게 변화하지 않는 것을 말하는 것으로서, 특징점의 대응점을 찾을 때 속도와 방향이 크게 변하는 대응쌍을 제외하는 것을 말한다. 따라서 작은 속도 변화 제약에 의해 얻어진 궤적은 특징점의 이동 속도와 방향이 크게 변하지 않기 때문에 부드러운 궤적을 나타내게 된다. 공통 운동 제약은 서로 가까이 있는 특징점의 운동 특성은 서로 비슷하다는 것을 말하는 것으로서 특징점의 대응점을 찾을 때 그 주위에 있는 특징점의 운동 방향과 속도와 유사하도록 대응점을 찾는 것을 말한다. 또, 강체 운동 제약은 임의의 프레임에서 두 특징점 사이의 거리는 인접한 프레임에서도 일정하게 유지된다는 것을 말하며, 일관된 정합은 한 특징점에 대응하는 인접한 프레임의 특징점은 유일하다는 제약이다. 최대 속도 제약은 특징점의 최대 속도를 사전에 알고 있다고 했을 때, 한 특징점의 대응점은 그 특징점으로부터 최대 속도 이내의 범위에 존재한다고 가정하는 것으로서, 대응 가능한 정합쌍의 수를 크게 줄여준다. 특징점의 대응쌍을 제약하는 이러한 제약 조건들은 실제 동영상에서 나타날 수 있는 특징점들의 운동 특성을 고려하여 특징점 대응 문제에 적용함으로써 잘못된 대응점을 찾을 가능성을 크게 줄여준다.

1. Sethi와 Jain의 방법[1]

Sethi 등은 특징점의 궤적을 구하기 위해 특징점의 운동 방향과 운동 속도를 고려하는 경로 일관성 함수(path coherence function)를 정의하여 정합의 척도로 사용했으며 특징점의 궤적을 구하는 문제를 전체 프레임에 존재하는 궤적들의 최적화 문제로 풀었다. 정합에 사용한 경로 일관성 함수는 식 1과 같다.

$$\Psi(X_{ik-1}, X_{ik}, X_{ik+1}) = w_1\left(1 - \frac{\overline{X_{ik-1}X_{ik}} \cdot \overline{X_{ik}X_{ik+1}}}{\|\overline{X_{ik-1}X_{ik}}\| \, \|\overline{X_{ik}X_{ik+1}}\|}\right) + w_2\left(1 - \frac{2[\|\overline{X_{ik-1}X_{ik}}\| \, \|\overline{X_{ik}X_{ik+1}}\|]^{1/2}}{\|\overline{X_{ik-1}X_{ik}}\| + \|\overline{X_{ik}X_{ik+1}}\|}\right) \quad (1)$$

여기에서 $\Psi(X_{ik-1}, X_{ik}, X_{ik+1})$는 k 번째 프레임의 i 번째 궤적의 경로 일관성을 나타내고 있으며 X_{ik}는 k 번째 프레임의 i 번째 특징점을 의미한다. "|| ||"와 "·" 연산자는 벡터의 크기와 내적(inner product)을 각각 의미한다. 또, w_1과 w_2는 움직임의 속도와 방향에 대한 가중치이다. 이 식의 두 항은 움직임의 속도와 움직임의 방향을 각각 고려하고 있는데, 특징점의 대응점을 찾기 위해서는 정확한 w_1과 w_2 값을 필요로 한다. 그러나 최적의 w_1과 w_2는 영상의 움직임 특성을 알아야 얻을 수 있을 뿐 아니라 특징점의 운동 방향과 운동 속도는 변화하는 것으로써 임의의 한 값으로 고정시킬 수 없다. 따라서 Sethi와 Jain의 방법은 일반적으로 정확한 궤적을 얻을 수 있지만 그림 1에서 보이는 바와 같이 운동 방향과 속도에 대한 가중치에 따라 그 결과가 달라지는 문제가 있다. 그림 1은 서로 반대 방향으로 이동하고 있는 두 개의 특징점을 나타내고 있는데 k-2, k-1 프레임의 검정색으로 채워진 원은 이미 찾아낸 궤적이고 k 번째 프레임의 원은 현재 처리 중인 프레임에 있는 점으로써 경로 일관성 함수를 계산하여 그 대응

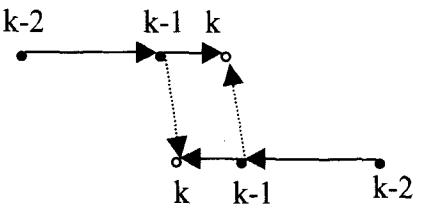


그림 1. 가중치에 따른 궤적의 변화
Fig. 1. Different trajectories with different weights in Sethi's method.

점을 결정하게 된다. 이때, 운동 방향 성분인 w_1을 0.1로, 이동 속도 성분인 w_2을 0.9로 주게 되면 점선에 의한 특징점을, 반대로 w_1을 0.9로, w_2을 0.1로 하면 실선에 의한 특징점을 대응점으로 결정하게 된다.

