움직임 벡터의 시간적 연속성을 이용한 고속 움직임 추정 알고리즘

이경환[†] · 류권열[†] · 최정현[††]

요 약

본 논문에서는 움직임의 시간적 연속성을 이용한 새로운 움직임 추정 알고리듬을 제안한다. 기존의 전역 탐색 방법 (FSA)에서와 같이 전역 탐색 영역 (GSR)을 정하고, GSR 안에 시간적으로 인접한 블록들의 움직임 벡터들로 예측한 국부 탐색 영역 (LSR)들을 정한다. 여기서 시간적으로 인접한 블록은 이전 프레임의 블록 중 움직임 벡터로 볼 때 현재 프레임에 영향을 끼칠 가능성이 있는 블록이다. 그리고 이 LSR들로 만든 영역에 대해서만 움직임 탐색을 행한다. 모의 실험 결과, 제안한 방법은 기존의 FSA 방법에 비해 탐색 영역을 줄여 계산량을 현저히 낮추면서도 우수한 화질을 유지하는 것을 볼 수 있었다.

Fast Motion Estimation Algorithm using Temporal Continuity of Motion Vector

Kyeong Hwan Lee[†], Kwon Yeol Ryu[†] and Jung Hyun Choi[††]

ABSTRACT

This paper proposes a new motion estimation algorithm using the temporal continuity of motion. We set up a wide global search region (GSR), which basically corresponds to the search region of FSA, and local search regions (LSRs), the positions for which are predicted by the motion vectors of the temporal neighbor blocks, are constructed in the GSR. The previous frame blocks that possibly have effects on the current block are to be the temporal neighbor blocks. Then Motion estimation is only performed in the areas made by LSRs. Experimental results show that the proposed method can maintain visual qualities with significant reductions of complexity by reducing search regions, when compared to the conventional FSA.

Key words: motion estimation, BMA, FSA, MAD, motion vector

1. 서 론

동영상 신호에는 높은 시간적 및 공간적 중복성이 존재하므로 이를 제거함으로써 높은 부호화 효율을 가질 수 있다. 시간적인 중복성을 제거하는 움직임 추정방법으로는 화소 단위의 추정 방법인 PRA (pel recursive algorithm)와 블록 정합 방법인 BMA (block matching algorithm)가 있는데, 이중 PRA는 복잡하고 계산량이 많아서 잘 사용되지 않고, 대부분의 경우 하드웨어 실현이 간단하고 계산량이 적은 BMA를 사용하여 움직임을 추정한다[1,2]. BMA 방법은 이전 프레임의 현재 블록을 중심으로 일정한 탐색 영역을 정하여 왜곡을 계산하여 가장 왜곡이 작은 블록과 현재 블록의 차이를 움직임 벡터로 정하여 움직임 보상을 하는 방법으로, 일반적으로 탐색 영역의 전체를 화소단위로 천이시켜 탐색하는 전역 탐색 방법, 즉 FSA (full search algorithm)을 사용한다. 그러나 FSA를 이용하여 정확한 움직임 탐색을 하려면 탐색 영역이 넓어야 하므로 계산량이 증가하

접수일 : 2002년 9월 3일, 완료일 : 2003년 4월 16일
[†] 정회원, 위덕대학교 멀티미디어공학과 조교수
[††] 기술신용보증기금 대구기술평가센터 차장

는 단점이 있다[3].

동영상 신호에는 높은 시간적 및 공간적 상관성이 존재하며, BMA로 추정한 움직임 벡터 또한 시간적 및 공간적 인접 블록들 간에 비슷하게 나타난다. 이 특성을 이용하여, 인접 블록들의 움직임 벡터들로써 현재 블록의 움직임을 예측하고, 이를 기준으로 움직임을 추정하여 성능을 향상시킨 방법들이 제안되었다[4,5]. 이들 예측 움직임 추정 방법들에서는, 시간적 및 공간적 인접 블록들의 움직임 벡터들에 가중치를 곱한 합(weighted sum)으로써 현재 블록의 움직임을 예측한다. 또한 현재 프레임의 먼저 움직임 탐색한 공간적 인접 블록들과 이전 프레임의 현재 블록, 즉 시간적 인접 블록의 움직임 벡터를 모두 이용하고 가중치를 부여하여 문턱값(threshold)에 의해 움직임을 예측하는 방법이 제안되어 더욱 정확한 움직임을 추정하였다[6]. 그러나 이들 방법은 결국 하나의 움직임만을 예측하여 이를 중심으로 움직임 탐색을 행하므로, 움직임이 복잡하거나 빠른 부분의 경우 인접 블록들의 움직임 벡터들 간에 상관성이 떨어지는 단점이 있다.

한편 화소 밝기의 시공간적인 경사의 관계로부터 화소 단위의 움직임을 추정하는 방법인 광류 추정 방법(optical flow algorithm)을 이용하여 작은 삼각형 부영역에 선형함수를 적용하여 움직임장(motion field)을 추정하는 다중해상도 (multi-resolution) 광류 추정 방법이 제안되어 FSA에 비해 좋은 성능을 나타내었는데[7], 움직임이 크고 복잡한 영역에서는 추정 오차가 증가하는 단점이 있다.

BMA 방법으로 구한 블록의 움직임은 프레임의 진행에 따라 시간적으로 계속되는 특성이 있다[8,9]. 이를 이용하면 현재 블록의 움직임을 적응적으로 예측할 수 있는데, 현재 블록에 대한 탐색 영역은 이전 프레임에서 구성되므로 이 탐색 영역에 포함되는 시간적 인접 블록들의 움직임 벡터는 현재 블록의 움직임을 구하기 위한 좋은 근거가 된다. 그러므로 이전 프레임의 시간적 인접 블록들의 움직임을 이용하여 현재 블록의 움직임 특성을 예측할 수 있으며, 이에 따라 탐색 영역을 효율적으로 정할 수 있다. 이들 시간적 인접 블록들의 움직임 벡터는 이미 이전 프레임의 부호화 과정에서 알고 있는 정보이므로, 이를 구하기 위한 별도의 계산은 필요 없다.

