적응적 탐색 영역 예측을 이용한 고속 움직임 추정

류 권 열*

Fast Motion Estimation using Adaptive Search Region Prediction

Kwon-yeol Rru*

요 약

본 논문은 적응적 탐색 영역과 새로운 3단계 탐색을 이용하는 고속 움직임 추정을 제안한다. 제안한 방법은 이웃 블록의 모션벡터로부터 현재 블록의 움직임를 예측하고, 예측된 움직임 정보를 이용하여 탐색 영역을 적응적으로 설정함으로써 움직임 보상 영상의 화질이 0.43dB ~ 2.19dB 향상되었다. 또한 새로운 3단계 탐색 패턴을 적용하여 블록 당 계산량을 기존의 방법에 비해 1.3% ~ 1.9% 감소시킴으로써 고속 움직임 추정이 가능함을 알 수 있었다.

ABSTRACT

This paper proposes a fast motion estimation using an adaptive search region and a new three step search. The proposed method improved in the quality of motion compensation image as 0.43dB ~ 2.19dB, according as it predict motion of current block from motion vector of neigher blocks, and adaptively set up search region using predicted motion information. We show that the proposed method applied a new three step search pattern is able to fast motion estimation, according as it reduce computational complexity per blocks as 1.3% ~ 1.9% than conventional method.

키워드

adaptive search region, block matching algorithm, motion vector, motion estimation

Ⅰ. 서론

동영상 신호에는 공간적 및 시간적 중복성이 존재하는데, 이러한 중복성을 효과적으로 제거함으로써 부호화 효율을 높인다. 공간적으로 인접하는 화소들은 서로 상관성을 가지는데, 대부분의 에너지가 저주파 성분에 분포하는 자연 영상의 경우에는 이웃 화소들 간의 유사한 밝기값을 갖는 특성을 나타내고, 이러한 특성을 이용하여 공간적 중복성을 제거한다. 시간적 중복성이란 동영상을 구성하는 인접 프레임들 간에 존재하는 중복성을 의미하며, 이전 프레임과 현재 프레임에 대한 움직임을 추정하여 예측 영상을 생성한 후, 현재 프레임과 예측 영상의 차이를 부호화함으로써 시간적 중복성을 제거한다. 동영상 부호화에서 움직임 추정에는 많은 계산량이 요구되기 때문에 실시간 시스템 구현 등을 위해서는 계산량 감소가 필수적이다. 움직임 추정에 소요되는 계산량 감소를 위한 연구에서 블록 매칭 알고리듬(block matching algorithms, BMA)은 대표적인 분야 중의 하나이며, 이는 H.261, H.263, MPEG 등과 같은 동영상 표준에 적용되고 있다[1,2].

BMA는 입력 영상을 매크로블록(macro block) 단위로 분할 한 후, 현재 프레임의 블록과 가장 유사한 블록을

* 위덕대학교 소프트웨어공학부

접수일자 2008. 02. 26

이전 프레임에서 탐색하고, 두 블록간의 움직임 정보를 이용하여 모션벡터를 생성한다. BMA를 기반으로 하는 연구로는 전역 탐색(full search, FS), 3단계 탐색(three step search, TSS), 교차 탐색(cross search, CS), 4단계 탐색(four step search, FSS) 등이 있으며, 전역 탐색은 탐색 영역의 모든 탐색점을 대상으로 움직임 정보를 계산하므로 영상의 화질은 우수하지만, 계산량이 너무 많아 실시간 시스템 구현에 어려움이 있다[3-8]. 3단계 탐색은 움직임 추정에 소요되는 계산량을 가장 효과적으로 감소시키는 방법으로 평가되고 있으나 단일모드 에러 표면 가정(unimodal error surface assumption, UESA)을 만족하지 못할 때는 최적의 움직임 추정이 어렵고, 움직임 보상 영상 화질이 전역 탐색보다 떨어지는 단점이 있다. 따라서 본 논문에서는 탐색 영역을 적응적으로 예측하여 최적의 움직임을 추정함으로써 움직임 보상 영상의 화질을 향상시키고, 새로운 3단계 탐색을 이용하여 탐색점의 수를 줄임으로써 움직임 추정에 소요되는 계산량을 감소시키는 방법을 제안한다. 본 논문은 2장에서 기존의 3단계 탐색을 설명하고, 3장에서 적응적 탐색 영역 예측을 이용하는 제안한 3단계 탐색을 기술하며, 4장 및 5장에서 실험 결과 및 결론을 나타낸다.

