동작정보 추출을 위한 가변적 탐색 영역과 블록 크기의 정합

장 석 우[†] · 김 봉 근[††] · 김 계 영[†††] · 최 형 일[††††]

요 약

본 논문에서는 영상으로부터 블록 단위의 동작벡터를 추출하기 위해 가변적인 크기의 블록을 사용하는 정합 방법을 제안한다. 제안된 정합 알고리즘은 탐색영역의 크기를 적응적으로 조절하고, 보다 정확한 동작벡터를 추출할 때까지 블록의 크기를 동적으로 설정한다. 본 논문에서는 탐색영역의 중심위치를 결정하기 위해 시간의 경과에 따른 블록의 동작변화는 작다고 가정한다. 또한, 탐색영역의 범위는 공간적으로 인접한 블록들의 동작 균일성에 따라 조절된다. 블록의 크기를 결정하는 과정은 작은 크기의 블록에서부터 정합을 시작한다. 정합 정확도가 좋을 경우 블록의 크기를 조금 확대한 후 본 논문에서 정한 정합 기준이 만족될 때까지 정합 과정을 반복한다. 실험 결과는 제안한 가변적인 블록정합 방법이 보다 정확한 블록 단위의 동작벡터를 추출함을 보여준다.

Size-Variable Block Matching for Extracting Motion Information

Seok-Woo Jang[†] · Bong-Keun Kim[††] · Gye-Young Kim[†††] · Hyung-Il Choi[††††]

ABSTRACT

This paper proposes a size-variable block matching algorithm for motion vector extraction. The proposed algorithm dynamically determines the search area and the size of a block. We exploit the constraint of small velocity changes of a block along the time to determine the origin of the search area. The range of the search area is adjusted according to the motion coherency of spatially neighboring blocks. The process of determining the size of a block begins matching with a small block. If the matching degree is not good enough, we expand the size of a block a little bit and then repeat the matching process until our matching criterion is satisfied. The experimental results show that the proposed algorithm can yield very accurate block motion vectors. Our algorithm outperforms other algorithms in terms of the estimated motion vectors, though our algorithm requires some computational overhead.

키워드 : 동작벡터(Motion vector), 평가함수(Evaluation function), 동작정보(Motion information), 블록정합(Block matching algorithm)

1. 서 론

동작정보는 연속적으로 입력되는 영상 사이의 시·공간적인 연관성으로부터 추출할 수 있으며 비디오 압축 및 비디오 회의 등의 분야에서 많이 사용된다[1, 2]. 연속적인 동영상으로부터 중복된 부분을 찾아내는 일반적인 방법은 n-1번째 프레임을 이용하여 n번째 프레임을 예측하는 것이다. 일반적으로, n번째 프레임과 n-1번째 프레임은 현재 프레임과 이전 프레임으로 각각 불려진다.

영상으로부터 동작정보를 추출하기 위해 전통적으로 많이 사용하는 방법은 화소 단위의 동작벡터를 추출하는 광류(optical flow) 측정에 의한 방법과 블록 단위의 동작벡터를 추출하는 블록정합 방법이 있다. 화소 단위의 동작벡터를 추출하는 방법은 정확도가 높고 종류가 다양하지만 영상의 전체 범위에 걸쳐 화소 단위의 복잡한 연산을 수행하므로 병렬처리 환경이 아닐 경우 계산의 복잡도가 매우 높다. 따라서 계산의 복잡도를 줄인 블록 단위의 동작벡터를 추출하는 블록정합 방법을 많이 사용한다.

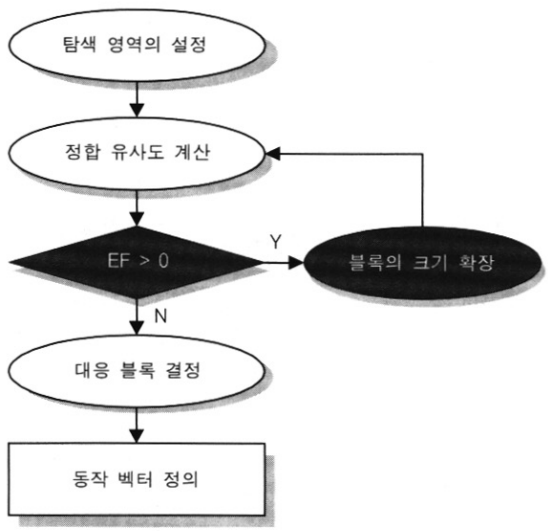
블록 단위의 동작 측정 기법은 현재 많은 분야에서 사용된다[3]. 이 기법에서는 현재 프레임을 모양과 크기가 동일한 사각형의 블록으로 분할한 후 블록 단위의 특징을 정의한다. 그리고 정의된 특징에 의해 구성되는 정합 척도를 이전 프레임의 일정한 탐색영역 내의 블록들에 적용하여 가장 높은 정합 척도를 가지는 블록을 찾는다. 정합된 두 블록 사이의 상대적인 위치는 현재의 블록과 연관된 동작벡터를 정의하고, 동작벡터들의 전체 집합은 동작 필드를 정의한다.