본 논문에서는 움직임의 시간적 연속성을 이용한 움직임 추정(temporal predictive motion estimation, TPME) 방법을 제안하였다. 이 방법에서는, 기본적으로 넓은 전역 탐색 영역(global search region, GSR)을 설정하고, 이 중 시간적 인접 블록들에 의해서 예측된 탐색 영역에 대해서만 탐색을 행한다. 시간적 인접 블록들의 움직임을 이용하여 국부 탐색 영역(local search region, LSR)의 위치를 GSR 범위 안에서 예측한다. 각각의 인접 블록들의 움직임 벡터로써 LSR의 위치를 예측할 수 있으며, 인접 블록들 수만큼의 LSR이 구해진다. 배경 부분과 같이 인접 블록들의 움직임이 비슷한 영역에서는, LSR이 GSR 내의 유사한 위치에서 많이 겹치게 되고, 이는 탐색할 영역이 예측에 의해서 줄어들므로 계산량이 감소하고, 크고 불규칙한 움직임이 있는 영역과 같이 인접 블록들의 움직임의 상관성이 떨어질 경우에는 LSR이 넓게 분산되므로 탐색할 영역이 증가하게 되어 정확한 움직임 추정을 하게되어, 제안한 방법을 이용하면 고속 적응적 움직임 추정이 가능하다.

다양한 움직임 특성을 가지는 실험 영상들에 대한 모의 실험 결과, 제안한 TPME 방법은 움직임이 균일한 영상에 대해서는 기존의 움직임 예측 방법들과 같이 전역 탐색 FSA 방법에 비해 탐색영역을 줄일 수 있어 고속 움직임 추정이 가능하였으며, 움직임이 큰 영상에 대해서 적응적인 움직임 추정을 행하여 기존 방법들에 비해 우수한 PSNR을 나타내었다.

2. 기존의 움직임 탐색 방법

2.1 FSA 움직임 탐색방법

블록 단위의 움직임 추정 방법인 블록 정합 방법(BMA)은, 블록 기반의 변환 부호화와 호환성이 좋고 알고리듬이 간단하므로, 동영상 부호화에서 널리 사용되고 있다. 또한 한 개의 움직임 벡터가 블록의 움직임을 대표하여 구해지기 때문에, 높은 데이터 압축률이 요구되는 동영상 부호화에 적합하다.

BMA로서 가장 기본적인 움직임 추정 방법인 전역 탐색 방법(FSA)에서는, 이전 프레임에서 현재 블록과 같은 위치를 중심으로 탐색 영역을 정하고, 이 영역 내의 모든 탐색점들에 대한 블록 정합을 행하여, 그 중 최소 오차를 가지는 탐색점의 좌표를 움직임 벡터로 구한다.

BMA에서는, 프레임을 겹치지 않는 블록들로 나

누고 각 블록에 대해서 움직임 벡터를 구한다. 블록의 크기를 $M \times M$, 프레임 내에서 블록의 위치 좌표를 (m, n)이라 하면, $B_k(m, n)$는 k번째 프레임에서의 (m, n) 위치의 블록을 나타낸다. $B_k(m, n)$의 움직임 벡터를 $V_k(m, n)$라 하면, $V_k(m, n)$는,

$$\boldsymbol{V}_k(m, n) = [v^v, v^h]^T \tag{1}$$

와 같이 표현되고, 여기서 v^v, v^h는 각각 $V_k(m, n)$의 수직, 수평 성분을 나타낸다.

BMA에서 블록의 정합 오차를 구하기 위한 왜곡 척도(distortion measure)로는, *MSD* (mean square difference)와 성능이 비슷하면서 이보다 계산량이 훨씬 적은 *MAD* (mean absolute difference)를 널리 사용한다.

전역 탐색 블록 정합 방법에서, $B_k(m, n)$의 움직임 벡터 $V_k(m, n)$를 구하기 위한 블록의 *MAD*는,

$$MAD_{(i,j)}(m, n) = \frac{1}{M^2} \sum_{0 \le x, y < M} | I_k(m \cdot M + x,\ n \cdot M + y) - \widetilde{I_{k-1}}(m \cdot M + x + i,\ n \cdot M + y + j) | \tag{2}$$

와 같이 표현되고, 여기서, (i, j)는 탐색 영역에서의 탐색점 좌표를 나타내고, $I_k(x, y)$는 원 영상 (original image) k번째 프레임에서 (x, y) 위치의 화소 밝기값 (pixel intensity)을 나타내고, $\widetilde{I_k}(x, y)$는 복원 영상 (reconstructed image) k번째 프레임에서 (x, y) 위치의 화소 밝기값을 나타낸다. 본 논문에서는 수식 전개의 일관성을 위해서, 프레임에서 화소, 블록, 및 탐색점의 좌표는 행렬에서 원소(element)를 나타낼 때와 같이 행 (row)과 열 (column)로 표현한다. 즉, 위치 좌표 (x, y)는 x행 y열을 나타낸다. $B_k(m, n)$의 움직임 벡터 $V_k(m, n)$는,

$$\boldsymbol{V}_k(m, n) = \arg \min_{(i,j)} MAD_{(i,j)}(m, n) \tag{3}$$

와 같이 구할 수 있다. 즉, 탐색 영역 내의 탐색점들 중에서, *MAD*가 가장 작은 탐색점 좌표 (i, j)가 바로 움직임 벡터가 된다.

블록 정합 방법에서의 움직임 벡터를 그림 1에 나타내었다. 이전 프레임 전체를 탐색 영역으로 둘 수도 있으나, 일반적으로 그림 1에서와 같이 현재 블록과 동일한 위치를 중심으로 한 제한된 탐색 영역을 이전 프레임에서 구성하며, 움직임 벡터는 탐색 영역

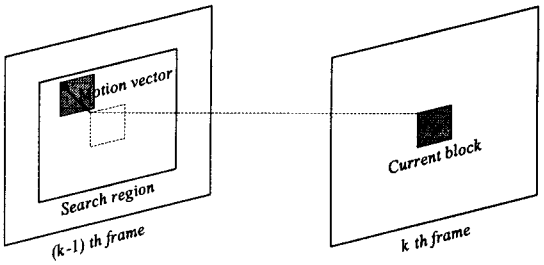


그림 1. 블록 정합 방법에서의 움직임 벡터

내에서 현재 블록과 가장 유사한 영역에 대한 탐색점 좌표가 된다.

