시·공간적 주변 블록들의 움직임을 이용하여 적응적으로 탐색 범위 조절을 하는 고속 움직임 추정

이상학*

Fast Motion Estimation with Adaptive Search Range Adjustment using Motion Activities of Temporal and Spatial Neighbor Blocks

Sang-hak Lee*

요 약

시·공간적으로 인접한 주변 블록들의 움직임 정보들을 이용하여 적응적으로 탐색 범위를 조절하는 고속 움직임 추정 기법을 제안한다. 기존의 탐색 범위 조절을 통한 고속 움직임 추정 기법들은 참조 프레임에 있는 모든 블록들의 움직임 벡터들 중에서 최댓값을 이용하여 현재 블록의 탐색 범위 조절을 하기 때문에 움직임이 작은 블록들에 대해서는 최적의 탐색 범위 조절을 못할 수 있다. 본 논문에서는 참조 프레임과 현재 프레임에서 현재 블록과 인접한 주변 블록들의 움직임 정보를 동시에 고려하여 최적의 탐색 범위를 결정하여 움직임 추정을 고속으로 할 수 있는 움직임 추정 기법을 제안한다. 시간적으로 인접된 블록들 즉 참조 프레임에 있는 주변 블록들의 움직임 정보를 이용하면 움직임이 작은 블록의 탐색 범위의 크기를 적절하게 작게 할 수 있어 고속의 움직임 추정을 할 수 있다. 실험을 통하여 본 논문에서 제안한 시·공간적 주변 블록들의 움직임 정보를 이용하는 탐색 범위 조절을 통한 고속 움직임 추정 기법이 비트율과 움직임 추정 오차는 거의 비슷하게 유지하면서 기존의 방법인 Simple Dynamic Search Range(SDSR) 기법보다 탐색 점의 수를 약 15% 더 감소시킬 수 있음을 확인하였다.

ABSTRACT

This paper propose the fast motion estimation algorithm with adaptive search range adjustment using motion activities of temporal and spatial neighbor blocks. The existing fast motion estimation algorithms with adaptive search range adjustment use the maximum motion vector of all blocks in the reference frame. So these algorithms may not control a optimum search range for slow moving block in current frame. The proposed algorithm use the maximum motion vector of neighbor blocks in the reference frame to control a optimum search range for slow moving block. So the proposed algorithm can reduce computation time for motion estimation. The experiment results show that the proposed algorithm can reduce the number of search points about 15% more than Simple Dynamic Search Range(SDSR) algorithm while maintaining almost the same bit-rate and motion estimation error.

키워드

H.264/AVC, MPEG-4, motion estimation, search range

* 동양대학교 정보통신공학부(lsh@dyu.ac.kr)
접수일자 : 2010. 06. 15 심사(수정)일자 : 2010. 07. 05 게재확정일자 : 2010. 08. 05

Ⅰ. 서 론

블록 기반의 움직임 추정 기법은 현재의 많은 비디오 부호화 표준안 등에서 널리 사용되어 오고 있다. 기존의 움직임 추정 기법 중에서 가장 간단한 블록 정합 움직임 추정 기법은 전역 탐색 기법인 FSA(Full Search Algorithm)가 있다. 이 기법은 탐색 창 내에서 모든 가능한 탐색 점 전부에 대해서 움직임 추정을 하여 최적의 MV(Motion Vector)를 구하고 있다. 그러나 이 FSA는 너무 많은 계산을 요구하므로, 현재까지 다양한 고속 움직임 추정 기법들이 연구되어 오고 있다[1-4].

지금까지 연구되어 오고 있는 고속 움직임 추정 기법들 중에서 움직임 탐색 범위를 줄임으로써 고속 움직임 추정을 구현하는 방법들도 많이 연구되어 오고 있는 한 분야이다. 탐색 범위의 크기는 움직임 추정에서 계산 양에 영향을 주는 중요한 요인 중에 하나이다. 작은 탐색 범위에서 움직임 추정을 한다면 계산 양은 줄일 수 있으나 움직임 추정 성능 저하의 요인이 된다. 따라서 움직임 추정 성능은 그대로 유지하면서 계산 양을 줄일 수 있는 적절한 탐색 범위가 필요로 하다.