Ⅱ. 기존의 3단계 탐색

BMA 기반 3단계 탐색은 현재 프레임의 각 블록과 가장 유사한 블록을 이전 프레임의 탐색 영역 내에서 3단계로 탐색하여 찾은 후, 두 블록간의 움직임 정보를 이용하여 모션벡터를 생성한다. 두 블록 간의 유사도 측정 기준은 계산량이 적은 MAD(mean absolute difference)를 사용하며, 식 (1)과 같다.

$$MAD(x,y)=\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}\left|\begin{array}{l}H_c(u+i,v+j)\\-H_p(u+x+i,v+y+j)\end{array}\right| \quad (1)$$

식 (1)에서 H_c 및 H_p는 현재 프레임 및 이전 프레임의 화소값을 나타내고, u, v는 현재 블록의 좌측상단 좌표를 나타내며, x, y는 현재 블록이 이동한 거리이며, MAD가 최소 일 때의 x, y 값을 현재 블록의 움직임 정보로 추정한다. 현재 프레임과 이전 프레임의 움직임 정보에 대한 상관도는 그림 1과 같다.

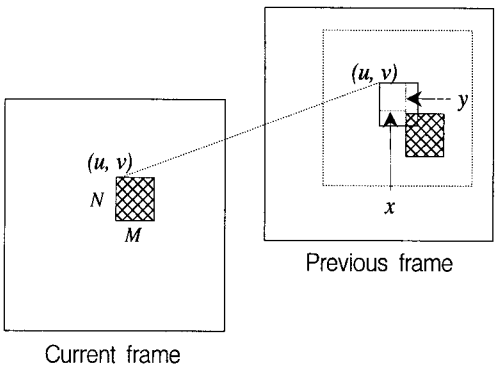

그림 1. 움직임 정보의 상호 관계
Fig.1 Correlation of motion information

일반적인 3단계 탐색에서 탐색 영역의 크기가 -7~+7인 경우, 첫 단계는 탐색 영역의 중심을 기준으로 9개의 탐색점을 비교하여 최적의 탐색점을 찾고, 두 번째 단계는 첫 번째 단계에서 추정된 최적의 탐색점을 중심으로 8개의 탐색점을 비교하여 최적의 탐색점을 찾는다. 그리고 세 번째 단계는 두 번째 단계에서 추정된 최적의 탐색점을 중심으로 8개의 탐색점을 비교하여 최적의 탐색점을 찾은 후, 현재 블록과의 움직임 정보를 이용하여 모션벡터를 생성한다. 탐색 영역의 크기가 -7~+7인 경우 전역 탐색은 225개의 탐색점에 대한 계산을 수행하지만, 3단계 탐색은 25개의 탐색점에 계산을 수행하므로 전역 탐색에 비해 탐색 시간이 빠르기 때문에 실시간 시스템 구현이 용이하다. 일반적인 3단계 탐색 패턴은 그림 2와 같고, ❸, ③, 3으로 표시된 부분은 각 단계에서 MAD 값이 최소인 탐색점을 낙타낸다.

3단계 탐색 중에서 ESS(efficient simple search)는 탐색점의 수를 줄이기 위해 각 단계마다 ❷, ❶, ❸ 탐색점에 대한 MAD를 계산한 후, MAD 값의 결과에 따라 네 가지 탐색 패턴으로 구분하고, 네 가지 경우에 따라 1~3개의 탐색점을 추가하여 비교한다.

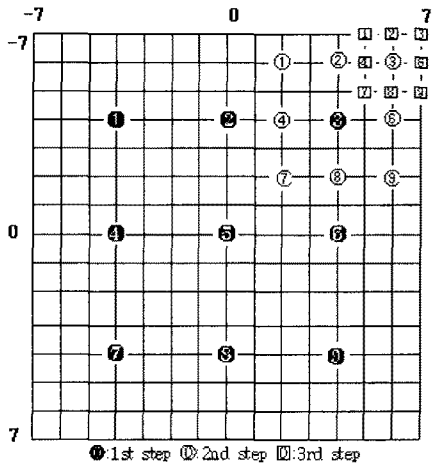

그림 2. 3단계 탐색 과정
Fig. 2 Three step search processing

기존의 방법인 ESS의 탐색 패턴은 그림 3과 같다.