동작벡터 측정을 위한 많은 블록정합 기법들이 개발되었으며, 많은 문헌에서 그 성능이 평가되었다. 블록 단위의 동

† 정 회 원 : 숭실대학교 생산기술연구소 연구원
†† 정 회 원 : 청주과학대학 컴퓨터학과 교수
††† 종신회원 : 숭실대학교 컴퓨터학과 교수
†††† 정 회 원 : 숭실대학교 미디어학부 교수
논문접수 : 2002년 7월 15일, 심사완료 : 2003년 6월 3일

작벡터를 추출하기 위해 일반적으로 사용되는 알고리즘은 전역 탐색 블록정합 알고리즘(FSBMA : full search BMA)이다[4]. 이 알고리즘은 영상을 사각형의 블록으로 분할한 후 정합 척도를 탐색영역 내의 블록들에 적용하여 정합 유사도가 가장 높은 블록을 찾는다. 그러나 전역 탐색 블록정합 알고리즘은 탐색영역의 내부 전체를 탐색하면서 가능한 모든 블록들에 대한 정합을 수행하므로 처리 시간이 오래 걸린다. 따라서 처리 시간을 줄이기 위해 3단계 탐색에 의한 방법(TSS : three step search), 새로운 3단계 탐색에 의한 방법(NTSS : new three step search), 계층적인 1차원 탐색에 의한 방법, 4단계 탐색에 의한 방법(FSS : four step search) 등이 제안되었다[5-8]. 이들 대부분의 기존의 방법들은 후보 블록이 탐색될 탐색영역을 정의하는 방법에 초점을 두고 있다. 그러나, 블록의 크기를 적당하게 선택하는 것도 블록정합의 정확도를 결정하는 데 큰 역할을 수행한다. 일반적으로, 큰 크기의 블록들은 조밀하지는 않지만 강건적인 동작의 예측에 적당하고, 작의 크기의 블록들은 지역적인 예측에 적당하지만 잡음에 민감하다.

본 논문에서는 블록 단위의 동작 벡터를 추출하기 위해 탐색 영역과 블록의 크기를 가변적으로 조절하는 블록 정합 알고리즘을 제안한다. 제안된 블록 정합에서 사용되는 탐색 영역의 중심 위치는 블록 정합의 초기에 분할된 블록의 중심 위치와 이전 시점의 블록에서 추출한 동작 벡터를 이용하여 정의되며, 탐색 영역의 크기는 이전 시점에서 추출한 동작 벡터들과 이들의 신뢰도를 기반으로 설정된다. 그리고 탐색 영역의 크기는 동작 벡터의 신뢰도에 따라 연속적으로 변화하며, 동작 벡터의 신뢰도는 탐색 영역의 크기를 동적으로 변화시키는 가중치의 역할을 하도록 정의된다. 블록의 크기를 결정하는 과정은 작은 크기의 블록 정합으로부터 시작한다. 정합 유사도가 좋지 않을 경우 블록의 크기를 조금 확대한 후 본 논문에서 정한 정합 기준이 만족될 때까지 정합 과정을 반복한다.



(그림 1) 가변적 크기의 블록정합의 전체 구조

(그림 1)은 제안된 정합 알고리즘의 전체적인 구조를 보여준다. 그림에서 보는 바와 같이 블록정합의 정합 유사도는 현재 크기의 블록으로 탐색영역의 후보 블록들에 대해 계산된다. 평가 함수는 정합 유사도를 기반으로 블록의 크기의 적합성을 판단한다. 평가 함수 EF가 양의 값을 갖는 것은 블록의 크기를 확대해야 한다는 것을 의미한다. 그렇지 않을 경우, 현재의 블록의 크기가 적당하다고 판단하고 대응하는 블록으로부터 동작벡터를 계산한다.

본 논문의 전체 구성은 다음과 같다. 1장에서는 연구 동기와 가변적 크기의 블록정합 알고리즘의 전체적인 개요를 설명하였다. 2장에서는 탐색영역을 동적으로 설정하는 방법에 대해 설명하고, 3장에서는 정합 척도로 사용되는 정합 유사 함수를 정의하는 방법 및 정합 정도를 판단하는 평가 함수를 기반으로 블록의 크기의 적합성을 결정하는 방법에 대해 설명한다. 4장에서는 제안된 블록정합 방법의 성능을 비교하는 실험 결과에 대해 설명하며, 5장에서는 결론을 기술한다.

2. 탐색영역 설정

본 장에서는 블록정합에서 사용되는 탐색영역을 동적으로 설정하는 방법에 대해 설명한다. 본 논문에서는 고정적인 탐색영역을 사용하는 기존의 블록정합 방법과는 달리 이전 시점에서 구한 동작벡터들과 이들의 신뢰도를 측정하여 탐색영역의 중심 위치와 크기를 동적으로 변화시킨다.

블록정합은 현 시점 영상의 해당 블록과 가장 유사한 후보 블록을 이전 시점 영상의 탐색영역 내에서 찾는 과정이다. 일반적으로, 블록정합에서 사용하는 탐색영역 SA(Search Area)를 중심 위치와 크기를 나타내는 벡터로 표현하면 식 (1)과 같다.

$$\begin{aligned} \boldsymbol{SA} &= (\boldsymbol{p}(x,y),\ \boldsymbol{s}(x,y)) = ((x,y),(xs,ys)) \\ &= (x,y,xs,ys) \end{aligned} \tag{1}$$

여기서, $\boldsymbol{p}(x,y)$는 탐색영역의 중심 위치를 나타내는 벡터이고, $\boldsymbol{s}(x,y)$는 탐색영역의 크기, 즉 가로와 세로의 길이를 나타내는 벡터이다. 본 논문에서는 식 (1)과 같은 표현법을 기반으로 t 시점의 i번째 블록 B_i의 블록정합시 적용하게 될 탐색영역 $\boldsymbol{SA}_t(B_i)$를 식 (2)와 같이 정의한다.

$$\begin{aligned} \boldsymbol{SA}_t(B_i) &= (\boldsymbol{p}_t^{B_i}(x,y),\ \boldsymbol{s}_t^{B_i}(x,y)) \\ \boldsymbol{p}_t^{B_i}(x,y) &= \boldsymbol{c}_{t-1}^{B_i}(x,y) + MV_{t-1}(B_i) \\ \boldsymbol{s}_t^{B_i}(x,y) &= \boldsymbol{s}_{\min}(x,y) + (1 - CF_{t-1}(B_i)) \\ &\quad \cdot (\boldsymbol{s}_{\max}(x,y) - \boldsymbol{s}_{\min}(x,y)) \end{aligned} \tag{2}$$

식 (2)에서 탐색영역의 중심 위치 $\boldsymbol{p}_t^{B_i}(x,y)$는 블록정합 초기에 분할된 블록 B_i의 중심 위치 $\boldsymbol{c}_{t-1}^{B_i}(x,y)$와 $t-1$ 시점에서 추출한 블록 B_i의 동작벡터 $MV_{t-1}(B_i)$의 합으로 정의된다. 영상의 움직임은 급격히 변화하기 보다는 완만하게 변화한다는 연속성 제약 사항(motion continuity constraint) [9]에 근거하여 $t-1$ 시점에서 추출한 동작벡터를 이용해서 t 시점에서의 동작벡터의 위치를 예측할 수 있다.