2.2 기존의 움직임 예측을 이용한 움직임 탐색방법

블록 정합 방법으로 움직임을 추정하였을 때, 프레임 내에서 공간적으로 인접한 블록들의 움직임 벡터들은 높은 상관성을 가진다. 따라서 현재 프레임에서 인접 블록의 움직임 벡터들을 이용하여 현재 블록의 움직임 벡터를 예측하고, 이를 기준으로 움직임을 추정하는 방법(interblock predictive motion estimation, IBPME)이 제안되었다. 이 방법에서는 그림 2에서와 같이 현재 블록의 움직임을 추정하기 위한 탐색 영역을 공간적 인접 블록의 움직임 벡터들로써 예측한 만큼 이동한다. 움직임을 추정할 때에는, 움직임 예측에 의해 이동된 탐색 영역에 대해서 현재 블록과의 블록 정합을 행한다.

한편 현재 블록의 움직임 벡터를 구할 때, 이전 프레임의 같은 위치를 중심으로 인접한 블록들의 움직임 벡터들을 이용하여, 움직임을 예측하고 추정하는 방법(interframe predictive motion estimation, IFPME)이 제안되었다. 이 방법에서는 그림 3에서와 같이 이

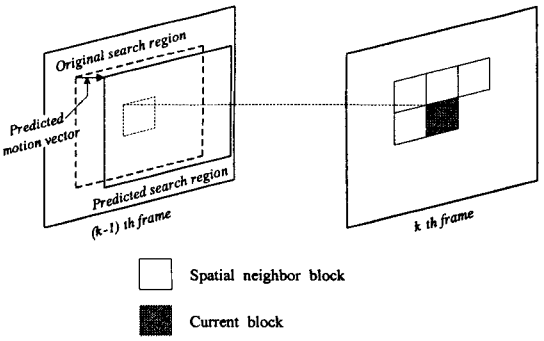


그림 2. 블록 간의 예측을 이용한 움직임 추정 방법

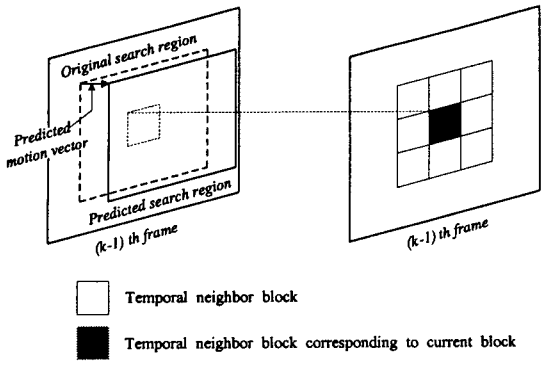


그림 3. 프레임 간의 예측을 이용한 움직임 추정 방법

전 프레임의 시간적 인접 블록들로써 현재 블록의 움직임 벡터를 예측하고, 예측된 움직임 벡터만큼 현재 블록의 탐색 영역을 이동시켜서 현재 블록의 움직임을 추정한다.

이러한 움직임 예측을 이용한 움직임 추정 방법들에서는, 시간적 및 공간적 인접 블록들의 움직임 벡터들에 가중치를 곱한 합으로써 현재 블록에 대한 예측 움직임 벡터를 구하고, 탐색 영역을 예측 벡터만큼 이동하여 움직임을 추정 한다. 그러므로 탐색 영역의 크기를 FSA보다 줄이면서 움직임 추정 성능을 향상시킬 수 있었다. 그러나 이 방법에서도 FSA에서와 마찬가지로 탐색 영역의 크기를 적절히 정하기가 어렵다. 또한 인접 블록의 움직임 벡터들이 비슷한 분포를 가지는 영역에 대해서는 이와 같은 움직임 예측이 정확하지만, 인접 블록의 움직임 벡터들의 상관성이 떨어질 때에는, 한 개의 예측 벡터가 모든 인접 블록들의 움직임을 제대로 표현할 수가 없다.

현재 프레임의 움직임 특성을 판단할 수 있는 가장 좋은 근거는 이전 프레임에서의 움직임이다. 이전 프레임의 움직임 벡터는 이미 부호화 과정에서 알고 있는 것이므로 부가적인 계산량이 필요 없다.

그러므로, 시간적 인접 블록들의 움직임을 이용하면 현재 블록의 움직임을 효율적으로 예측할 수 있으며, 이 때에는 반드시 여러 인접 블록들의 움직임을 고려하여야 한다. 인접 블록들의 움직임이 균일할 경우는 배경 부분 혹은 움직임이 아주 작은 영역이므로, 탐색 영역을 작게 하여도 충분하다. 또한 예측 움직임 벡터들이 크고 불규칙할 경우는 구체적인 물체의 움직임 또는 큰 움직임이 일어나는 영역이므로, 이 영역에 대해서는 탐색 영역을 크게 하여 움직임 추정을 정확히 해주어야 한다.

3. 제안한 TPME 방법

가장 기본적인 블록 정합 방법인 전역 탐색 방법(FSA)에서는, 이전 프레임의 정해진 탐색 영역 내의 모든 탐색점들에 대해서 탐색을 행하여, 그 중 현재 블록과 가장 유사한 영역을 찾는다. 그러므로 탐색 영역을 크게 설정하면, 움직임 보상 영상의 화질이 향상된다. 그러나 이에 따른 급격한 계산량 증가가 큰 문제점이다.

3.1 움직임의 시간적 연속성

동영상 신호에서 프레임 간의 시간 간격은 1/30초 정도로 짧기 때문에, 어떤 프레임에서 물체 또는 영역의 움직임이 나타났다면 이 움직임은 프레임의 진행에 따라 계속되는 성질이 있다. 즉, 이전 프레임의 어떤 영역이 시간적으로 이동하여 현재 프레임의 한 블록으로 나타났다면, 다음 프레임에서 이 블록은 이전 프레임에서 현재 프레임으로 이동한 만큼 옮겨서 나타날 확률이 높다.

그림 4에서는 움직임의 시간적 연속성에 대하여 나타내었다. (k-1)번째 프레임에서 블록의 움직임 벡터는 이 블록이 (k-2)번째 프레임의 어느 위치에서 이동되었는가에 대해서 나타내고 있으며, 이 블록의 실제 움직임(actual motion)은 그림 4에서의 화살표와 같이, 움직임 벡터와 크기는 같고 방향은 반대로 표현된다. 블록의 움직임이 시간적으로 연속적이라고 가정하면, 그림 2에서와 같이 k번째 프레임에서 이 블록의 위치는 (k-2)번째에서 (k-1)번째 프레임으로의 실제 움직임만큼 이동하여 나타나거나 혹은 그 근방에서 나타날 것이다. 그러므로 k번째 프레임에서 각 블록들의 움직임 벡터를 구할 때에는(k-1)번째 프레임의 시간적 인접 블록들의 움직임 벡터들로써 효율적으로 예측할 수 있다.