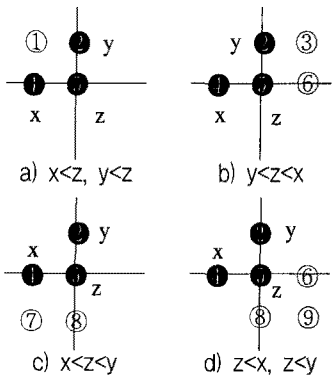

그림 3. ESS의 탐색 패턴
Fig. 3 Search pattern of ESS

그림 3에서 x, y, z 는 ❶, ❷, ❺ 탐색점에서 MAD 값을 나타내며, 기호 ① 등은 MAD 값 비교 결과에 따라 추가되는 탐색점을 나타낸다. 기존의 방법인 ESS의 탐색 순서는 그림 4와 같다. 그림 4에서와 같이 ESS는 단계별 평균 탐색점 수가 다섯 개이므로, 블록 당 소요되는 전체 탐색점 수는 15개임을 알 수 있다.

```
Procedure
 do computation MAD(2), MAD(4), MAD(5)
  case MAD(4)<MAD(5)&MAD(2)<MAD(5)
   check MAD(1) & select min(MAD())
  end case
  case MAD(2)<MAD(5)<MAD(4)
   check MAD(3)&MAD(6)
   select min(MAD())
  end case
  case MAD(4)<MAD(5)<MAD(2)
   check MAD(7)&MAD(8)
   select min(MAD())
  end case
  case MAD(5)<MAD(4)&MAD(5)<MAD(2)
   check MAD(6)&MAD(8)&MAD(9)
   select min(MAD())
  end case
 do end & go next step
end
```

그림 4. ESS의 탐색 순서
Fig. 4 Search procedure of ESS

Ⅲ. 제안한 3단계 탐색

3.1 적응적 탐색 영역 예측

현재 블록과 공간적으로 인접한 이웃 블록들이 동일한 객체에 위치한다면, 이웃 블록의 상호 관계로부터 현재 블록의 움직임 예측이 가능하며, 이웃 블록들의 상호 관계는 그림5와 같이 나타낸다. 그림 5에서 블록 A의 모션벡터를 $\mathbf{A}$, 블록 B의 모션벡터를 $\mathbf{B}$, 블록 T의 모션벡터를 $\mathbf{T}$라고 할 때, 각 블록의 모션벡터는 식 (2), 식 (3), 식 (4)와 같이 나타낼 수 있다.

$$\mathbf{A} = a_x \mathbf{i} + a_y \mathbf{j} \tag{2}$$

$$\mathbf{B} = b_x \mathbf{i} + b_y \mathbf{j} \tag{3}$$

$$\mathbf{T} = t_x \mathbf{i} + t_y \mathbf{j} \tag{4}$$

식 (2), 식(3) 및 식(4)에서 a_x, b_x, t_x는 x 방향의 움직임 정보를 나타내고, b_y, b_y, t_y는 y 방향의 움직임 정보를 나타내며, 모션벡터 $\mathbf{A}$, $\mathbf{B}$, $\mathbf{T}$는 현재 프레임과 이전 프레임 사이의 단위 시간당 객체의 이동 속도를 나타낸다.

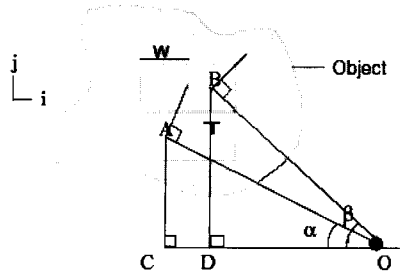

그림 5. 이웃 블록의 모션벡터 상호 관계
Fig. 5 Correlation of motion vector for neighbor blocks

그림 5에서 이웃 블록 A, B, T가 동일한 객체에 위치한다면, 블록 A의 이동 속도 ω_a 와 블록 B의 이동 속도 ω_b 가 동일하며, OD 및 OB는 식 (5) 및 식 (6)과 같다.

$$OD = OA\cos\alpha - W = OB\cos\beta \quad (5)$$

$$BD = OA\sin\alpha + W = OB\sin\beta \quad (6)$$

식 (5) 및 식 (6)으로부터 OB와 OA는 식 (7) 및 식 (8)과 같이 표현된다.