식 (2)에서 탐색영역의 크기 $\boldsymbol{s}_t^{B_i}(x,y)$는 본 논문에서 사용하는 탐색영역의 최소 크기와 최대 크기를 나타내는 $\boldsymbol{s}_{\min}$과 $\boldsymbol{s}_{\max}$ 및 블록 B_i에서 추출되는 동작벡터의 신뢰도 $CF_{t-1}(B_i)$를 이용하여 정의된다. 식 (2)에서와 같이 탐색영역의 크기는 동작벡터의 신뢰도에 따라 연속적으로 변화하며, 동작벡터의 신뢰도는 탐색영역의 크기를 확대(amplify) 또는 축소(deamplify)시키는 가중치의 역할을 하도록 정의된다. 즉, $t-1$ 시점에서 추출된 동작벡터가 높은 신뢰도를 가질 경우에는 블록정합이 정확하고 안정적으로 수행되었다는 것을 나타내므로 t 시점에서 탐색영역의 크기를 축소하여 계산의 복잡도를 최소화하고, $t-1$ 시점에서 추출된 동작벡터가 낮은 신뢰도를 가질 경우에는 블록정합이 불안정하게 수행되었다는 것을 나타내므로 t 시점에서 탐색영역의 크기를 확대하여 블록정합의 정확도를 최대화할 수 있도록 한다. 본 논문에서는 이러한 개념을 기반으로 동작벡터의 신뢰도를 식 (3)과 같이 정의한다. 식 (3)에서와 같이 $t-1$ 시점의 동작벡터의 신뢰도는 블록 B_i에서 구한 동작벡터와 블록 B_i의 인접 블록들에서 구한 동작벡터들을 비교하여 구한다. 일반적으로, 인접한 블록들은 유사한 동작벡터를 가질 확률이 높으므로 블록 B_i에서 구한 동작벡터와 인접 블록들에서 구한 동작벡터들의 차이가 클수록 블록은 큰 움직임을 가질 확률이 높고, 동작벡터들의 차이가 작을수록 블록은 작은 움직임을 가질 확률이 높다.

$$CF_{t-1}(B_i) = \frac{K_1}{1 + K_2 \cdot MD_{t-1}(B_i)} \tag{3}$$

$$MD_{t-1}(B_i) = \frac{\| r_{t-1} - \mu_{t-1} \|^2}{\sigma_{t-1}^2}$$

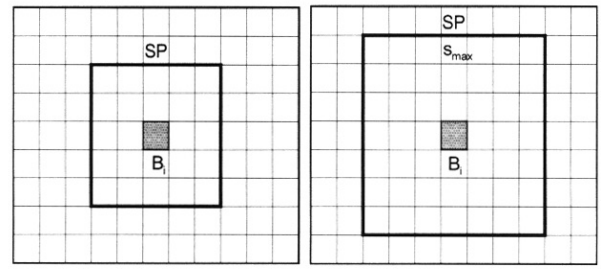
$$r_{t-1} = MV_{t-1}(B_i)$$

$$\mu_{t-1} = \frac{\sum_{j=i-4}^{i+4} MV_{t-1}(B_j)}{9}$$

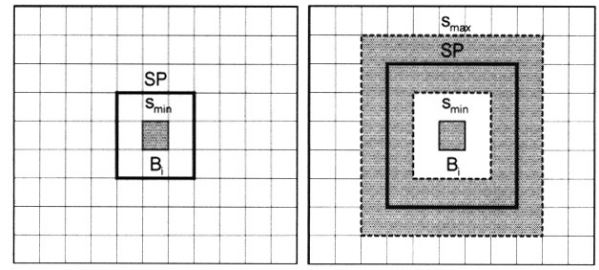
$$\sigma_{t-1}^2 = \frac{\sum_{j=i-4}^{i+4} (MV_{t-1}(B_j) - \mu_{t-1})^2}{9}$$

식 (3)의 $MD_{t-1}(B_i)$는 $t-1$ 시점의 i번째 블록 B_i의 동작벡터인 r_{t-1}의 마할라노비스 거리(Mahalanobis Distance)를 나타낸다. 즉, 인접 블록에서의 동작벡터들과 비교하여 블록 B_i의 동작벡터의 차이가 얼마나 상이한 값을 갖는가를 평가한다. 그리고 식 (3)에서 K_1과 K_2는 상수로서 K_1은 신뢰도가 취할 수 있는 최대값을 결정하고, K_2는 신뢰도의 범위를 결정하는 역할을 한다. 본 논문에서는 실험적으로 이들을 각각 0.1과 1로 설정하였다.

(그림 2)는 탐색영역의 크기를 고정적으로 설정하는 기존의 방법과 적응적으로 설정하는 제안된 방법을 보여주고 있다. (그림 2)(a)는 탐색영역의 크기가 고정적으로 설정되는 기존의 방법이고 (그림 2)(b), (그림 2)(c), (그림 2)(d)는 본 논문에서 제안한 방법으로 동작벡터의 신뢰도를 나타내는 신뢰도 CF 값이 0, 1, 그리고 0과 1 사이의 값을 가질 때의 탐색영역의 크기 변화를 보여준다.