2. Rangarajan과 Shah의 방법[2]

Rangarajan 등은 식 2와 같이 특징점의 운동 방향과 운동 속도를 고려한 인접 균일성 함수(proximal uniformity function)를 정의하여 정합의 척도로 사용하였다.

$$\delta(X_p^{k-1}, X_q^k, X_r^{k+1}) = \frac{\left|\overline{X_p^{k-1}X_q^k} - \overline{X_q^kX_r^{k+1}}\right|}{\sum_{x=1}^{m}\sum_{z=1}^{m}\left|\overline{X_x^{k-1}X_{\phi^{k-1}(x)}^k} - \overline{X_{\phi^{k-1}(x)}^kX_z^{k+1}}\right|} + \frac{\left|\overline{X_q^kX_r^{k+1}}\right|}{\sum_{x=1}^{m}\sum_{z=1}^{m}\left|\overline{X_{\phi^{k-1}(x)}^kX_z^{k+1}}\right|} \quad (2)$$

여기에서 $\delta(X_p^{k-1}, X_q^k, X_r^{k+1})$는 k-1, k, k+1 번째 프레임에 각각 존재하는 p, q, r 번째 특징점이 이루는 궤적의 인접 균일성 함수를 말하며, X_q^k는 k 번째 프레임의 q번째 특징점, m은 프레임에 존재하는 특징점의 개수, 그리고 $X_{\phi^{k-1}(x)}^k$는 k-1 번째 프레임의 x 번째 특징점과 대응되는 k 번째 프레임의 특징점을 각각 의미한다.

식 2의 의미를 살펴보면, 앞에 있는 항목은 운동의 방향과 속도를 고려하는데 특징점이 완벽하게 일치되는 경우 이 값은 0을 갖는다. 뒤에 있는 항목은 운동의 크기를 고려하는데 가까운 거리에 있는 특징점일수록 적은 값을 갖도록 하여 정합의 우선 순위를 둔다. 이 식은 대응 가능한 모든 쌍을 이용한 상대적 비율을 계산함으로써 실제 정합되지 않은 특징들이 비율 계산에 영향을 끼치게 되고 그 결과 잘못된 정합이 생길 수 있다. 또한 인접한 프레임에서의 특징점은 가까운 거리에 존재한다는 가정은 일반적으로 타당하지만 특징점이 빠른 속도로 이동하는 경우에는 가까운 거리에 존재하는 정합쌍이 항상 옳다고 할 수는 없다. 그림 2는 운동 균일성 함수를 이용한 궤적의 문제점을 보이고 있는데 3개의 프레임에 각각 3개씩의 특징점이 있는 영상을 한 평면에 나타낸 것으로서 k-1, k 번째 프레임의 검게 채워진 원은 이미 찾아낸 궤적이고 k+1 번째 프레임의 원은 식 2에 의해 그 궤적을 결정해야 하는 특징점이다. k+1 번째 프레임에서 x 번째 궤적을 결정할 때, 특징점의 운동 방향을 고려하고 있는 식 2의 첫 번째 항은 k-1 번째 프레임과 k 번째 프레임에 의해 결정된 특징점의 모든 운동벡터와 k 번째 프레임과 k+1 번째 프레임과의 대응에 의해 만들어 질 수 있는 모든 운동 벡터의 차이의 합을 분모로 하고, x 번째 궤적의 k-1 번째 프레임과 k 번째 프레임에 의한 운동 벡터와 k 번째 프레임상의 x 번째 궤적의 특징점과 k+1 번째 프레임의 특징점에 의해 나타나는 운동벡터의 차이를 분자로 한다. 따라서 주어진 프레임 k+1에서 분모의 값은 상수가 되며 분자의 합에 의해 식 2의 첫 번째 항은 결정된다. 그림 2에서 k+1 번째 프레임의 특징점 중 가장 위에 있는 특징점은 두 운동 벡터의 차이가 가장 적기 때문에 x 번째 궤적에 포함될 가능성이 높으며 이는 적은 속도변화 제약을 고려했을 때 타당성이 있는 선택이다. 그러나 k+1 번째 프레임의 특징점 중 x 번째 궤적에 속하지 않는 나머지 특징점의 위치에 따라 분모의 값이 변하며, 특히 분모의 값이 커질 경우 - 예를 들면 k+1 번째 프레임의 특징점 중 가장 아래에 있는 특징점의 위치가 그림 2보다 오른쪽에 위치하는 경우 - 분자의 값은 상대적으로 작아져서 실제로 정합되는 특징점(가장 위에 있는 특징점)과 정합되지 않은 특징점에 대한 운동벡터의 변별력이 떨어지게 된다.

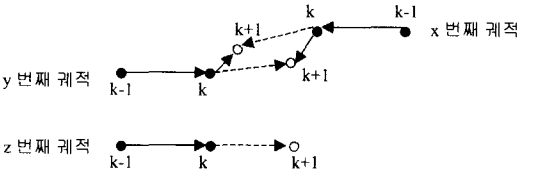


그림 2. Rangarajan 등의 방법에 의한 궤적의 변화
Fig. 2. Different trajectories in Rangarajan's method.