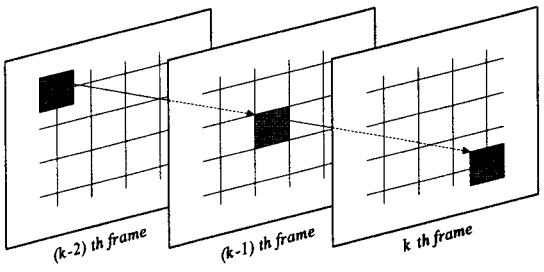


그림 4. 움직임의 시간적 연속성

그러나 블록 단위의 움직임 추정 방법인 BMA의 특성상 프레임의 시간적인 진행에 따라 블록 격자가 바뀌므로 현재 블록의 움직임은 이전 블록들의 부분들이 현재 블록을 구성하는 만큼 움직임 특성이 혼합되어서 표현된다. 따라서 현재 블록의 움직임을 예측할 때에는, 한 개의 예측 움직임 벡터로 표현하는 것보다 탐색 영역 전체에 속하는 모든 인접 블록들의 움직임 벡터를 이용하여 예측하는 것이 타당하다. 블록의 움직임은 시간적 연속성을 가지며, 현재 블록의 탐색 영역에 속하는 시간적 인접 블록들의 움직임 벡터를 이용하면, 현재 블록의 움직임 특성을 적절하게 표현할 수 있다. 시간적 인접 블록의 움직임 벡터는 이미 이전 프레임의 부호화시에 구한 것이기 때문에 부가적인 계산이 필요 없다. 또한, 여러 개의 인접 블록들 각각의 움직임 특성을 모두 고려하여 현재 블록의 움직임을 예측하므로, 한 개의 예측 움직임 벡터로 현재 블록의 움직임을 예측하는 기존의 예측 움직임 추정 방법들에 비해서 견실한 예측 (robust prediction)이 가능하다.

3.2 움직임의 시간적 연속성을 이용한 움직임 추정 방법

블록 정합 움직임 추정 방법에서 블록의 움직임은 시간적인 연속성을 가지며, 이 특성을 이용하여 현재 블록의 움직임 특성을 예측할 수 있고 탐색 영역도 효과적으로 구할 수 있다. 따라서 기본적으로 FSA와 같이 넓은 탐색 영역을 설정해 두고서, 그 영역의 범위 내에서 인접 블록들의 움직임 벡터를 이용하여 탐색 영역을 예측하고 이에 대해서만 탐색을 행함으로써, 계산량을 줄일 수 있다.

본 논문에서는, 움직임의 시간적 연속성을 이용하여 시간적 인접 블록들의 움직임 벡터로써 탐색 영역을 예측하고, 이를 기준으로 움직임을 추정하는 방법(TPME)을 제안한다. 제안한 TPME 방법에서는, FSA에서와 같이 넓은 탐색 영역을 설정해두고, 시간적 인접 블록들의 움직임 벡터를 이용하여 각각에 대해서 예측 탐색 영역을 구한다. 이 때, 예측 탐색 영역은 정해둔 넓은 탐색 영역의 범위 내에서 예측된다.

제안한 TPME 방법을 설명하면 다음과 같다. 먼저 넓은 범위의 전역 탐색 영역(global search region, GSR)을 설정한다. GSR의 탐색 범위를 수직 방향으로 $-H \sim +H$, 수평 방향으로 $-W \sim +W$이라 하고, 현재 블록 $B_k(m, n)$에 대한 GSR을 $S_g(m, n)$으로 나타내면, 그 크기는,

$$| S_g(m, n) | = (2 \cdot H + 1) \times (2 \cdot W + 1) \qquad (4)$$

와 같다.

현재 블록 $B_k(m, n)$의 움직임을 예측하기 위한, 이전 프레임의 시간적 인접 블록들의 범위를 나타내는 인덱스 집합을 Θ_w라 하면, 시간적 인접 블록들의 집합 $\Omega_w(m, n)$는,

$$\Omega_w(m, n) = \bigcup_{p, q \in \Theta_w} B_{k-1}(m+p, n+q) \qquad (5)$$

과 같고, 탐색 영역에 포함되는 시간적 인접 블록들이 $\Omega_w(m, n)$에 속하게 된다.

이전 프레임의 시간적 인접 블록들 중에는, 블록의 실제 움직임 (real motion) 방향이 현재 블록 위치로 향하는 것이 있고, 현재 블록과 멀어지는 방향의 것도 있다. 제안한 TPME 방법에서는 전역 탐색 방법을 기반으로 움직임을 예측하고 추정한다. 따라서 현재 블록의 위치를 기준으로, 시간적 인접 블록들의 실제 움직임이 현재 블록 방향인 것만 고려하여 탐색 영역을 예측해 주기 위해서, 시간적 인접 블록의 움직임이 현재 블록 위치와 상반되는 방향이면 이 블록은 현재 블록의 움직임 예측에서 제외시킨다.

본 논문에서는, 움직임 추정시의 탐색점 좌표(움직임 벡터)의 부호를 다음과 같이 정한다. 탐색 영역의 중앙의 탐색점 좌표를 원점으로 두고, 원점을 기준으로 위쪽과 왼쪽 방향을 (-) 부호로 정하고, 아래쪽과 오른쪽 방향을 (+) 부호로 정한다. 실제 움직임이 현재 블록 방향인 것들의 집합을 $\Omega_c(m, n)$이라 하면, 현재 블록 $B_k(m, n)$의 시간적 인접 블록들의 집합 $\Omega_w(m, n)$에 속하는 블록들 중에서, $\Omega_c(m, n)$에 속하는 인접 블록들은,

Procedure *Find* $\Omega_c(m, n)$ *from* $\Omega_w(m, n)$

for ($p, q \in \Theta_w$) *do*
  case $p = q = 0$:
    $B_{k-1}(m+p, n+q) \in \Omega_c(m, n)$
  end case
  case $p = 0$ and $sig(q) = sig(v^h_{p,q})$:
    $B_{k-1}(m+p, n+q) \in \Omega_c(m, n)$
  end case
  case $sig(p) = sig(v^v_{p,q})$ and $q = 0$:
    $B_{k-1}(m+p, n+q) \in \Omega_c(m, n)$
  end case
  case $sig(p) = sig(v^v_{p,q})$ and $sig(q) = sig(v^h_{p,q})$:
    $B_{k-1}(m+p, n+q) \in \Omega_c(m, n)$

end case
end

end *Find* $\Omega_c(m,n)$ *from* $\Omega_w(m,n)$ (6)