(a) 고정적인 크기 설정 (b) 동적인 크기 설정(CF = 0)



(c) 동적인 크기 설정 (CF = 1) (d) 동적인 크기 설정 (0 < CF < 1)

(그림 2) 탐색영역의 크기 설정

3. 가변적 크기의 블록정합

본 장에서는 블록 사이의 정합 척도로 사용되는 정합 유사 함수를 정의하는 방법, 그리고 블록의 크기를 가변시키는 방법에 대해 설명한다. 블록 크기의 가변적인 설정은 본 논문에서 정의한 평가 함수를 이용한다. 즉, 평가 함수의 조건이 만족될 때까지 블록의 크기를 증가시킨다.

n×n 크기의 블록이 주어질 경우, 블록 단위의 동작 측정은 탐색영역 안에서 정합 유사도가 가장 높은 블록을 찾는다. 블록 사이의 정합 척도로는 여러 가지 기법을 고려할 수 있다[10]. 본 논문에서는 명암값을 이용하여 정합 유사 함수를 식 (4)와 같이 정의한다.

$$DBS(i,j;u,v;n)=\left(1-\frac{1}{n^2}\sum_{y=0}^{n-1}\sum_{x=0}^{n-1}\left|\frac{I_t(i+x,j+y)-I_{t-1}(i+u+x,j+v+y)}{I_{\max}}\right|\right)\times 100 \quad (4)$$

식 (4)에서 n은 블록의 크기를 의미하고, (i, j)는 영상의 (i, j)에 위치하는 블록의 시작 좌표를 의미하며, (u, v)는 기준 블록과 정합되는 블록 사이의 변위를 의미한다. 그리고 I_n, I_{n-1}은 현재 시점과 이전 시점 영상의 명암값을 의미하고, $I_{\max}$는 명암값의 최대값 255를 의미한다. 그리고 DBS는 0에서 100사이의 범위를 갖도록 정규화된다.

본 논문에서 제안하는 가변적 크기의 블록정합 방법은 구분력 있는 최대 정합 유사도를 추출할 때까지 블록의 크기를 확대하면서 블록 사이의 정합을 반복적으로 수행한다. 이를 위해서 본 논문에서는 다음과 같은 두 가지 평가 기준을 설정하여 사용한다. 첫째, 블록 크기의 확대 여부에 대한 기준을 설정한다. 즉, 정해진 탐색영역에서 추출한 정합 유사도의 분포 중 최대 정합 유사도가 구분력이 있는가를 평가한다. 이는 최대 정합 유사도 $DBS_{\max}$를 가지는 위치를 중심으로 인접 블록과의 DBS의 차이를 비교하여 평가한다. 즉, 본 논문에서는 $DBS_{\max}$를 가지는 블록과 인접 블록에서의 DBS의 차이 중 최소값을 경사도(peakness)로 정의하고, 경사도가 임계치 TH 이하일 경우에만 블록을 확대한다. 둘째, 블록의 크기를 확대하면서 정합 유사도를 반복적으로 계산할 경우 블록 크기의 확대 지속 여부에 대한 기준을 설정한다. 본 논문에서는 블록의 크기를 확대함에 따라 경사도의 1차 미분값이 +에서 −로 변하는 변곡점을 발견할 때까지 반복한다.

위에서 언급한 두 가지 평가 기준을 바탕으로 본 논문에서 사용하는 평가 함수를 수학적으로 정형화하면 식 (5)와 같다.

$$\begin{aligned}\Phi(x(n),y(n)) &= \max\begin{bmatrix} e_1+TH(x(n),y(n)) \\ e_2+GD(x(n),y(n)) \end{bmatrix} \\ &\quad\times\frac{e_1+TH(x(n),y(n))}{e_2+GD(x(n),y(n))} \qquad (5)\\ TH(x(n),y(n)) &= T_{PK}-PK(x(n),y(n)) \\ GD(x(n),y(n)) &= PK(x(n),y(n)) \\ &\quad -PK(x(n-1),y(n-1)) \\ PK(x(n),y(n)) &= \min_{-1\le i^*,j^*\le 1}[DBS(x(n),y(n)) \\ &\quad -DBS(x(n)+i^*,y(n)+j^*)]\end{aligned}$$

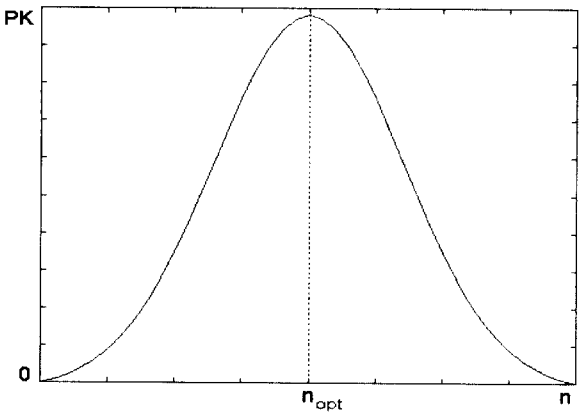
$$where\ \begin{cases} e_1 & : \text{infinitesimal positive number} \\ e_2 & : \text{infinitesimal negative number} \\ T_{PK} & : \text{threshold value of peakness} \end{cases}$$

식 (5)에서 $(x(n), y(n))$은 블록의 크기가 n×n일 때의 최대 정합 유사도 $DBS_{\max}$를 가지는 블록의 중심 위치를 의미하고, i^*와 j^*는 위치가 $(x(n), y(n))$인 블록의 인접 블록의 중심 위치를 의미한다. 그리고 PK는 본 논문에서 정의한 정합 유사도 사이의 경사도를 의미하고, GD는 경사도의 1차 미분값을 의미한다. GD와 TH의 값의 범위에 따른 평가 함수 Φ 값의 범위는 <표 1>과 같다.