한편 그림 2의 k+1 번째 프레임에서 x 번째 궤적을 결정할 때, 특징점의 이동 거리를 고려하고 있는 식 2의 두 번째 항은 k 번째 프레임의 모든 특징점과 k+1 번째 프레임상의 모든 특징점에 의해 나타날 수 있는 대응쌍에 의한 운동벡터의 크기의 합을 분모로 하고, x 번째 궤적의 k 번째 프레임의 특징점과 k+1 번째 프레임의 특징점에 의해 나타나는 운동벡터의 크기를 분자로 한다. 따라서 프레임 k+1이 주어졌을 경우 분모의 값은 상수가 되며, k 번째 프레임의 x 번째 궤적의 특징점에 대응하는 점은 k+1 번째 프레임에서 분자의 크기가 가장 작은 중앙에 위치하는 특징점이 될 가능성이 높다. 그러나 첫 번째 항과 마찬가지로 k+1 번째 프레

임의 특징점 중 x 번째 궤적에 속하지 않는 나머지 특징점의 위치에 따라 분모의 값이 변하며 특히 분모의 값이 커질 경우 - 예를 들어 k+1 번째 프레임의 가장 아래에 있는 특징점의 위치가 훨씬 더 아래에 있는 경우 -에는 분자의 값이 상대적으로 작아져서 실제 정합되는 특징점과 정합되지 않은 특징점 사이의 변별력이 떨어지게 된다.

그림 2에서 k+1 번째 프레임의 특징점 중 x 번째 궤적에 포함되는 특징점을 결정할 때 점선에 의한 궤적이 작은 속도변화 제약을 가장 잘 만족하며 시각적으로도 가장 타당한 특징점으로 보이지만, k+1 번째 프레임에서 가장 아래에 있는 특징점이 오른쪽으로 이동할수록 식 2의 첫 번째 항의 분모가 커져서 방향성분의 변별력이 떨어지게 되고, 그 결과 실선에 의한 궤적을 만들어낼 가능성이 높아진다. 식 2의 이러한 문제는 실제로 정합되지 않은 특징점들이 정합 척도에 영향을 끼치게 때문에 발생한다.

3. 기존 방법의 문제점

Sethi 등이 사용한 경로 일관성 함수나 Mehrotra가 사용한 움직임 균일성 제약(motion uniformity constraint)[3] 등은 작은 속도 변화 제약의 범주에 포함되는 것인데, 이것은 물체들의 운동에서 일반적으로 나타날 수 있는 운동 특성을 가정하였기 때문에 부드러운 궤적을 얻을 수 있도록 하며, 잘못된 정합쌍을 찾을 가능성을 줄여준다. 한편, Chetverikov 등은 최대 속도 제약[4]을 사용하여 대응 가능한 정합쌍의 수를 줄였는데 Sethi 등이 사용하였던 경로 일관성 함수를 정합의 척도로 사용하였다. Rangarajan 등이 정의하여 사용한 인접 균일성 함수는 작은 속도 변화 제약과 더불어 임의의 특징점은 짧은 시간 동안에는 멀리 이동하지 않는다는 가정을 사용하였다. 그러나 물체의 운동 특성을 고려하는 이러한 개념을 구체적인 수식으로 표현하는 것, 즉 정합쌍을 구하기 위한 정합의 척도를 결정하는데 있어서 여러 가지 제약으로 움직임 특성을 반영할 수 있도록 하는 것은 개념적으로는 간단하지만, 구체적인 수식으로 이러한 운동특성을 나타내는 것은 대단히 어려운 일이다. 기존의 방법에서 정합을 위한 척도가 이러한 제약을 잘 반영하지 못함으로써 잘못된 궤적을 얻는 경우가 많이 발생함을 볼 수 있다.

본 논문에서는 작은 속도변화 제약과 일관된 정합이라는 제약을 사용하여 가중치가 부여된 Euclidean 거리를 정합의 척도로 사용한다. 이때 동영상마다 다르게 나타날 수 있는 움직임 특성을 반영할 수 있도록 가중치를 특징점의 운동 특성에 따라 자동으로 조정함으로써 기존의 방법들에 비해 정확한 궤적을 얻을 수 있는 방법을 제안한다.

Ⅲ. 제안하는 방법

1. 특징 벡터

N개의 특징점이 M개의 프레임에 각각 주어졌을 때 N개의 궤적을 찾는 문제는 인접한 프레임에서 특징점 사이의 대응점을 찾음으로써 해결할 수 있다. 이를 위해 다음과 같은 두 가지의 가정을 둔다.

1) 각 프레임의 특징점은 단 한 개의 궤적에만 포함된다(일관된 정합).
2) 인접한 프레임에서 특징점의 이동 방향과 이동 거리는 크게 변하지 않는다(작은 속도변화).

특징점 간의 대응점을 찾기 위해 사용한 특징점의 속성은 특징점의 공간좌표 (x, y), 이동거리 (d), 이동방향 (m) 등 3가지이다. 따라서 이동하는 특징점은 식 3과 같은 4차원의 벡터로 나타낼 수 있다.

$$X_k^i = \begin{bmatrix} x_k^i \\ y_k^i \\ d_k^i \\ m_k^i \end{bmatrix} \tag{3}$$

여기에서 X_k^i는 k 번째 프레임의 i 번째 특징점을 의미하며 (x_k^i, y_k^i)는 k 번째 프레임에 있는 i 번째 특징점의 공간 좌표를, d_k^i와 m_k^i는 k 번째 프레임에 있는 i 번째 특징점의 이동방향과 이동거리를 각각 의미한다.