과 같이 구한다. 여기서, $v^v_{p,q}$, $v^h_{p,q}$는 각각 $V_{k-1}(m+p, n+q)$의 수직, 수평 성분, 즉,

$$\boldsymbol{V}_{k-1}(m+p, n+q) = [v^v_{p,q}, v^h_{p,q}]^T \quad (7)$$

와 같고, $sig(\cdot)$는 부호를 나타내는 시그넘(sig-num) 함수로서,

$$sig(t) = \begin{cases} 1, & t>0 \\ -1, & t<0 \\ 0, & t=0 \end{cases} \quad (8)$$

와 같다. $S_g(m, n)$의 중심점을 $CP_g(m, n)$이라 하고, 이전 프레임의 $(m+p, n+q)$ 위치의 인접 블록 $B_{k-1}(m+p, n+q)$의 움직임 벡터 $V_{k-1}(m+p, n+q)$로 예측한 국부 탐색 영역 (local search region, LSR)을 $S_l(p, q)$라 하면, $S_l(p, q)$의 중심점 $CP_l(p, q)$는,

$$\boldsymbol{CP}_l(p, q) = \boldsymbol{CP}_g(m, n) + \boldsymbol{V}_{k-1}(m+p, n+q) \quad (9)$$

와 같이 구해지고, 이는 그림 5에서와 같이 표현된다. 이 때에, $S_l(p, q)$의 크기는 $S_g(m, n)$ 크기보다 작다.

제안한 TPME 방법에서는, 기존의 탐색 영역이 움직임의 시간적 연속성을 이용하여 구한 새로운 탐색 영역 $S_{tp}(m, n)$, 즉

$$S_{tp}(m, n) = S_g(m, n) \cap S_l(p, q) \quad (10)$$

으로 바뀌게 된다. 이 방법에서는 FSA 기반으로 탐색 영역을 설정하므로, 그림 5에서와 같이 $S_g(m, n)$에 포함되는 예측 탐색 영역에 대해서만 탐색을 행 한다.

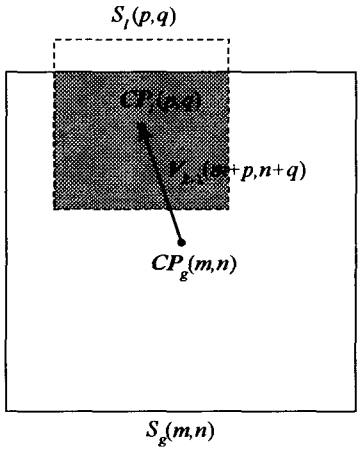
제안한 TPME 방법을 탐색점을 기준으로 표현하면 다음과 같다. 전역 탐색 영역 $S_g(m, n)$의 크기가 $(2 \cdot H+1) \times (2 \cdot W+1)$이므로, $S_g(m, n)$에 대응되는 탐색점 행렬(search point matrix) $\boldsymbol{A}$를,

$$\boldsymbol{A} = \begin{bmatrix} a_{-H,-W} & \cdots & a_{-H,0} & \cdots & a_{-H,W} \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ \cdots & \cdots & a_{0,0} & \cdots & \cdots \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ a_{H,-W} & \cdots & a_{H,0} & \cdots & a_{H,W} \end{bmatrix} \quad (11)$$

와 같이 정의한다. 여기서, 행렬 $\boldsymbol{A}$의 성분 (element), 즉 $a_{i,j}$는 (i, j) 좌표의 탐색점을 나타낸다.

FSA에서는 행렬 $\boldsymbol{A}$의 모든 탐색점들에 대해서 정합을 행하여 움직임 벡터를 구한다. 그러나 제안한 방법에서는 기본적으로 전역 탐색 영역(GSR)을 행렬 $\boldsymbol{A}$와 같이 정해 두고, 이 중에서 시간적 인접 블록들의 움직임 벡터로써 예측된 영역에 대해서만 탐색을 행한다. 먼저 행렬 $\boldsymbol{A}$의 모든 성분들을 '0'으로 리셋 (reset) 시켜 둔다. 그리고, 시간적 인접 블록들에 대해서 예측한 탐색 영역에 포함되는 요소들(탐색점들)만 '1'로 셋 (set)하여 최종 움직임 탐색 행렬을 구성하고, 이들에 대해서만 탐색을 행한다.

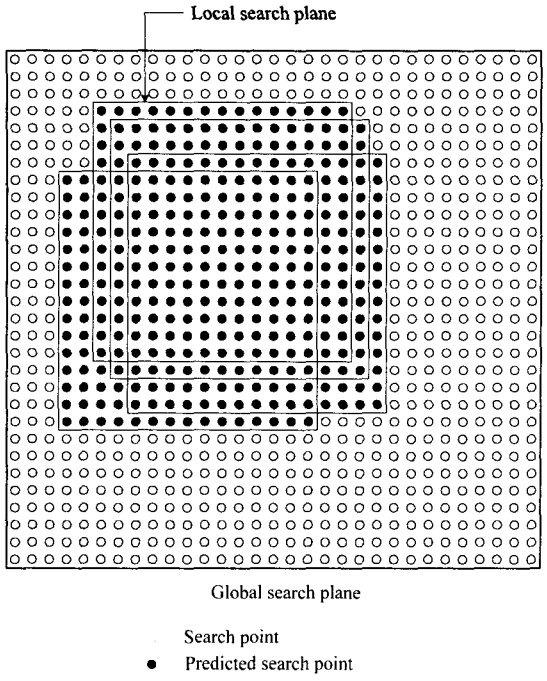
$$a_{i,j} = \begin{cases} 1, & a_{i,j} \text{ is selected as a predicted search point} \\ 0, & a_{i,j} \text{ is not selected} \end{cases} \quad (12)$$



predicted search region, $S_{tp}(m,n)$
$S_g(m,n)$ GSR of $B_k(m,n)$
$\boldsymbol{CP}_g(m,n)$ center point of $S_g(m,n)$
$S_l(p,q)$ LSR corresponding to $B_{k-1}(m+p,n+q)$
$\boldsymbol{CP}_l(p,q)$ center point of $S_l(p,q)$