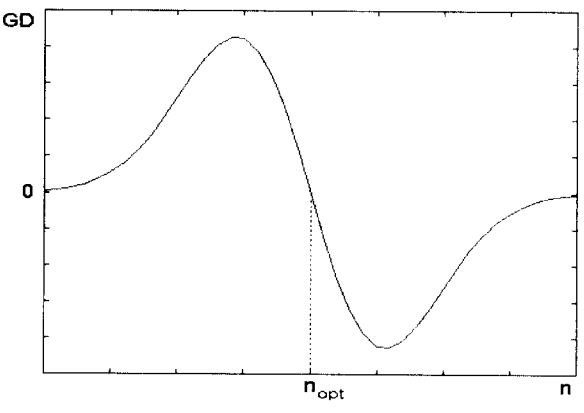
〈표 1〉 평가 함수의 값의 범위

$\Phi(x(n), y(n))$		$TH(x(n), y(n))$		
		$TH>0$	$TH=0$	$TH<0$
$GD(x(n),y(n))$	$GD>0$	+	+	-
	$GD=0$	-	-	-
	$GD<0$	-	-	-

(그림 3)은 PK의 값에 따른 최적의 블록의 크기 n_{opt}를 보여주며, (그림 4)는 GD의 값에 따른 n_{opt}를 보여준다. (그림 3)과 (그림 4)에서 확인할 수 있듯이 PK의 일차미분 값인 GD가 0에 도달할 때에 최적의 블록의 크기를 구할 수 있다. 본 논문에서는 정합 유사도에 적용하는 임계치와 경사도를 결합하여 평가 함수를 정의함으로써 간편하게 블록 크기의 확대 여부를 결정하면서 블록정합을 반복할 수 있다.



(그림 3) PK와 n_{opt}와의 관계



(그림 4) GD와 n_{opt}와의 관계

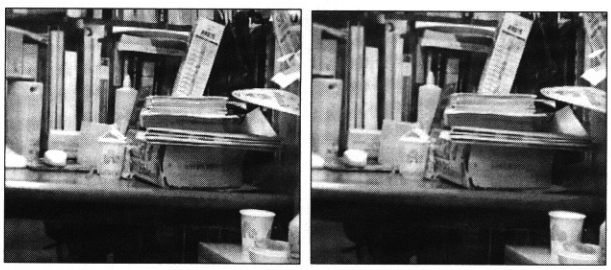
위에서 기술한 가변적 크기의 블록정합 방법을 간략한 수도 코드(pseudo code) 형태로 정리하면 <표 2>와 같다. <표 2>에서는 제안한 블록정합의 계산의 복잡도를 개선하기 위해서 탐색 공간 내에서 구한 정합 유사도의 분포를 모두 사용하지 않고 가능성이 높은 정합 유사도만을 선택하여 사용한다. 이를 위해 전역적 탐색 공간 SA_G와 선택적 탐색 공간 SA_L을 정의하여 사용한다.

〈표 2〉 가변적 크기의 블록정합의 수도 코드

A Size-Variable Block Matching Algorithm
step 1 : 영상을 n×n 크기의 블록으로 분할한다 step 2 : 탐색 공간 SA를 동적으로 설정한다 step 3 : n_{min}, n_{max}를 설정한다. step 4 : n ← n_{min}으로 설정한다. step 5 : 탐색 공간을 탐색하면서 정합 유사도 DBS를 계산한다. step 5.1 : n == n_{min}이면 전역적 탐색 공간 SA_G을 탐색하면서 정합 유사도 DBS를 계산한다. step 5.2 : n ≠ n_{min}이면 선택적 탐색 공간 SA_L을 탐색하면서 정합 유사도 DBS를 계산한다. step 6 : DBS를 오름차순으로 정렬(DBS_1, DBS_2, …, DBS_{max})하고 DBS_k에서 DBS_{max}를 가지는 위치를 선택적 탐색 공간 SP_L로 설정한다(k = max×4/5). step 7 : 블록의 크기가 n일 때의 DBS_{max}를 가지는 위치 (x(n), y(n))를 찾는다. step 8 : 평가 함수(Evaluation Function) $\Phi(x(n), y(n))$를 계산한다. step 9 : 블록의 크기의 확대 여부를 결정한다. step 9.1 : $\Phi < 0$ 또는 n == n_{max}이면 이 위치의 블록을 대응 블록으로 선언한다. step 9.2 : $\Phi > 0$ 이면 n ← n + 1으로 설정하고 step 5.2로 간다. step 10 : 대응 블록을 대표하는 좌표점의 차이로 동작벡터를 추출한다.