2. 정합 척도

본 논문에서는 식 3과 같은 4차원 벡터들에 대해서 가중치가 부여된 Euclidean 거리를 정합의 척도로 사용하는데 이는 식 4와 같이 나타낼 수 있다.

$$D_k^{ij} = (X_{k-1}^i - X_k^j)\, W\, (X_{k-1}^i - X_k^j) \tag{4}$$

여기에서 j는 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 j 사이의 Euclidean 거리를

나타낸다. W는 식 5, 식 6과 같은 가중치 행렬로 주어진다.

$$W=\begin{bmatrix} w_x & 0 & 0 & 0 \\ 0 & w_y & 0 & 0 \\ 0 & 0 & w_d & 0 \\ 0 & 0 & 0 & w_m \end{bmatrix} \tag{5}$$

$$w_x w_y w_d w_m = 1 \tag{6}$$

만약, 특징점의 속성 중 하나가 다른 속성과 비교해서 변화가 크면 그 가중치를 작게 하여 정합 할 때 다른 속성과 균등히 Euclidean 거리에 영향을 주도록 균형을 맞춘다. 반대로 속성 중 하나가 인접한 프레임에서 매우 미세한 변화를 보인다면 그것에 대한 가중치를 크게 하며, 특징점의 속성이 연속되는 프레임에서 전혀 변하지 않으면 그 속성에 대응하는 가중치는 0이 되서 Euclidean 거리를 계산할 때 전혀 영향을 주지 않게 된다.

3. 특징점 정합

특징점 정합 과정은 두 단계로 이루어지는데, 첫 번째 단계에서는 식 4를 이용한 정합을 수행한다. k-1 번째 프레임의 i 번째 특징점에 대한 D_k^{ij}를 구하여 최소의 D_k^{ij}를 갖는 k 번째 프레임의 대응점을 구한다. 이때 가중치 행렬 W의 초기값은 이전 프레임의 최종 정합의 결과에 따라 결정되는데 이는 서로 인접한 프레임의 운동 특성은 크게 변하지 않기 때문에 이전 프레임에서 사용된 가중치는 대부분의 경우 현재 프레임에서 사용해도 큰 오차를 보이지 않기 때문이다. 식 4를 이용한 정합을 수행한 후, 두 번째 단계에서 현재 프레임의 정합 오류를 최소화하기 위해 가중치를 조절하게 된다.

두 번째 단계에서는 가중치 행렬 W의 값을 조정하는데 첫 번째 단계에서 얻은 정합쌍을 이용한다. 최적의 가중치 W는 식 7과 같이 모든 궤적에서의 D_k^{ij}의 값을 최소화 할 수 있어야 한다.

$$\min_{w_x, w_y, w_d, w_m} \sum_{i=1}^{N} D_k^{ij} \tag{7}$$

여기에서 N은 특징점의 수를 말한다. 식 7을 식 6의 가중치 제약 조건에 따라 Lagrange multiplier λ를 적용하면 다음과 같다.

$$D_k^T = \sum_{i=i}^{N} D_k^{ij} + \lambda(w_x w_y w_d w_m - 1) \tag{8}$$

여기에서 D_k^T는 k 번째 프레임 상의 모든 특징점의 D_k^{ij}의 합을 말한다. D_k^T를 최소화 하는 W를 얻기 위해 식 8을 w_x로 미분하면 다음과 같은 식을 얻을 수 있다.

$$\lambda = \frac{\sum_{i=1}^{N}(x_{k-1}^i - x_k^j)^2}{w_y w_d w_m} \tag{9}$$

유사한 방법으로 식 8을 w_y, w_d, w_m로 각각 미분하면 다음의 식들도 얻을 수 있다.

$$\lambda = \frac{\sum_{i=1}^{N}(y_{k-1}^i - y_k^j)^2}{w_x w_d w_m} \tag{10}$$

$$\lambda = \frac{\sum_{i=1}^{N}(d_{k-1}^i - d_k^j)^2}{w_x w_y w_m} \tag{11}$$

$$\lambda = \frac{\sum_{i=1}^{N}(m_{k-1}^i - m_k^j)^2}{w_x w_y w_d} \tag{12}$$

식 6, 9, 10, 11, 12를 풀면 다음과 같이 D_k^T를 최소화 하는 W를 얻을 수 있다.

$$w_x = [\sum_{i=1}^{N}(y_{k-1}^i - y_k^j)^2]^{1/4} [\sum_{i=1}^{N}(d_{k-1}^i - d_k^j)^2]^{1/4} [\sum_{i=1}^{N}(m_{k-1}^i - m_k^j)^2]^{1/4} [\sum_{i=1}^{N}(x_{k-1}^i - x_k^j)^2]^{-3/4} \tag{13}$$

$$w_y = \frac{w_x \sum_{i=1}^{N}(x_{k-1}^i - x_k^j)^2}{\sum_{i=1}^{N}(y_{k-1}^i - y_k^j)^2} \tag{14}$$

$$w_d = \frac{w_x \sum_{i=1}^{N}(x_{k-1}^i - x_k^j)^2}{\sum_{i=1}^{N} d_{k-1}^i - d_k^j)^2} \tag{15}$$

$$w_m = \frac{w_x \sum_{i=1}^{N}(x_{k-1}^i - x_k^j)^2}{\sum_{i=1}^{N}(m_{k-1}^i - m_k^j)^2} \tag{16}$$

첫 번째 단계와 두 번째 단계를 가중치의 변화가 없을 때까지 반복하면 D_k^T를 최소화하는 가중치 값에 수렴하게 되고, 수렴된 가중치에 의한 Euclidean 거리를 계산하여 그 값이 최소인 정합쌍을 최종 대응쌍으로 결정하게 된다.