$$OB = \frac{W(\sin\alpha+\cos\alpha)}{\sin(\beta-\alpha)} \quad (7)$$

$$OA = \frac{W(\sin\beta+\cos\beta)}{\sin(\beta-\alpha)} \quad (8)$$

그리고 모션벡터 ..와 .. 의 외적은 식 (9)와 같다.

$$\text{x} = \| \| \| \sin(\beta-\alpha)\text{k}, \quad (9)$$

where k= x

식 (7), 식 (8) 및 식 (9) 로부터 블록 A의 이동 속도 ω_a 와 블록 B의 이동 속도 ω_b 는 식 (10) 및 식 (11)과 같이 표현된다.

$$\|\omega_a\| = \frac{\|A\|}{OA} = \frac{(B\times A)\cdot \text{k}}{W\|B\|(\sin\beta+\cos\beta)} \quad (10)$$

$$= \frac{(\text{B}\times\text{A})\cdot\text{k}}{W(b_x+b_y)}$$

$$\|\omega_b\| = \frac{\|B\|}{OB} = \frac{(B\times A)\cdot \text{k}}{W\|A\|(\sin\alpha+\cos\alpha)} \quad (11)$$

$$= \frac{(\text{B}\times\text{A})\cdot\text{k}}{W(a_x+a_y)}$$

블록 A와 B가 동일 객체에 위치할 때, $\|\omega_a\|$와 $\|\omega_b\|$는 동일한 값을 가지므로, 식 (10) 및 식 (11)에서 $(a_x+a_y)=(b_x+b_y)$ 임을 알 수 있다. 그러므로 현재 블록의 모션벡터는 ..는 이웃블럭 모션벡터의 구성 요소인 a_x 및 b_y를 이용하여 식 (12)와 같이 나타 낼 수 있다.

$$.. = a_x + b_y \quad (12)$$

따라서 본 논문에서는 이웃 불록의 모션벡터로부터 현재 블록의 움직임 정보를 예측하고, 예측된 움직임 정보를 이용하여 탐색 영역을 적응적으로 설정함으로써 움직임 보상 영상의 화질을 향상시키는 방법을 제안한다.

3.2 제안한 3단계 탐색 패턴

본 논문에서는 각 단계마다 ❺, ❻, ❽ 탐색점에 대한 MAD를 계산한 후, MAD 값의 결과에 따라 네 가지 탐색 패턴으로 구분한 후, 각 경우에 따라 1~1.5개의 탐색점을 추가하는 3단계 탐색을 제안한다. 제안한 방법의 탐색 패턴은 그림 6과 같고, 여기서 x, y, z는 각 탐색점에서 계산한 MAD 값을 나타낸다. 그림 6에서와 같이 탐색 구간이 7일 때 첫 번째 단계에서 평균 탐색점의 수는 4.13 ((=4+4+4+4.5)/4) 개이며, 두 번째 및 세 번째 단계에서 탐색점의 수는 각각 3.13((=3+3+3+3.5)/4)개 이므로, 하나의 블록에 대한 움직임을 추정하기 위해 필요한 탐색점의 수는 10.39개이다. 따라서 제안한 방법은 일반적인 3단계 탐색에 비해서는 2.4(=25/10.39) 배의 탐색 시간이 감소하고, 기존의 방법인 ESS에 비해서는 1.45(15/10.39) 배의 탐색 시간이 감소하며, 전역 탐색 방법에 비해서는 21.66 (=225/10.39) 배의 탐색 시간이 감소함을 알 수 있다.