기존의 움직임 탐색 범위 조절을 통한 고속 움직임 추정 기법은 DSWA(Dynamic Search Window Adjustment)[1], MAS(Motion Adaptive Search)[2], MWFA(Modified Window Follower Algorithm)[3]와 SDSR(Simple Dynamic Search Range)[4] 등이 있다. 이들 중에서 특히 MWFA와 SDSR은 모두 시간적 그리고 공간적 정보를 이용하여 탐색 범위 조절을 하는 성능이 우수한 고속 움직임 추정을 하는 방법이다. 그러나 이러한 기존의 방법은 시간적 정보로 참조 프레임에 있는 모든 블록들의 움직임 벡터들 중에서 최댓값을 이용하므로 움직임이 작은 블록들에는 세밀하게 적응하지 못하고 있다.

특히 MWFA는 바로 전에 부호화된 블록의 움직임 추정시 블록 정합 오차의 척도인 SAD(Sum of Absolute Difference) 값을 현재 블록의 탐색 범위 결정에 사용한다. 이는 움직임이 작아도 영상의 복잡도가 크거나 QP(Quantization Parameter)가 크면 SAD 또한 크게 나타날 수 있어 불필요하게 탐색 범위 크기가 커지게 되고, 또 그 반대가 있을 수도 있다. 그리고 SDSR은 MWFA에 비해서는 블록의 움직임 크기에 잘 적응하여 탐색 범위를 조절할 수 있는 방법이지만, 움직임이 복잡한 영상의 경우에는 보다 효과적으로 탐색 범위 조절을 하지 못하고 있다.

따라서 본 논문에서는 SDSR 기법에서 제안한 참조 프레임 전체의 움직임 정보 및 현재 프레임 안에서의 현재 블록 주변 4개 블록의 움직임 정보도 이용하지만, 보다 더욱 세밀한 조절을 위하여 현재 블록과 시간적으로 인접된 참조 프레임 안에서 같은 위치 블록과 주변 블록 8개의 움직임 정보를 이용하여 탐색 범위 조절을 하는 방법을 제안한다. 시간적으로 인접된 주변 블록들의 움직임 정보를 이용함으로써 움직임이 작은 많은 블록들은 불필요하게 큰 탐색 범위를 가지지 않을 수 있고, 현재 프레임에서 주변 블록보다는 움직임 큰 블록은 현재 블록의 움직임에 적절한 탐색 범위를 가질 수가 있다.

2장에서는 기존의 SDSR 기법을 기반으로 하지만 성능 향상을 위하여 시·공간적으로 인접한 주변 블록들의 움직임 정보를 이용하여 탐색 범위 조절을 하는 ASDSR(Advanced Simple Dynamic Search Range) 기법을 제안하고, 3장에서 시뮬레이션을 통해 제안한 방법의 성능을 검토한다. 끝으로 4장에서 결론을 맺는다.

Ⅱ. ASDSR

MV는 현재 블록과 참조 프레임에서 탐색 범위 내의 최적으로 정합되는 블록 사이의 변위를 나타내며, 이 MV의 각 성분의 절대 크기는 탐색 범위 크기보다는 작거나 같아야만 한다. 탐색 범위를 *SR*(Search Range)이라 하고, $MVmax$는 식 (1)과 같이 한 블록 MV의 x, y 성분의 절대 크기 중에서 최댓값이라 하면, $MVmax$와 *SR*과의 관계를 식 (2)와 같이 나타낼 수 있다[4].

$$MVmax = \max[|MVx|, |MVy|] \quad \cdots\cdots (1)$$

$$SR = MVmax + D \quad \cdots\cdots (2)$$

여기서 D는 SR과 $MVmax$와의 차이를 나타낸다. 이상적인 SR은 D = 0이 되도록 하여 $MVmax$와 같게 된다. 따라서 최적의 SR이 설정되면 움직임 추정할 때 불필요한 탐색을 줄임으로써 고속의 움직임 추정을 할 수 있다. 적응적인 탐색 범위 조절 기법은 각 블록마다 D를 최소화하여 SR이 $MVmax$에 근접하게 되도록 하여야 한다.