입력 영상열에서 처음 두 개의 프레임에 대한 정합을 수행할 때 처음 반복에서는 수동으로 가중치 값을 결정해야 하는데, 첫 프레임은 이전 프레임과의 운동관

계가 없기 때문에 정합에 사용된 3가지 속성 중 공간 좌표만 이용하여 정합을 수행할 수 있도록 한다. 즉, 처음 두 프레임에 대한 정합을 수행할 때 가중치의 초기값은 식 17과 같으며 정합을 수행하면서 공간좌표에 대한 가중치가 자동으로 조정된다.

$$W=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&0\\0&0&0&0\end{bmatrix} \quad (17)$$

4. 특징점 추적 알고리즘

제안하는 특징점 추적 알고리즘을 요약하면 다음과 같다.

1 단계 : 궤적의 초기화(k=1, 2)

1.1 W의 초기값을 식 17과 같이 한다.

1.2 식 4를 이용하여 초기 정합쌍을 구하고 이를 이용해 D_k^T를 계산한다.

1.3 1.2에서 얻어진 결과에 따라 W의 값을 식 13, 14, 15, 16을 이용해 계산한다.

1.4 D_k^T가 이전 반복과 같으면 1.2에서 얻은 정합쌍을 최종 정합쌍으로 결정하고 반복을 멈춘다.

1.5 1.2 - 1.4 과정을 반복한다.

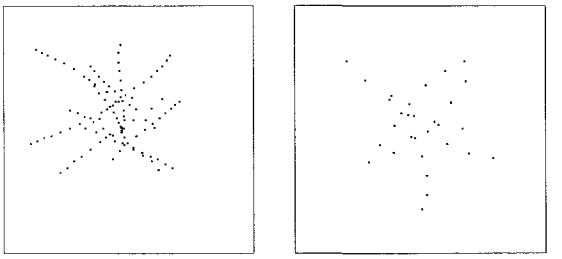
2 단계 : 나머지 프레임의 추적(k=3, 4, … , M)

2.1 식 4를 계산하여 초기 정합쌍을 구하고 이를 이용해 D_k^T를 계산한다.

2.2 2.1에서 얻어진 결과에 따라 W의 값을 식 13, 14, 15, 16을 이용해 계산한다.

2.3 D_k^T가 이전 반복과 같으면 2.1에서 얻은 정합쌍을 최종 정합쌍으로 결정하고 반복을 멈춘다.

2.4 2.1 - 2.3 과정을 반복한다.



(a) 영상열 A (b) 영상열 B

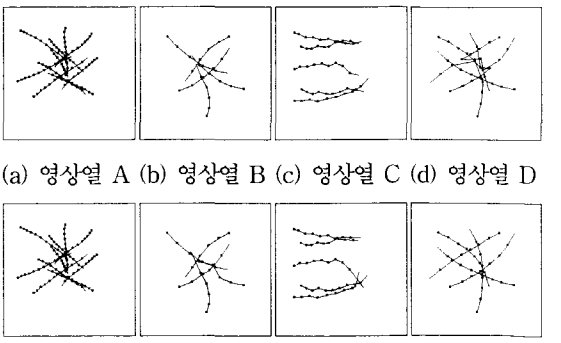
(c) 영상열 C (d) 영상열 D

그림 3. 입력 영상열
Fig. 3. Input sequence.

Ⅳ. 실험결과 및 성능 평가

제안하는 알고리즘의 타당성을 입증하기 위하여 본 논문에서 제안하는 방법과 Sethi 등의 방법, 그리고 Rangarajan 등의 방법을 서로 비교하였다. 이를 위하여 제안하는 알고리즘을 Pentium Ⅱ MMX 300 MHz PC, Windows 98 환경에서 Delphi로 구현하였다. 실험에 사용한 영상은 256*256 해상도로써 5-10개의 궤적이 존재하는 6-11개의 프레임으로 구성된 영상열이다. 그림 3은 실험에 사용한 영상열들을 보이고 있는데 영상열 A는 10개의 궤적이 존재하는 11개의 프레임을, 영상열 B는 5개의 궤적이 존재하는 6개의 프레임을, 영상열 C는 5개의 궤적이 존재하는 10개의 프레임을, 영상열 D는 5개의 궤적이 존재하는 8개의 프레임을 한 평면 위에 각각 나타낸 것이다.