4. 실험 결과

(a) 입력 영상 t　　(b) 줌인된 영상

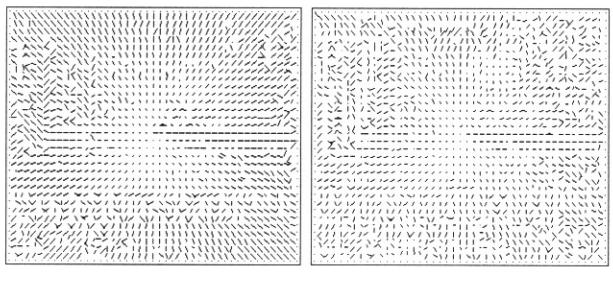


(c) 회전된 영상　　(d) 여러 가지 동작이 포함된 영상

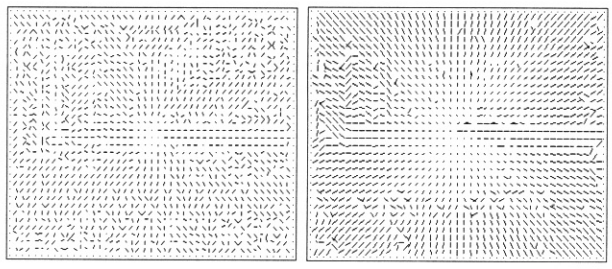
(그림 5) 입력 영상

본 장에서는 제안한 가변적 크기의 블록정합 알고리즘에 대한 성능을 평가한다. 본 논문에서는 기존의 방법들과 제안한 방법과의 성능을 평가하기 위해 동작벡터의 정확성을 중심으로 결과를 비교 분석한다. 본 논문에서 사용한 입력 영상의 크기는 320×240이다. (그림 5)는 실험에서 사용한 4개의 입력 영상을 보여준다. (그림 5)(a)는 실내 환경에서 t 시점에서 촬영된 영상이며, 동작벡터 추출을 위한 기준 영상으로 사용된다. (그림 5)(b)와 (그림 5)(c)는 각각 1.05배율/frame의 속도로 줌인되는 입력 영상과 2도/frame의 속도로 시계 방향으로 회전되는 입력 영상을 보여준다. 그리고 (그림 5)(d)는 네 가지 종류의 카메라의 동작이 포함된 매우 복잡한 환경에서 촬영된 영상으로서 2 pixels/frame 속도의 수평 이동, -2 pixels/frame 속도의 수직 이동, 1.05배율/frame 속도의 줌인, 그리고 2도/frame 속도의 시계 방향으로의 회전 동작이 혼합된 영상이다.

본 논문에서 제안된 가변적인 크기의 블록정합 방법과 기존의 블록정합 방법들을 비교하기 위해서 본 논문에서는 전역 탐색에 의한 블록정합 방법, 3단계 탐색에 의한 블록정합 방법, 그리고 4단계 탐색에 의한 블록정합 방법을 구현하였다. (그림 6)은 (그림 5)(a)와 (그림 5)(b) 입력 영상을 이용하여 추출한 동작벡터를 보여준다. 이상적으로 추출된 동작벡터는 영상의 중심으로부터 방사선의 모양으로 밖으로 분출하는 형태가 되어야 한다. (그림 6)에서 확인할 수 있듯이 제안한 블록정합 방법이 보다 정확하게 동작벡터를 추출함을 시각적으로 확인할 수 있다.



(a) 전역 탐색에 의한 방법　　(b) 3단계 탐색에 의한 방법

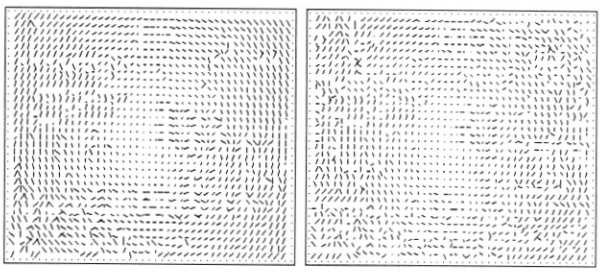


(c) 4단계 탐색에 의한 방법　　(b) 제안한 방법

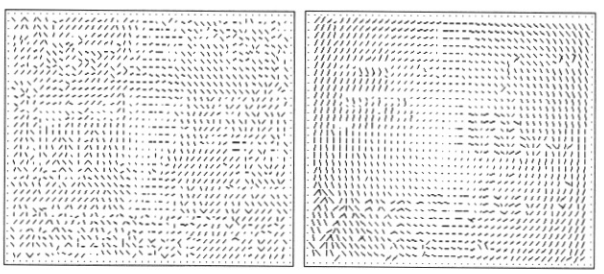
(그림 6) 동작벡터 추출 결과(줌인)

(그림 5)(c)와 (그림 5)(d)에 대한 실험 결과는 (그림 7)과

(그림 8)에 나와 있다. 이상적으로, (그림 7)의 동작벡터는 입력 영상의 중심을 기준으로 원의 형태로 시계 방향으로 회전해야 하며, (그림 8)의 동작벡터는 영상의 중심으로부터 남동쪽으로 2pixels 떨어진 위치를 중심으로 나선형 모양을 형성해야 한다.

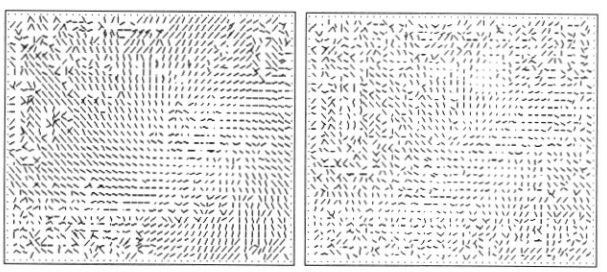


(a) 전역 탐색에 의한 방법 (b) 3단계 탐색에 의한 방법

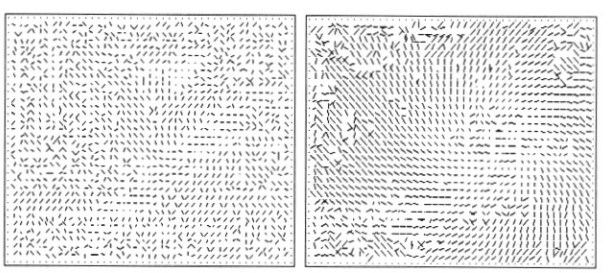


(c) 4단계 탐색에 의한 방법 (b) 제안한 방법

(그림 7) 동작벡터 추출 결과(회전)



(a) 전역 탐색에 의한 방법 (b) 3단계 탐색에 의한 방법



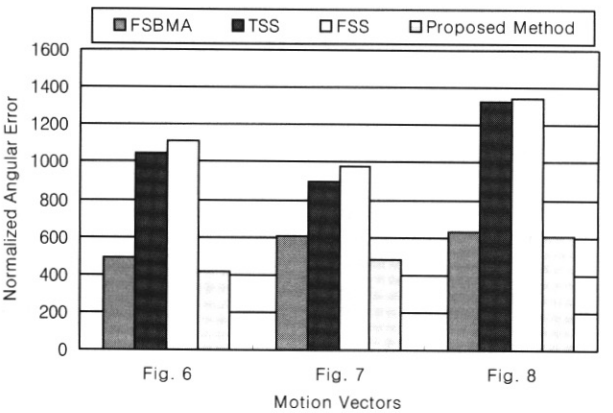
(c) 4단계 탐색에 의한 방법 (b) 제안한 방법

(그림 8) 동작벡터 추출 결과(혼합)