표 1은 기존의 고속 움직임 추정 기법들의 SR 조절 성능 평가를 여러 가지 실험 동영상에 대하여 실시해 본 결과이다. 실험 소프트웨어는 H.264/AVC 참조 소프트웨어인 JM13.0[5]에 기존의 기법을 구현하여 사용하였다. $MVmax$의 분산은 JM13.0에 있는 FFPS(Fast Full Pel Search) 기법을 사용하여 움직임 추정을 하였을 때 구해진 한 프레임 내에서 각 블록의 $MVmax$들의 분산을 구한 후에 프레임 당 평균을 나타내고 있으며, SR 조절 성능은 프레임 당 평균 D값과 FFPS기법 대비 평균 D값의 성능 향상 퍼센트로 나타내었다. 물론 각 기법의 부호화 성능인 PSNR값은 거의 동일하게 나타났다.

표 1. 고속 움직임 추정 기법들의 SR 조절 성능
Table 1. Performance of Conventional Methods

Testing sequence	Var. of $MVmax$	Avg. of $D(SR - MVmax)$				
		FFPS	MWFA		SDSR	
Akiyo QCIF	0.1	24.0	0.7	-97%	1.6	-93%
Mobile QCIF	1.0	23.7	4.8	-80%	1.5	-94%
Foreman QCIF	4.9	23.1	6.8	-71%	5.3	-77%
Coastguard CIF	5.1	23.4	12.2	-48%	5.8	-75%
Flower CIF	9.1	23.4	15.7	-33%	7.3	-69%
Stefan CIF	10.6	22.8	19.2	-16%	11.3	-50%
Bus CIF	21.8	22.0	21.3	-3%	14.4	-35%
Tempete CIF	103.9	22.7	20.5	-10%	12.0	-47%
Football CIF	250.6	19.6	19.5	-1%	17.4	-11%
Improvement				-40%		-61%

표 1을 살펴보면 기존의 SR 조절 기법들 중에서 SDSR 기법이 거의 모든 실험 동영상에 대해서 가장 뛰어난 성능을 나타내고 있다.

Chen 등이 제안한 이 SDSR 기법[4]은 참조 프레임의 모든 블록들의 $MVmax$ 중에서의 최댓값과 현재 블록과 공간적 상관성이 높은 주변 블록들의 $MVmax$에 따라 SR값을 결정하는 방법이다. 동영상에서 움직임의 정도는 아주 변화가 많으며, 한 프레임에서도 많은 부분에서 서로 움직임이 다를 수 있으므로 이 SDSR 기법은 시간적 그리고 공간적인 움직임의 상관성을 고려하여 SR의 조정을 하고자 하였다.

여기서 공간적 주변 블록들이란 그림 1에서 현재 프레임의 블록C1, 블록C2, 블록C3, 그리고 블록C4를 말한다.