그림 4는 Sethi 등의 방법에 대한 실험 결과를 보이고 있는데 (a)와 (a′), (b)와 (b′), (c)와 (c′) 그리고 (d)와 (d′)은 각각 그림 3의 영상열 A, 영상열 B, 영상열 C 그리고 영상열 D에 대한 궤적이다. 그림 4의 (a), (c), (d)는 w_1을 0.1로, w_2를 0.9로 준 결과인데 이 가중치는 Sethi 등이 그들의 논문에서 실험에 사용한 가

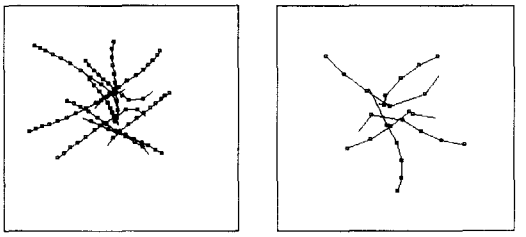


(a) 영상열 A (b) 영상열 B (c) 영상열 C (d) 영상열 D

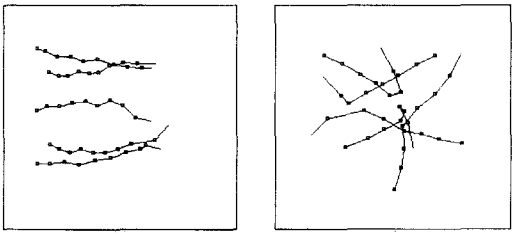
(a) 영상열 A (b) 영상열 B (c) 영상열 C (d) 영상열 D

그림 4. Sethi 등의 방법을 이용한 실험 결과
Fig. 4. Trajectory of Sethi's method.

중치이며, (a′), (b′), (c′), (d′)은 w_1을 0.9로 w_2를 0.1로 준 결과이다. 그림 4의 결과 중 (a), (a′), (b), (c), (d′)만이 올바른 궤적을 나타내고 있는데 이 실험 결과는 w_1, w_2의 값에 따라 Sethi 등의 방법의 성능이 크게 달라짐을 보여 주고 있다. 따라서 Sethi 등의 방법은 사전에 최적의 w_1, w_2의 값을 알아야 신뢰성 있는 특징점 추적이 가능하다. 그러나 영상열마다 특징점의 운동 특성이 다르기 때문에 최적의 w_1, w_2의 값도 다르며 동일 영상열 내에서도 매 프레임마다 특징점의 운동 특징이 변할 수 있기 때문에 사전에 최적의 w_1, w_2의 값을 아는 것은 사실상 어렵다고 할 수 있다.



(a) 영상열 A (b) 영상열 B

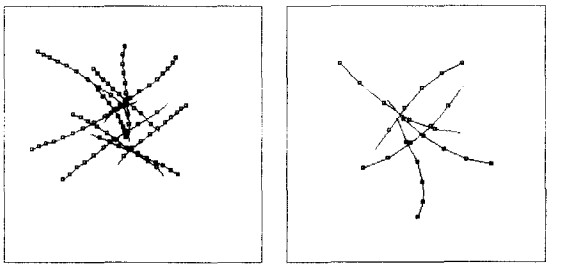


(c) 영상열 C (d) 영상열 D

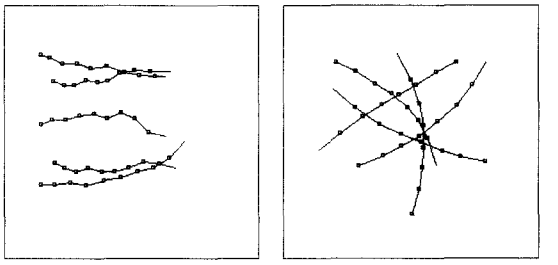
그림 5. Rangarajan 등의 방법을 이용한 실험 결과
Fig. 5. Trajectory of Rangarajan's metho.

그림 5는 Rangarajan 등의 방법에 대한 실험 결과인데 그림 5(a) 영상에서 중앙 우측 부분에 있는 궤적은 실제로 정합되지 않는 정합쌍이 정합에 영향을 끼치기 때문에 발생한 오류이며 그림 5(b), (c), (d)에도 특징점의 운동 특성을 정확히 반영하지 못하여 발생한 오류가 나타남을 볼 수 있다. 특히 그림 5(d)는 가까운 거리에 있는 정합쌍을 선택함으로써 발생한 오류가 특히 심각함을 보여 준다.

제안하는 방법을 이용한 실험 결과는 그림 6에 제시하였다. 그림 6은 그림 3의 영상열들에 대한 실험 결과인데, 제안하는 방법은 특징점 정합을 위한 가중치를 고정시키지 않고 특징점의 운동 특성에 따라 적응적으로 자동 조절함으로써 기존의 방법에서 발생했던 문제들이 해결되는 것을 볼 수 있다. 표 1, 2, 3, 4는 본 논



(a) 영상열 A (b) 영상열 B



(c) 영상열 C (d) 영상열 D

그림 6. 제안하는 방법을 이용한 실험 결과
Fig. 6. Trajectory of the proposed method.

문에서 비교한 알고리즘들과 제안하는 알고리즘의 성능을 정량적으로 나타내었는데, 실제 궤적과 각 알고리즘이 만들어 내는 궤적 사이의 Euclidean 거리를 각 영상열에 대한 프레임별로 나타내었다.