그림 5. 시간적 인접 블록의 움직임 벡터를 이용한 탐색 영역 예측

탐색점 행렬 $\boldsymbol{A}$를 그림으로 나타내면, 그림 6에서와 같이 모든 점들이 탐색점이며, 이 중 검은색 점들은 시간적 인접 블록들의 움직임 벡터로 예측된 탐색점을 나타내며, 이는 탐색점 행렬 $\boldsymbol{A}$에서 '1'로 셋된 탐색점과 같다. 실제 움직임 추정은 검은색 탐색점들에 대해서만 행한다. 전역 탐색 영역과 국부 탐색 영



(a)

그림 6. 시간적 인접 블록들의 움직임 분포가 (a) 균일한 경우 및 (b) 불규칙한 경우에 대한 제안한 TPME 방법에서의 탐색 영역 예측

역이 겹치는 탐색점에 대해서만 탐색을 행하며, 전역 탐색 범위를 벗어나서 예측되는 탐색점들에 대해서는 탐색을 행하지 않는다. 따라서 제안한 TPME 방법은 적응적인 움직임 추정을 행하므로, 일반적으로 예측된 국부 탐색 영역들이 서로 많이 중첩되므로 탐색해야할 탐색점 수가 줄어 계산량을 줄일 수 있고, 시간적 인접 블록들의 움직임이 불규칙할 경우에는 국부 탐색 영역들이 서로 분산되므로 탐색해야 할 탐색점 수가 증가하여 정확한 움직임 추정을 할 수 있게 된다.

4. 실험결과 및 고찰

제안한 방법에 대한 성능을 확인하기 위하여 모의 실험을 행하였다. 실험 영상으로는, 352×240 크기와 256 밝기의 SIF (source input format) 영상인 FOOTBALL, FLOWER GARDEN, MOBILE, 및 TABLE TENNIS 등의 영상들을 사용하였고, 각 영상의 프레임 수는 60 프레임으로 하였다. FOOTBALL 영상에는 운동 선수의 움직임이 크고 불규칙적인 움직임이 많이 존재하며, FLOWER GARDEN 영상에는 카메라의 패닝 (panning)으로 인해 큰 나무가 일정하게 움직이고 있으며, MOBILE 영상은 고정된 배경에 기차가 움직이는 영상으로서 움직임이 비교적 작고 일정하며, TABLE TENNIS 영상에는 프레임이 진행되면서 카메라의 줌 아웃 (zoom out)이 있으며 선수의 팔 부분과 탁구 라켓의 움직임과 탁구공의 빠른 움직임이 존재한다.

움직임 추정을 위한 블록 크기는 16×16으로 두었으며, 제안한 방법과 기존의 방법의 비교를 위한 탐색범위는 표 1에서와 같이 두었다. 그리고 왜곡 척도 (distortion measure)로는 계산량이 적으면서 성능이 우수한 *MAD*를 사용하였으며, 화질의 척도로 PSNR (peak signal-to-noise ratio)을 이용하였다. 또한 제안한 방법과 기존 방법들과의 계산량 비교는 블록 당 평균 탐색점 수로 비교하였는데, −15~+15 탐색범위의 FSA I 에서 블록 당 평균 탐색점 수는 961 ($=31^2$) 이며, 이를 100 %로 두고서 각 방법들의 상대적인 계산량으로 비교하였다.

실험 영상들에 대하여 기존의 FSA 및 제안한 TPME 방법에 대한 모의 실험 결과를 표 2 및 표 3에 나타내었다.

표 2에서는, 각 방법에 대한 움직임 보상 영상 화질을 객관적인 척도인 평균 PSNR로 비교하였다. FLOWER GARDEN 및 MOBILE 영상에 대해서는 모든 방법들이 거의 같은 결과를 나타내었다. FLOWER GARDEN 영상은 카메라의 패닝이 일어나는 영상으로서, 대부분의 영역에서 큰 움직임 없이 일정한 움직임이 존재하므로, 탐색 영역의 크기와 예측 방법에 관계없이 그 결과가 같게 나타났다. 기차와 공이 느리게 움직이는 MOBILE 영상에 대한 평균 PSNR도, FLOWER GARDEN 영상에 대한 결과와 마찬가지로, 모든 방법에서 같게 나타남을 확인할 수 있다. 그러나 움직임이 크고 불규칙적인 FOOTBALL 영상과 카메라의 줌 아웃과 공의 빠른 움직임이 있는

표 1. 모의 실험에서 기존 방법 및 제안한 방법의 탐색 범위

Algorithms		Search range
Conventional	FSA1	−15~+15
	FSA2	−7~+7
Proposed TPME		GSR −15~+15 LSR −7~+7

표 2. 기존 방법과 제안한 방법의 평균 PSNR 비교

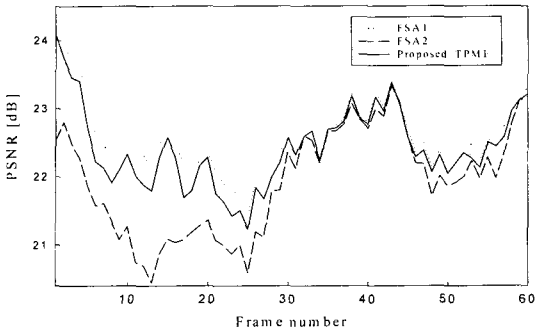
Algorithms / Sequences	Conventional				Proposed TPME
	FSA I	FSA II	IBPME	IFPME	
FOOTBALL	22.59	21.93	22.21	22.20	22.42
FLOWER GARDEN	25.25	25.24	25.24	25.24	25.24
MOBILE	23.43	23.43	23.43	23.43	23.43
TABLE TENNIS	29.48	28.91	29.01	29.03	29.22

표 3. 기존 방법과 제안한 방법의 계산량 비교

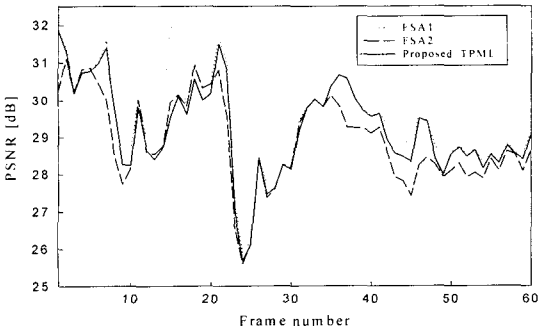
Algorithms / Sequences	Conventional		Proposed TPME
	FSA1	FSA2	
FOOTBALL	100.0	23.4	32.4
FLOWER GARDEN	100.0	23.4	24.9
MOBILE	100.0	23.4	23.7
TABLE TENNIS	100.0	23.4	23.8