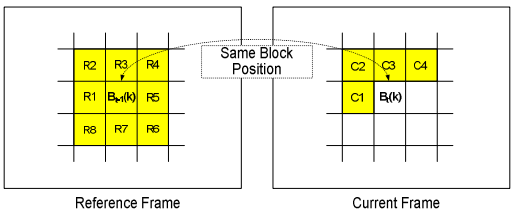

그림 1. 주변 블록들
Fig. 1 Neighboring blocks

기존의 SDSR 기법을 표 2에 나타내었다. 표 2에서 SR_FRAME_k는 인접된 프레임 간에는 움직임의 상관성이 있다는 것을 가정하고 이 가정을 이용하기 위하여 참조 프레임 내에서 모든 블록들을 대상으로 구한 $MVmax$의 최댓값이고, MV_MAX_i는 인접된 주변 블록들의 움직임과는 상관성이 아주 크다고 가정하고 이를 고려하기 위한 현재 프레임에서 주변 블록들, 즉 블록C1, 블록C2, 블록C3, 그리고 블록C4의 최대 $MVmax$ 값이다. 즉 SDSR 기법은 현재 블록의 움직임은 공간적으로 인접된 주변 블록들의 움직임과는 상관성이 아주 크다는 가정과 프레임들 사이의 시간적인 움직임의 상관성을 고려하여 제안된 기법이다.

그러나 SDSR 기법은 SR_FRAME_k의 크기에 비해 현재 블록의 움직임이 아주 작은 경우, 또는 공간적 주변 블록들의 움직임과는 상관성이 아주 작은 경우에 문제가 될 수 있다.

시간적 그리고 공간적인 움직임의 상관성이 작은 동영상은 한 프레임에서 MV 크기의 분산이 크게 나타나며 한 프레임 내에서 움직임이 복잡한 형태를 가지고 있다. 그래서 이러한 동영상에서는 기존의 SDSR기법의 SR 조절이 각 블록의 움직임의 특성에 따라 신속하게 일어나지 못하여 보다

효과적인 움직임 추정을 하지 못할 수 있다. 표 1에서 Football 동영상의 선수들 움직임과 같이 시간적 그리고 공간적인 상관성이 아주 작은 동영상의 경우에는 *MVmax*의 분산이 크게 나타나고, SDSR 기법이 *MVmax*의 분산이 클수록 성능 개선의 정도가 작아짐을 볼 수 있다.

표 2. 기존의 SDSR 알고리즘
Table 2. Simple Dynamic Search Range Algorithm

Step 1 : Determine the search range in frame level.
SR_FRAME_k = max $[MV_{xs}, MV_{ys}] + \gamma$, $\gamma \geq 0$
$t \in$ {all blocks in $(k-1)$th frame}

Step 2 : Adjust the search range in macroblock level.
$s \in$ {The left, above left, above, above right blocks of tth block}
If any of neighbor blocks is not available
MV_MAX_t = max [max $[MV_{xs}, MV_{ys}]$, SR_FRAME_k]
Else
MV_MAX_t = max $[MV_{xs}, MV_{ys}]$

Step 3 : Determine the final search range for tth block.
// Adjust *SR* in block level
If ($MV\_MAX_t \geq SR\_FRAME_k$)
$SR\_BLOCK_t = MV\_MAX_t + \beta$, $\beta \geq 0$
Else
$SR\_BLOCK_t = \alpha \cdot MV\_MAX_t + (1-\alpha) \cdot SR\_FRAME_k$,
$0 \leq \alpha \leq 1$
// *SR* constraint
If ($SR\_BLOCK_t \leq 1$)
$SR\_BLOCK_t = 1$
Else if ($SR\_BLOCK_t \geq$ max search range)
SR_BLOCK_t = max search range

그림 2는 Bus(CIF 150 프레임) 실험 동영상에 대해서 SDSR 기법의 프레임 당 *MVmax*의 분산과 *SR*과 *MVmax*와의 차이인 *D*의 평균값의 변화를 나타내고 있다. 이 그림에서도 *MVmax*의 분산이 크면 *D*의 평균값이 커지고, *MVmax*의 분산이 작으면 *D*의 평균값이 작아지는 현상을 볼 수 있다.