Sethi 등의 방법은 영상열 D에서 오류가 대단히 심각함을 볼 수 있는데, 정합을 수행할 때 Sethi 등이 사용하였던 가중치를 그대로 적용했기 때문이다. 만약 각 영상열의 운동 특성을 반영할 수 있는 최적의 가중치가 Sethi 등의 방법에 주어진다면 Sethi 등의 방법은 제안하는 알고리즘과 동일한 성능을 보일 수 있다.

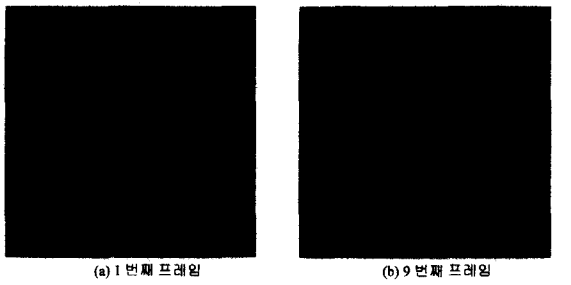
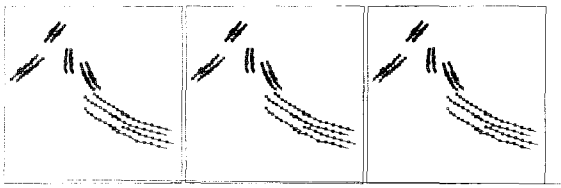


그림 7. Ettlinger-Tor 영상열
Fig. 7. Ettlinger-Tor sequence.

실제 영상열에 대한 실험 결과는 그림 7과 그림 8에 나타내었는데 입력 영상열은 Karlsruhe 대학에서 제공한 Ettlinger-Tor 영상열이고, 특징점은 주어지는 것으로 가정하였다. 실험에 사용한 특징점 추적 알고리즘들

은 적절한 가중치를 입력했을 경우 모두 올바른 궤적을 나타내었다. 그러나 Sethi 등의 방법은 가중치에 따라 정합 결과가 크게 달라지는 것을 실제 영상에서도 확인할 수 있었다.



(a) Sethi 등의 방법 (b) Rangarajan 등의 방법 (c) 제안하는 방법

그림 8. Ettlinger-Tor 영상열에 대한 실험 결과

Fig. 8. Experimental results for the Ettlinger-Tor sequence.

한편, 특징점 추적에서 영상열 중간에 정합의 오류가 발생하였을 때에 대한 고려도 필요하다. 기존의 방법들은 이전 프레임의 정합 결과를 다음 프레임의 정합에서 사용하기 때문에 특징점 추적을 수행하다 영상열 중간에 오류가 발생했을 경우에는 그 오류가 계속 누적되어 심각한 오류를 가진 궤적을 만들어 낸다. 따라서 기존의 방법들은 특징점 추적 중간에 오류가 발생했을 경우, 이후의 특징점 추적은 사실상 불가능하다. 그러나 제안하는 알고리즘에서는 이전 프레임의 정합 결과는 다음 프레임 정합에서 가중치 초기값만 결정하며 이후의 가중치 조정을 위한 반복 과정에서 현재 정합중인 프레임에 가장 적합한 가중치를 계산해 내기 때문에 이전 프레임에서 정합 오류가 발생하였다 하더라도 다음 프레임의 정합 결과에는 큰 영향을 끼치지 않으며, 가중치 조정 과정에서 이전 프레임의 오류를 최소화 할 수 있도록 현재 프레임에서 가장 적합한 가중치를 얻게 된다. 따라서 제안하는 알고리즘은 영상열 중간에 오류가 발생하더라도 이후의 영상열에 대한 특징점 추적을 계속 수행할 수 있다.

한편, 본 논문에서 실험에 사용한 영상열들에 대한 가중치는 대부분 3회 정도의 반복에서 최적의 가중치를 찾아냈으며, 4회의 반복만에 최적값을 찾는 경우도 1회 관찰 할 수 있었다.

2. 수행 시간의 비교

제안하는 알고리즘의 수행시간을 기존의 다른 알고리즘과 비교하였다. 제안하는 알고리즘은 매 프레임에서 대응점을 찾기 위해 가중치를 반복적으로 조정하고 Sethi 등이 제안한 방법은 전체 궤적을 찾기 위해 반복

표 1. 영상열 A에 대한 실제 궤적과 측정된 궤적 사이의 Euclidean 거리

Table 1. Euclidean distances between correct and estimated trajectories for sequence A.

	실제 궤적과 측정된 궤적 사이의 Euclidean 거리										
프레임 번호	1	2	3	4	5	6	7	8	9	10	11
Sethi 등의 방법	0	0	0	0	0	0	0	0	0	0	0
Rangarajan 등의 방법	0	0	0	0	0	0	0	0	0	41.3	143.1
제안하는 방법	0	0	0	0	0	0	0	0	0	0	0

표 2. 영상열 B에 대한 실제 궤적과 측정된 궤적 사이의 Euclidean 거리

Table 2. Euclidean distances between correct and estimated trajectories for sequence B.