TABLE TENNIS 영상에 대해서는 탐색 영역이 작은 FSA II에 비해 제안한 방법이 우수하며 탐색 영역이 큰 FSA I의 평균 PSNR이 근접함을 볼 수 있다. 이는 탐색 영역이 작으면 움직임이 큰 영상에 대해 움직임 추정이 제대로 되지 않지만, 제안한 방법은 움직임의 시간적 연속성을 이용하여 적응적으로 움직임을 추정하므로 탐색 영역을 크게 하지 않아도 효과적인 움직임 추정이 가능함을 확인할 수 있다. 또한 기존의 움직임 예측을 이용한 움직임 탐색 방법인 IBPME 및 IFPME 방법과도 비교하였을 때, 움직임이 적은 영상의 경우 비슷한 결과를 얻을 수 있었지만, 움직임이 많은 FOOTBALL이나 TABLE TENNIS 영상에서는 제안한 방법의 PSNR 성능이 더 우수함을 볼 수 있다. 이는 움직임이 적을 경우 추정으로 인한 탐색영역의 이동 범위가 크지 않기 때문에 유사한 결과를 보이는 데 비해, 움직임이 클 경우 제안한 방법은 움직임의 연속성을 효과적으로 이용하였기 때문에 정확한 탐색 영역의 이동을 보여줌을 알 수 있다. 물론 불규칙한 움직임이 있을 경우 제안한 방법은 탐색 영영 또한 사방으로 넓어지므로 기존 방법들의 정해진 크기의 탐색 영역에서 보다 정확한 탐색이 가능하다.

그림 7은 움직임이 큰 FOOTBALL 및 TABLE TENNIS 영상 60 프레임에 대한 연속적인 PSNR 비교를 그래프로 나타내고 있다. 제안한 방법은 탐색



(a)

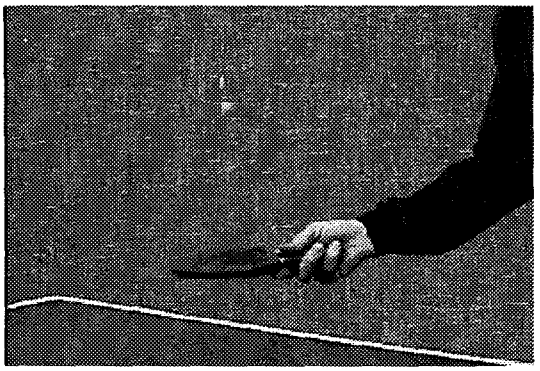


(b)

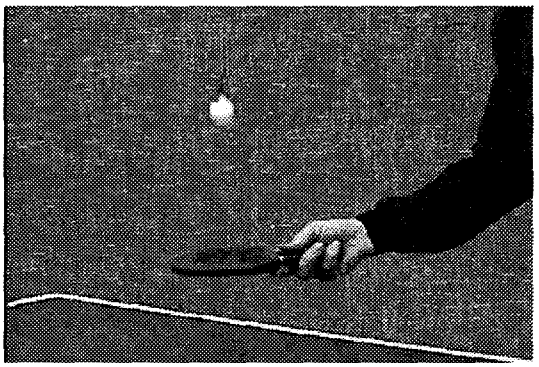
그림 7. (a) FOOTBALL 및 (b) TABLE TENNIS 영상에 대한 FSA와 제안한 TPME 방법의 PSNR 비교

영역의 최고 상한인 FSA 1 방법에 근접한 성능을 보였으며, 움직임이 큰 프레임인 FOOTBALL 10~20 프레임 그리고 TABLE TENNIS 40~50 프레임에서 더욱 좋은 결과를 보임을 알 수 있었다.

그림 8에서는 FSA 2 방법과 제안한 방법으로 움직임 보상한 TABLE TENNIS 영상의 한 프레임을 보여주고 있다. 움직임이 큰 탁구공 부분이 FSA 2 방법으로 탐색하였을 경우 정방형의 탐색 영역을 벗어나서 제대로 보상이 되지 못한 반면, 제안한 TPME 방법으로 보상하였을 경우 이전 프레임의 탁구공 움직임을 따라가서 탐색 영역이 정해지므로 비교적 정확한 탐색이 이루어 졌음을 볼 수 있다.



(a)



(b)

그림 8. (a) FSA 및 (b) 제안한 TPME 방법으로 움직임 보상 된 TABLE TENNIS 영상

표 3에서는, 각 방법의 계산량을 FSA1 방법에 대한 상대적인 백분율로 비교하였다. FSA I 방법에 대한 블록 당 평균 탐색점 수를 100%로 두었을 때, 각 방법들에 대한 블록당 평균 탐색점 수를 상대적으로 나타내었다. 제안한 TPME 방법에 대한 결과를 전체적으로 보면, 모든 시간적 인접 블록들의 움직임 벡터를 고려하여 이들 각각에 대해서 $-7\sim+7$ 크기의 LSR을 부여하기 때문에 그와 같은 크기의 단일 탐색 영역을 갖는 FSA II에 비해서 계산량은 다소 증가하지만, $-15\sim+15$ 크기의 탐색 영역을 갖는 FSA I에 비해서는 계산량이 매우 감소함을 볼 수 있다. FLOWER GARDEN, MOBILE 및 TABLE TENNIS 영상에 대해서는, 제안한 TPME 방법의 계산량이 기존의 방법들과 거의 같다. FLOWER GARDEN 및 MOBILE 영상의 경우, 모든 영역에 걸쳐서 움직임이 작고 균일하게 분포한다. 따라서 모든 인접 블록들에 대한 LSR이 GSR 내에서 대부분 겹치게 되며, 실제 움직임 추정을 행하는 탐색 영역은 거의 $-7\sim+7$ 탐색 범위가 된다. TABLE TENNIS 영상에서는 움직임이 일어나는 영역이 전체 영상에서 차지하는 비율이 아주 낮다. 그러므로 FLOWER GARDEN 및 MOBILE 영상에 대한 결과와 마찬가지로, 대부분의 영역에서 움직임이 없거나 작으므로, 실제 움직임 추정을 행하는 탐색 영역의 크기는 거의 $-7\sim+7$ 이 된다. 그러나 FOOTBALL 영상에 대한 결과를 보면, 계산량이 다른 영상에서 보다 다소 많이 증가함을 확인할 수 있다. 이는 FOOTBALL 영상의 경우, 움직임이 크고 불규칙한 영역이 많이 분포하므로, 시간적 인접 블록들의 움직임에 의해서 예측된 LSR의 위치가 GSR 내에서 분산된다. 그러므로 GSR과 LSR이 겹치는 부분이 커지게 되어 넓은 탐색 영역에 대해서 움직임 추정을 행하게 되어 탐색점 수가 증가하지만 이와 더불어서 움직임 보상 영상의 화질도 개선됨을 확인할 수 있다.