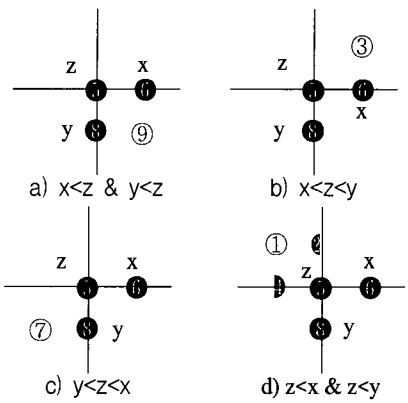

그림 6. 제안한 방법의 탐색 패턴
Fig. 6 Search pattern of proposed method

제안한 3단계 탐색 에 대한 순서는 그림 7과 같다.

```
Procedure
 do computation MAD(5), MAD(6), MAD(8)
  case MAD(6)<MAD(5)&MAD(8)<MAD(5)
   check MAD(9) & select min(MAD())
  end case
  case MAD(6)<MAD(5)<MAD(8)
   check MAD(3) & select min(MAD())
  end case
  case MAD(8)<MAD(5)<MAD(6)
   check MAD(7) & select min(MAD())
  end case
  case MAD(5)<MAD(6)&MAD(5)<MAD(8)
   if MAD(5)<MAD(6)<MAD(8)
    if MAD(2)<MAD(5)
     check MAD(1) & select min(MAD())
    else min(MAD())=MAD(5)
    end if
   if MAD(5)<MAD(8)<MAD(6)
    if MAD(4)<MAD(5)
     check MAD(1) & select min(MAD())
    else min(MAD())=MAD(5)
    end if
  end case
 do end & go next step
end
```

그림 7. 제안한 방법의 탐색 순서
Fig. 7 Search procedure of proposed method

Ⅳ. 실험 결과

제안한 방법에 대한 실험은 352×240 크기의 FOOTBALL, FLOWER GARDEN, MOBILE 및 TABLE TENNIS 영상을 사용하였고, 각 영상의 프레임 수는 60 프레임으로 하였다. FOOTBALL 영상은 움직임이 크고 불규칙적이며, FLOWER GARDEN 영상은 움직임이 작고 느리며, MOBILE 영상은 고정된 배경에 기차가 일정하게 움직이며, TABLE TENNIS 영상은 움직임이 빠른 특징이 있다. 움직임 추정을 위한 블록 크기는 16x16으로 하였으며, 탐색 영역은 -7~+7로 지정하였다. 블록 유사도 측정은 비교적 계산량이 적은 MAD를 사용하였고, 화질의 척도는 PSNR을 사용하였다. 계산량 비교는 블록당 평균 탐색점 수를 사용하였다.

제안한 방법에서 움직임 보상 영상의 화질에 대한 실험 결과는 그림 8과 같았다.

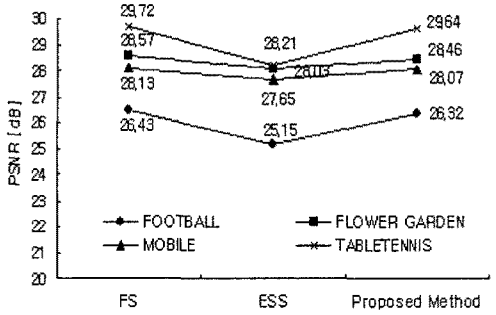

그림 8. 움직임 보상 영상의 평균 PSNR 비교
Fig. 8 Comparison of average PSNR for motion compensation image

그림 8에서와 같이 움직임이 느리고 일정한 FLOWER GARDEN 및 MOBILE 영상은 기존의 방법인 ESS에 비해 움직임 보상 영상의 화질이 각각 0.43dB 및 0.42dB 향상되고, 움직임이 불규칙적이고 빠른 FOOTBALL 및 TABLE TENNIS 영상은 각각 1.17dB 및 2.19dB 향상됨으로써 전역탐색에 근접하는 화질을 나타내었다. 따라서 제안한 방법은 이웃 블록의 모션벡터로부터 현재 블록의 움직임을 예측하고, 예측된 움직임 정보를 이용하여 탐색 영역을 적응적으로 설정함으로써 움직임 보상 영상의 화질이 각각 0.43dB~2.19dB 개선됨을 알 수 있다.

움직임이 불규칙적인 FOOTBALL 영상에 대한 연속적인 PSNR 비교는 그림 9에서와 같았으며, 움직임이 큰 10~20 프레임에서 더욱 좋은 결과가 나타남을 알 수 있다.

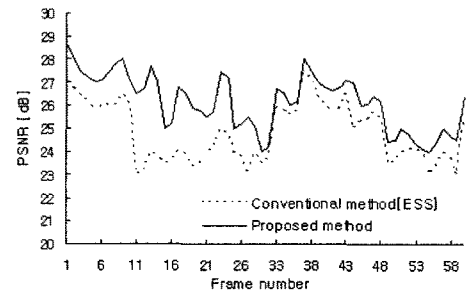

그림 9. FOOTBALL 영상의 연속적인 PSNR 비교
Fig. 9 Comparison of sequential PSNR for FOOTBALL image

제안한 방법에 대한 계산량 비교는 전역 탐색에서 한 블록 당 소요되는 평균 탐색점 수, 즉 225(=15^2)를 100%로 설정하고, 상대적인 계산량을 비교하였으며, 실험 결과는 표 1과 같았다.