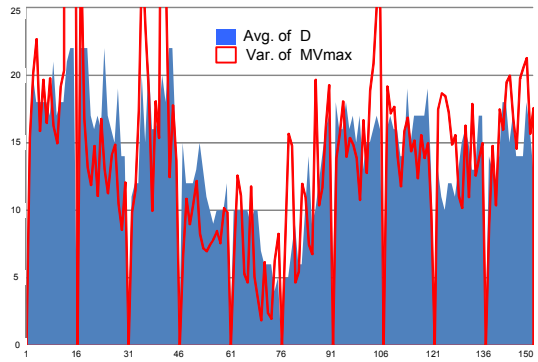

그림 2. Bus CIF 동영상에 대한 SDSR기법의 *D*의 평균과 *MVmax*의 분산
Fig. 2 Avg. of *D* & Var. of *MVmax* for Bus CIF sequence

따라서 본 논문에서는 현재 블록과 시간적으로 상관성이 높은 주변 블록들, 즉 참조 프레임 내에서 현재 블록과 같은 위치의 블록을 포함한 주변 블록들(그림 1의 참조 프레임 안에서 $B_{t-1}(k)$ 블록과 블록R1부터 블록R8까지를 말함)의 움직임 정보를 이용하여 시간적 그리고 공간적인 움직임의 상관성이 작은 동영상 즉 움직임이 복잡한 동영상에서도 보다 효과적으로 *SR*을 조절하는 ASDSR 기법을 제안한다.

본 논문에서 제안하는 ASDSR 기법을 표 3에 나타내었다. 기본적으로는 표 2에 나타낸 기존의 SDSR 기법과 유사하다. ASDSR 기법에서는 SDSR 기법에서 사용하는 참조 프레임 전체의 움직임 정보인 SR_FRAME_k(표 3의 *Step 2*) 및 현재 프레임 안에서의 현재 블록 주변 4개 블록의 움직임 정보인 MV_MAX_t(표 3의 *Step 3*)도 이용하고 있다.

그러나 움직임이 복잡한 동영상에서 더욱 세밀한 조절을 위하여 참조 프레임 내에서 현재 블록과 같은 위치의 블록을 포함한 주변 블록들 즉 시간적으로 인접된 주변 블록 9개 각각의 움직임 정보인 *MVmax*들로부터 $Ref\_MV\_MAX_t$(표 3의 *Step 1*)를 구한다. 시간적으로 인접된 주변 블록 9개는 그림 1의 참조 프레임 안에서 $B_{t-1}(k)$ 블록과 블록R1부터 블록R8까지를 말한다.

표 3. 제안하는 ASDSR 알고리즘
Table 3. Proposed Advanced SDSR Algorithm

Step 1 : Determine the search range in reference frame macroblock level.
$s \in$ {available blocks in the left, above left, above, above right, right, below right, below, below left blocks of tth block and tth block in $(k-1)$th frame}
$Ref\_MV\_MAX_t = \max [MV_{xs}, MV_{ys}] + \gamma$, $\gamma \geq 0$
// $Ref\_MV\_MAX_t$ constraint
If ($Ref\_MV\_MAX_t >$ max search range)
$Ref\_MV\_MAX_t$ = max search range

Step 2 : Determine the search range in reference frame level.
$SR\_FRAME_k = \max [Ref\_MV\_MAX_t]$
$t \in$ {all blocks in $(k-1)$th frame}

Step 3 : Determine the search range in current frame macroblock level.
$s \in$ {available blocks in the left, above left, above, above right blocks of tth block in kth frame}
$MV\_MAX_t = \max [MV_{xs}, MV_{ys}]$

Step 4 : Determine the final search range for tth block.
// Adjust SR in block level
If ($MV\_MAX_t \geq SR\_FRAME_k$)
$SR\_BLOCK_t = MV\_MAX_t + \beta$, $\beta \geq 0$
Else if ($MV\_MAX_t > Ref\_MV\_MAX_t$)
$SR\_BLOCK_t = \alpha \cdot MV\_MAX_t + (1-\alpha) \cdot SR\_FRAME_k$,
$0 \leq \alpha \leq 1$
Else
$SR\_BLOCK_t = Ref\_MV\_MAX_t$
// SR constraint
If ($SR\_BLOCK_t < 1$)
$SR\_BLOCK_t = 1$
Else if ($SR\_BLOCK_t >$ max search range)
SR_BLOCK_t = max search range