즉, 제안한 방법은 LSR과 같은 탐색 영역을 가지는 FSA II에 근접한 계산량으로 GSR과 같은 탐색 영역을 가지는 FSA I에 근접한 PSNR 성능을 보였으며, 움직임이 불규칙하고 큰 영상에 대해서는 움직임 탐색을 세밀히 하고 반대의 경우 계산량을 줄이는 적응적인 움직임 추정을 함을 알 수 있다.

5. 결 론

본 논문에서는, 움직임의 시간적 연속성을 이용하여 현재 블록의 움직임특성을 예측하고, 이를 이용하여 적응적으로 움직임을 추정하는 방법을 제안하였다. 블록의 움직임이 시간적으로 연속된다는 특성을 이용하여, 현재 블록의 탐색 영역을 구성하는 이전 프레임의 모든 인접 블록들의 움직임 벡터로써, 적절한 탐색 영역을 예측한다. 이전 프레임의 각 블록들의 움직임은 프레임이 진행되면서 현재 프레임에서도 연속될 확률이 높다. 즉, 현재 블록의 시간적 인접 블록들은 이전 프레임에서의 자기 자신의 움직임 벡터만큼 연속적으로 움직이는 특성이 있다. 그러므로 제안한 TPME 방법에서는, 넓은 전역 탐색 영역을 설정하고 시간적 인접 블록들의 움직임을 이용하여 국부 탐색 영역을 예측하며, 움직임 추정은 전역 탐색 영역과 국부 탐색 영역이 겹치는 탐색점들에 대해서만 행한다.

모의 실험 결과 제안한 방법은 일반적인 전역 탐색 방법에 비해 탐색영역을 크게 하지 않아도 우수한

PSNR 결과를 보였다. 즉 계산량을 줄일 수 있어 고속 움직임 탐색을 행하는 부호화기에 적합함을 알 수 있었다.

참 고 문 헌

[1] B. G. Haskell, A. Puri, and A. N. Netravali, *Digital Video: An Introduction to MPEG-2*, New York, NY: Chapman and Hall, 1997.

[2] K. R. Rao and J. J. Hwang, *Techniques and Standards for Image, Video, and Audio Coding*, Upper Saddle River, NJ: Prentice Hall, 1996.

[3] J. Lee and B. W. Dickinson, "Temporally adaptive motion interpolation exploiting temporal masking in visual perception," *IEEE Trans. Image Process.*, vol. 3, no. 5, Sept. 1994.

[4] L. Luo, C. Zou, X. Gao, and Z. He, "A new prediction search algorithm for block motion estimation in video coding," *IEEE Trans. Consumer Electronics*, vol. 43, no. 1, Feb. 1997.

[5] Y.-Q. Zhang and S. Zafar, "Predictive block-matching motion estimation for TV coding--Part II: Inter-frame prediction," *IEEE Trans. Broadcasting*, vol. 37, no. 3, pp. 102-105, Sept. 1991.

[6] C.-H. Hsieh, P.-C. Lu, J.-S. Shyn, and E.H. Lu, "Motion estimation algorithm using interblock correlation," *IEE Electronics Letters*, vol. 26, No. 5, Mar. 1990.

[7] N. Haddadi, Navid, and C.-C. Jay Kuo, "Fast computation of motion vectors for MPEG," *Proc. of SPIE: Visual Comm. and Image Proc.*, vol. 2094, Oct. 1993.

[8] Y.-Q. Zhang and S. Zafar, "Predictive block-matching motion estimation for TV coding--Part II: Inter-frame prediction," *IEEE Trans. Broadcasting*, vol. 37, no. 3, pp. 102-105, Sept. 1991.

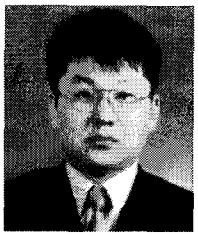
[9] J. Lee and B. W. Dickinson, "Temporally adaptive motion interpolation exploiting temporal masking in visual perception," *IEEE Trans. Image Process.*, vol. 3, no. 5, Sept. 1994.

이 경 환

1994년 2월 경북대학교 전자공학과 졸업(공학사)
1996년 2월 경북대학교 대학원 전자공학과 졸업(공학석사)
2000년 8월 경북대학교 대학원 전자공학과 졸업(공학박사)
2001년 3월~현재 위덕대학교 멀티미디어공학과 조교수
관심분야 : 영상 및 음향신호처리 및 압축, 멀티미디어 프로그래밍

류 권 열

1982년 8월 경북대학교 전자공학과 졸업(공학사)
1990년 8월 경북대학교 산업공학과 졸업(공학석사)
1998년 2월 부경대학교 전자공학과 졸업(공학박사)
1982년 8월~1986년 5월 삼성반도체통신(주) 연구원
1986년 6월~1995년 4월 포항공과대학교 전자계산소
1998년 4월~현재 위덕대학교 멀티미디어공학과 조교수
관심분야 : 디지털영상처리, 멀티미디어정보검색



최 정 현

1991년 2월 경북대학교 전자공학과 졸업
1993년 2월 경북대학교 대학원 전자공학과 졸업(공학석사)
2000년 2월 경북대학교 대학원 전자공학과 졸업(공학박사)
2000년 9월~2001년 3월 기술신용보증기금 부산기술평가센터 차장
2001년 3월~현재 기술신용보증기금 대구기술평가센터 차장
관심분야 : 영상신호처리 및 압축

교 신 저 자

이 경 환 780-713 경북 경주시 강동면 산 50번지 위덕대학교 컴퓨터멀티미디어공학부