동작벡터의 추출 결과에서 확인할 수 있듯이 기존의 블록정합 방법들 모두 명암값이 분명한 영역에서는 동작벡터를 어느 정도 정확히 추출하였지만, 명암값이 균일하게 분포하는 영역이나 에지나 라인 등이 존재하는 부분에서는 부정확한 블록정합으로 동작벡터를 잘못 추출하였다. 본 논문에서 제안된 방법은 이와 같은 영역에서 블록의 크기가 확대되므로 기존의 방법보다 동작벡터를 정확하게 추출할 수 있다. 특히, 입력 영상이 보다 복잡한 카메라의 동작을 포함할수록 본 논문에서 제안된 방법이 기존의 방법에 비해 보다 좋은 결과를 산출한다.

본 논문에서는 동작벡터의 정확성을 수량적으로 확인하기 위해 식 (6)과 같은 정규화된 방향 에러(normalized angular error)를 사용한다[11, 12]. 식 (6)에서 $\mathbf{V}_c^i$는 이미 알고 있는 정확한 동작벡터를 나타내고, $\mathbf{V}_e^i$는 측정된 동작벡터를 나타낸다.

$$E_\theta = \sum_{i=1}^{N} \cos^{-1}\left(\frac{(\mathbf{V}_c^i, 1) \cdot (\mathbf{V}_e^i, 1)}{\sqrt{1+\| \mathbf{V}_c^i \|^2}\sqrt{1+\| \mathbf{V}_e^i \|^2}} \right) \quad (6)$$



(그림 9) 정규화된 방향 에러

(그림 9)는 (그림 5)의 입력 영상으로부터 기존의 블록정합 방법과 제안한 블록정합 방법에서 추출한 동작벡터의 정규화된 방향 에러를 그래프로 보여준다. 그림에서 확인할 수 있듯이 제안한 방법의 방향 에러가 다른 방법의 방향 에러보다 작음을 알 수 있다. 기존의 방법 중 3단계 탐색에 의한 방법이 가장 나쁜 결과를 나타내고, 4단계 탐색에 의한 방법도 3단계 탐색에 의한 방법과 유사한 결과를 나타낸다. 전역 탐색에 의한 방법은 기존의 다른 두 가지 방법에 비해 좋은 결과를 나타내지만 제안된 방법보다는 나쁜 결과를 나타낸다.

<표 3>에서는 여러 가지 카메라의 동작이 포함된 입력 영상에 대해 탐색 영역 및 블록의 크기를 고정적으로 설정하는 기존의 전역 탐색 방법과 적응적으로 설정하는 제안된 방법과의 정량적인 성능 비교를 보여준다. 본 논문에서 사용한 탐색 영역의 크기 범위는 5×5에서 15×15까지이며, 블록 크기의 범위는 7×7에서 19×19까지이다. 기존의 블록정합 방법에서는 탐색 영역 및 블록의 크기를 고정시키는 반면 제안된 방법에서는 정합 정확도가 좋지 않은 영역에

서 탐색 영역 및 블록의 크기를 동적으로 변화시킨 후 재정합함으로써 블록 정합의 정확도를 개선할 수 있다.

〈표 2〉 탐색영역 및 블록의 크기 변화에 따른 성능 비교

방법	탐색 영역	블록	사용된 블록의 개수				에러
			7×7	11×11	15×15	19×19	
전역 탐색	7×7	7×7	300	0	0	0	134.709611
		11×11	0	300	0	0	107.001824
		15×15	0	0	300	0	94.068722
		19×19	0	0	0	300	97.334795
제안 방법	가변적	가변적	229	15	69	40	98.038978

5. 결 론

본 논문에서는 블록 단위의 동작벡터를 효과적으로 추출하는 가변적인 크기의 블록정합 알고리즘을 제안하였다. 제안된 블록정합 알고리즘은 탐색영역의 크기와 블록의 크기를 동적으로 설정한다. 본 논문에서는 탐색영역의 크기를 결정하기 위해 일반적으로 수용되는 동작 제약 사항인 "동작은 완만하고 천천히 변화한다"를 사용하였다. 그리고 구분력 있는 최대 정합 유사도를 추출할 때까지 블록의 크기를 확대하면서 블록 사이의 정합을 반복적으로 수행하였다.

탐색영역의 설정은 중심 위치 설정 및 크기 설정의 두 가지 단계로 구성된다. 탐색영역의 중심 위치는 영상의 움직임은 급격히 변화하기 보다는 완만하게 변화한다는 연속성 제약 사항에 근거하여 $t-1$ 시점에서 추출한 동작벡터를 이용하여 t 시점에서의 동작벡터의 위치를 예측하였다. 그리고 탐색영역의 크기 설정은 이전 시점에서 구한 동작벡터들의 신뢰도를 이용하였다. 즉, 동작벡터의 신뢰도가 높은 값을 가질 경우에는 블록정합이 정확하게 수행되어 오고 있음을 나타내므로 탐색영역의 크기를 축소시키고, 그렇지 않을 경우에는 탐색영역의 크기를 증가시킨다.

블록의 크기를 가변적으로 결정하는 과정에서는 블록의 크기의 적합성을 판단하는 평가 함수를 이용하였다. 본 논문에서 정의된 평가 함수는 블록 크기의 확대 여부에 대한 기준 및 블록 크기의 확대 지속 여부에 대한 기준을 설정하고 이를 수학적으로 정의하였다. 다시 말해, 평가 함수가 양의 값을 가질 경우 블록의 크기를 확대하도록 설정하였으며, 그렇지 않을 경우 현재의 블록의 크기가 적당하다고 판단하고 대응하는 블록으로부터 동작벡터를 계산하도록 설정하였다.