*Step 4*에서는 MV_MAX_t가 $Ref_MV_MAX_t$보다 큰 경우에는 기존의 SDSR기법과 동일하게 수행된다. 그러나 MV_MAX_t가 $Ref_MV_MAX_t$보다 크지 않은 경우에는 SR_BLOCK_t는 $Ref_MV_MAX_t$가 된다. 즉 현재 프레임에서 주변 블록의 움직임이 비교적 작은 경우에는 SR_FRAME_k의 영향을 전혀 받지 않도록 하여 불필요하게 큰 탐색 범위를 가지지 않을 수 있고, 현재 프레임에서 주변 블록보다는 움직임 큰 블록은 현재 블록의 움직임에 적절한 탐색 범위를 가질 수가 있다.

따라서 제안한 ASDSR기법은 움직임이 복잡한 동영상, 즉 시간적 그리고 공간적인 움직임의 상관성이 작고 MV 크기의 분산이 큰 동영상에서도 블록의 움직임 특성에 더욱 알맞게 SR 조절을 할 수 있도록 한 고속 움직임 추정 기법이다.

III. 시뮬레이션 결과 및 검토

본 논문에서 제안하는 시·공간적 주변 블록들의 움직임 정보를 이용하는 탐색 범위 조절을 통한 고속 움직임 추정 기법인 ASDSR 기법의 성능을 컴퓨터 시뮬레이션을 통하여 평가하였다. 실험 소프트웨어는 H.264/AVC 참조 소프트웨어인 JM13.0을 수정하여 구현하였다. 실험에서 제안된 기법의 성능을 평가하기 위하여 각 블록의 탐색 점들의 수를 관찰하였다. 그리고 부호화의 효율을 평가하기 위하여 양자화 계수를 상수로 하고 비트율 조절을 하지 않도록 한 후에 발생된 비트 수를 비교하였다. 그리고 사용된 탐색 범위인 SR 값이 적절하게 사용되었는지를 평가하기 위하여 SAD 값을 비교하였다. 본 실험에서 사용한 실험 조건을 표 4에 나타내었으며, 본 논문에서 제안하는 ASDSR 기법과 기존 기법들의 실험 결과를 표 5에 나타내었다. 그리고 실험에 사용된 동영상들의 종류는 표 6에 나타내었다.

표 4. 실험 조건
Table 4. Testing Conditions

Software: H.264/AVC JM13.0
Max. M.E. search range: ± 24 pixels
RDO : enabled
Number of reference frame: 1
Quantization parameter (QP): 36
GOP size: 15
Profile: baseline
Prediction structure: IPPP…
Fast ME(UMHexagonS): disable

표 5을 살펴보면 ASDR기법의 SR 조절 성능인 D값이 거의 모든 실험 동영상에 대해서 표 1에 나타낸 기존의 방법들 보다 훨씬 작음을 알 수 있다. 특히 기존의 SDSR 기법에 비해서 평균 19%의 성능 향상이 더 있음을 알 수 있다.