표 1. 제안한 방법에 대한 계산량 비교
Table. 1 Comparison of computational complexity for proposed method

Method / Sequences	Conventional(%)		Proposed (%)
	FS	ESS	
FOOTBALL	100.0	7.0	5.6
FLOWER GARDEN	100.0	7.0	5.3
MOBILE	100.0	7.0	5.1
TABLE TENNIS	100.0	7.0	5.7

표 1에서와 같이 기존의 방법인 ESS는 블록 당 소요되는 탐색점 수가 15개 이고, 전역 탐색에 대한 상대적인 계산량이 7% 요구되었고, 제안한 방법은 블록 당 소요되는 탐색점 수가 10~13개 이고, 전역 탐색에 대한 상대적인 계산량이 5.1%~5.7% 요구되었다. 따라서 제안한 방법은 기존의 방법인 ESS에 비해 블록 당 상대적인 계산량이 1.3%~1.9% 감소함을 알 수 있다.

V. 결 론

BMA 기반 3단계 탐색은 UESA을 만족하지 못할 때는 최적의 움직임 추정이 어렵고, 움직임 보상 영상의 화질이 전역 탐색에 비해 떨어지는 단점이 있다. 따라서 본 논문에서는 탐색 영역을 적응적으로 예측하여 최적의 움직임을 추정함으로써 움직임 보상 영상의 화질을 향상시키며, 새로운 3단계 탐색을 이용하여 탐색점의 수를 줄임으로써 움직임 추정에 소요되는 계산량을 감소시키는 방법을 제안하였다. 실험 결과 움직임이 느리고 일정한 FLOWER GARDEN 및 MOBILE 영상은 기존의 방법인 ESS에 비해 움직임 보상 영상의 화질이 각각 0.43dB 및 0.42dB 향상되고, 움직임이 불규칙적이고 빠른 FOOTBALL 및 TABLE TENNIS 영상은 각각 1.17dB 및 2.19dB 향상되었다. 또한 제안한 방법은 새로운 3단계 탐색 패턴을 적용하여 블록 당 계산량을 기존의 방법에 비해 1.3%~1.9% 감소시킴으로써 고속 움직임 추정이 가능함을 알 수 있으며, MPEG 등 동영상 실시간 부호화에 유용하게 활용될 수 있다.

참고문헌

[1] Bede Liu and Andre Zaccarin, "New fast algorithms for estimation of block motion vectors," IEEE Transactions CASVT, vol.3, no.2, pp.148-157, Apr. 1993.

[2] F. Dufaux and F. Moscheni, "Motion estimation techniques for digital TV: A review and a new contribution," Proceeding of the IEEE, vol.83, no.6, pp.858-876, June 1995.

[3] M. Ghanbari, "The cross search algorithm for motion estimation," IEEE Transactions Communications, vol.38, no.7, pp.950-953, July 1990.

[4] R. Li, B. Aeng, and Ming L. Liou, "A new three step search algorithm for block motion estimation," IEEE Transactions CASVT, vol.4, no.4, pp.438-442, Aug. 1994.

[5] M. J. Chen, L. G. Chen, and T. D. Chiueh, "One-dimensional full search motion estimation algorithm for video coding," IEEE Transactions CASVT, vol.4, no.5, pp.504-509, Oct. 1994.

[6] M. P. Lai and W. C. Ma, "A novel four-step search algorithm for fast block motion estimation," IEEE Transactions CASVT, vol.6, no.3, pp.313-317. June 1996.

[7] J. Lu and M. L. Liou, "A simple and efficient search algorithm for block matching motion estimation," IEEE transactions CASVT, vol.7, pp.429-433, Apr. 1997.

[8] D. H. Yu, S. K. Jang, and J. B. Ra, "Fast motion estimation for shape coding in MPEG-4," IEEE Transactions CASVT, vol.13, no.4, pp.358-363, Apr. 2003.

저자소개

류권열(Kwon-yeol Ryu)

1982년 경북대학교 전자공학과(공학사)
1990년 경북대학교 산업공학과(공학석사)
1998년 부경대학교 전자공학과(공학박사)

1986년~1995년 포항공과대학교 전자계산소
1996년 ~ 현재 위덕대학교 소프트웨어공학부 부교수
※관심분야 : 디지털영상처리, 멀티미디어저작권보호, 3D모델링, 게임프로그래밍