실험에서는 제안된 방법과 기존의 방법들을 정확성을 중심으로 비교 분석하였다. 제안한 방법은 명암값이 균일하게 분포한 영역, 또는 에지와 라인 등이 존재하여 블록정합이 저하될 수 있는 영역에서 블록의 크기를 적응적으로 확대하므로 블록정합의 정확도를 높일 수 있다. 특히, 영상이 복잡한 카메라의 동작을 포함할수록 기존의 방법에 비해 좋은 결과를 보였다. 향후 연구 계획으로는 보다 다양한 특징을 이용할 예정이며, 평가 함수의 수학적인 정형화를 좀 더 개선할 예정이다.

참 고 문 헌

[1] ISO/IEC 11172-2 (MPEG-1 Video), Information Technology-Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/s : Video, 1993.

[2] R. Srinivasan and K. R. Rao, "Predictive Coding Based on Efficient Motion Estimation," IEEE Transactions on Communications, COM-33, pp.1011-1015, 1985.

[3] Kyoung Won Lim, Byung Cheol Song and Jong Beom Ra, "Fast Hierarchical Block Matching algorithm Utilizing Spatial Motion Vector Correlation," Proceedings of Visual Communications and Image processing, Vol.3024, pp.284-291, 1997.

[4] B. Liu and A. Zaccarin, "New Fast Algorithms for the Estimation of Block Motion Vectors," IEEE Transactions on Circuits and Systems for Video Technology, Vol.3, No.2, pp.438-441, 1994.

[5] T. Koga, K. Iinuma, A. Hirano, Y. Iijima and T. Ishiguro, "Motion Compensated Interframe Coding for Video Conferencing," in Proceedings of NTC81, pp.C9.6.1-9.6.5, New Orleans, LA, 1981.

[6] R. Li, B. Zeng and M. L. Liou, "A New Three-Step Search Algorithm for Block Motion Estimation," IEEE Transactions on Circuits and Systems for Video Technology, Vol.4, No.4, pp.438-441, 1994.

[7] L. G. Chen, E. T. Chen, Y. S. Jehng and T. D. Chiueh, "An Efficient Parallel Motion Estimation Algorithm for Digital Image Processing," IEEE Transactions On Circuits and Systems for Video Technology, Vol.1, No.4, pp.378-385, 1991.

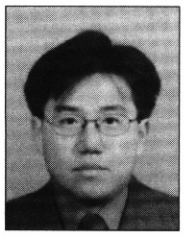
[8] Lai-Man Po and Wing-Chung Ma, "A Novel Four-Step Algorithm for Fast Block Motion Estimation," IEEE Transactions on Circuits and Systems for Video Technology, Vol.6, No.3, pp.313-317, 1996.

[9] Vincent S. and S. Hwang, "Tracking Feature Points in Time-Varying Images Using an Opportunistic Selection Approach," Pattern Recognition, Vol.22, pp.247-256, 1989.

[10] Ramesh Jain, Rangachar Kasturi, and Brian G. Schunck, *Machine Vision*, McGraw-Hill, 1995.

[11] Yu-Te Wu, Jeffrey Cohn and Ching-Chung Li, "Optical Flow Estimation using Wavelet Motion Model," International Conference on Computer Vision, pp.992-998, 1998.

[12] J. L. Barron, D. J. Fleet and S. S. Beauchemin, "Performance of Optical Flow Techniques," International Journal of Computer Vision, Vol.12, pp.43-77, 1994.



장 석 우

email : swjang@vision.ssu.ac.kr

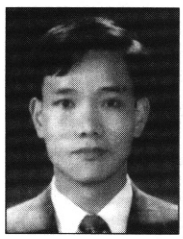
1995년 숭실대학교 전자계산학과(공학사)

1997년 숭실대학교 대학원 전자계산학과 (공학석사)

2000년 숭실대학교 대학원 컴퓨터학과(공학박사)

1998년~현재 숭실대학교 생산기술연구소 연구원

관심분야 : 컴퓨터 비전, 영상 처리, 동작 이해, 비디오 검색 등



김 봉 근

email : bkkim@cjnc.ac.kr

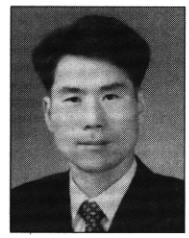
1987년 수원대학교 전자계산학과(이학사)

1991년 숭실대학교 대학원 전자계산학과 (공학석사)

1997년 숭실대학교 대학원 전자계산학과 (공학박사)

1993년~현재 청주과학대학 컴퓨터학과 부교수

관심분야 : 패턴 인식, 컴퓨터 비전, 영상처리 등



김 계 영

email : gykim@computing.ssu.ac.kr

1990년 숭실대학교 전자계산학과(공학사)

1992년 숭실대학교 대학원 컴퓨터학과 (공학석사)

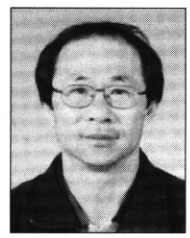
1996년 숭실대학교 대학원 컴퓨터학과 (공학박사)

1996년~1997년 한국전자통신연구원 포스닥 연구원

1997년~2001년 한국전력공사 전력연구원 선임연구원

2001년~현재 숭실대학교 컴퓨터학부 조교수

관심분야 : 컴퓨터 비전, 형태 인식, 생체 인식, 증강 현실, 영상 및 신호처리 등



최 형 일

email : hic@computing.ssu.ac.kr

1979년 연세대학교 전자공학과(공학사)

1982년 미시간대학교 전산공학과(공학석사)

1987년 미시간대학교 전산공학과(공학박사)

1987년~현재 숭실대학교 미디어학부 교수

관심분야 : 컴퓨터 비전, 패턴 인식, 퍼지 이론, 비디오 검색, 인터페이스 에이전트 등