	실제 궤적과 측정된 궤적 사이의 Euclidean 거리										
프레임 번호	1	2	3	4	5	6					
Sethi 등의 방법	0	0	0	0	0	0					
Rangarajan 등의 방법	0	0	0	0	88.4	210.8					
제안하는 방법	0	0	0	0	0	0					

표 3. 영상열 C에 대한 실제 궤적과 측정된 궤적 사이의 Euclidean 거리

Table 3. Euclidean distances between correct and estimated trajectories for sequence C.

	실제 궤적과 측정된 궤적 사이의 Euclidean 거리										
프레임 번호	1	2	3	4	5	6	7	8	9	10	
Sethi 등의 방법	0	0	0	0	0	0	10.2	14.1	14.1	14.1	
Rangarajan 등의 방법	0	0	0	0	0	0	10.2	14.1	37.5	58.8	
제안하는 방법	0	0	0	0	0	0	0	0	0	0	

표 4. 영상열 D에 대한 실제 궤적과 측정된 궤적 사이의 Euclidean 거리

Table 4. Euclidean distances between correct and estimated trajectories for sequence D.

	실제 궤적과 측정된 궤적 사이의 Euclidean 거리										
프레임 번호	1	2	3	4	5	6	7	8			
Sethi 등의 방법	0	0	0	12.6	48.2	198.6	309.6	456.6			
Rangarajan 등의 방법	0	0	0	0	29.1	94.9	252.7	460.9			
제안하는 방법	0	0	0	0	0	0	0	0			

표 5. 다양한 영상열에 대한 수행 시간의 비교

Table 5. Comparison of the execution time for various methods.

영 상 열	A	B	C	D	평균
Sethi 등의 방법(초)	0.60	0.33	0.49	0.43	0.463
Rangarajan 등의 방법(초)	0.49	0.28*	0.44	0.38*	0.398
제안하는 방법(초)	0.49	0.27	0.44	0.38	0.395

적인 연산을 필요로 한다. 반면 Rangarajan 등이 제안한 방법은 반복적인 연산은 수행하지 않는다. 각 영상열에 대해 파일에 대한 입출력을 포함하는 수행 시간을 1/1000초 단위로 20-30회씩 측정하여 그 중 최소의 수행시간을 표 5에 제시하였다. 표 5에서 "*" 표시는 해당 알고리즘이 만들어 내는 궤적의 오류가 심각한 경우를 의미한다. 제안하는 알고리즘은 Rangarajan 등의 방법과 거의 비슷한 수행시간을 필요로 하며 Sethi 등의 방법보다는 대체적으로 우수한 수행시간을 보임을 알 수 있다. 제안하는 알고리즘의 수행시간은 입력되는 영상열에 따라 크게 달라지는 문제 의존적(problem dependent)인 수행시간을 필요로 하기 때문에 입력 영상열에 따라 달라질 수 있다.

V. 결론 및 향후 연구과제

본 논문에서는 영상열에 주어진 특징점의 궤적을 추적할 수 있는 강건한 알고리즘을 제안하였다. 정합의 척도는 가중치가 부여된 Euclidean 거리를 사용하였는데, 이전 프레임에서 수렴된 가중치를 초기값으로 하여 정합하였고, 정합결과를 이용해 가중치를 반복적으로 조정하였다. 제안된 방법은 특징점 정합을 위한 가중치를 특징점의 운동 특성에 따라 적응적으로 조절함으로써 기존의 다른 방법보다 정확한 궤적을 얻었다.

향후 연구과제로는 프레임의 중간에 특징점의 사라짐, 나타남, 가려짐 등의 불완전한 궤적을 추적할 수 있는 알고리즘의 개발이 요구되며, 궤적 추적과정에서 특징점을 자동으로 찾는 알고리즘의 개발도 요구된다.

참 고 문 헌

[1] 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.

[2] K. Rangarajan and M. Shah, "Establishing Motion Correspondence," CVGIP : Image Understanding, Vol. 54, No. 1, pp. 56-73, 1991.

[3] R. Mehrotra, "Establishing Motion-based Feature Point Correspondence," Pattern Recognition, Vol. 31, No. 1, pp. 23-30, 1998.

[4] D. Chetverikov and J. Verestoy, "Tracking Feature Points : A New Algorithm," Proc. ICPR 98, pp. 1436-1438, 1998.

[5] 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.

[6] D. P. Kottke and Y. Sun, "Motion Estimation Via Cluster Matching," IEEE Trans. PAMI., Vol. 16, No. 11, pp. 1128-1132, 1994.

[7] R. M. Haralick, L. G. Shapiro, Computer and Robot Vision, Vol. II, Addison-Wesley Pub. Comp., 1993.

[8] D. H. Ballard, C. M. Brown, Computer Vision, Prentice-Hall, 1982.

저 자 소 개

丁 鍾 勉(正會員) 第 33卷 B編 第 5號 參照

1992년 한양대학교 전자계산학과 졸업(학사). 1994년 한양대학교 전자계산학과 졸업(석사). 1995년~현재 한양대학교 전자계산학과 박사과정. 관심분야 : 컴퓨터비전, 패턴인식, MPEG 4, MEPG 7

文 泳 植(正會員) 第 33卷 B編 第 5號 參照

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.