표 5. 제안한 ASDSR 기법의 성능 평가
Table 5. Performance of ASDSR algorithm

Test sequence	*D*		Search points for MB				Bits for P frame				SAD for P frame			
	ASDSR		FFPS	MWFA	SDSR	ASDSR	FFPS	MWFA	SDSR	ASDSR	FFPS	MWFA	SDSR	ASDSR
Akiyo	1.1	-95%	2401	7	18	10	769	776	768	767	1017	1020	1019	1019
Mobile	1.5	-94%	2401	121	19	19	13000	12983	13034	13007	3037	3039	3039	3040
Foreman	3.3	-86%	2401	262	174	85	2928	2951	2938	2942	1722	1729	1728	1731
Coastguard	3.4	-85%	2401	697	185	74	15148	15153	15179	15184	2064	2066	2070	2071
Flower	3.3	-86%	2401	1116	269	67	42506	42402	42360	42313	2449	2447	2447	2450
Stefan	5.5	-76%	2401	1747	655	190	31665	31704	31800	31869	2328	2330	2341	2348
Bus	6.9	-69%	2401	2266	1102	310	31197	31226	31260	32295	2526	2527	2527	2573
Tempete	3.5	-85%	2401	1989	751	92	29274	29284	29535	28748	2324	2324	2342	2323
Football	11.5	-41%	2401	2381	1989	1037	28821	28687	28870	28865	2414	2406	2418	2419
Improvement		-80%		-51%	-76%	-91%		0.13%	0.22%	0.29%		0.06%	0.26%	0.45%

표 6. 실험 동영상
Table 6. Test sequences

Name	Resolution	No. of frames	Motion activity (var. of MVmax)
Akiyo	QCIF	300	0.1
Mobile	QCIF	300	1.0
Foreman	QCIF	300	4.9
Coastguard	CIF	300	5.1
Flower	CIF	250	9.1
Stefan	CIF	90	10.6
Bus	CIF	150	21.8
Tempete	CIF	260	103.9
Football	CIF	260	250.6

그리고 움직임 복잡도의 척도인 *MVmax*의 분산이 클수록 더 많은 성능 향상이 있음을 알 수 있다.

실제 움직임 추정기법 속도의 척도인 블록 당 움직임 탐색 점의 수를 비교해 보면, 기존의 FFPS 기법에 비해 평균 91%의 감소를 보이고 있으며, SDSR 기법보다도 평균 15%의 감소를 더 보이고 있다. 그리고 탐색 점의 수에 대한 성능 평가에서도 움직임의 복잡도가 클수록 성능 개선이 더 있음을 알 수 있다.

움직임 추정 기법 속도의 성능은 평균 *D*의 값이 얼마나 작은지 그리고 탐색 점의 수가 얼마나 작은지를 보면 알 수 있다. 그러나 여기서 간과할 수 없는 조건은 움직임 추정 기법의 성능 즉 부호화시 발생되는 비트 수와 움직임 추정 오차에 영향을 주지 않아야 한다.

P프레임에 대한 평균 발생된 비트 수와 움직임 추정 오차의 척도로 많이 사용하는 평균 SAD값의 변화는 표 5에 나타나 있다. 기존의 FFPS 기법에 비해 발생된 비트 수는 평균 0.29%의 증가를 보이고 있으며, SAD는 평균 0.45%의 증가를 보이고 있다. SDSR 기법에 비해 발생된 비트 수는 평균 0.07% 더 증가하였으며, SAD는 평균 0.19% 더 증가하였다.

그러나 이 정도의 변화는 부호화 속도 개선의 정도에 비해 부호화기 전체의 성능에는 영향이 아주 미미하다고 할 수 있다. 그러나 움직임 복잡도가 아주 큰 동영상인 Tempete와 Football의 경우에는 기존의 SDSR 기법에 비해 비트 수와 SAD가 더 적은 상태에도 속도의 개선이 아주 많이 됨을 볼 수 있었다.

한편, 각 블록들에 대한 $Ref\_MV\_MAX_t$ 계산이 기존의 SDSR 기법 보다 더 필요하지만, 블록들의 움직임 추정 오차 계산에 비해 아주 작기 때문에 무시할 수 있다. 또한 표 3의 *Step 1*과 *Step 2*를 동시에 수행할 수 있어 계산양이 크게 늘어나지 않아 속도 저하를 가져오지 않을 것이다.

그림 3은 Bus(CIF 150 프레임) 실험 동영상에 대해서 ASDSR 기법의 프레임 당 *MVmax*의 분산과 *SR*과 *MVmax*와의 차이인 *D*의 평균값의

변화를 나타내고 있다. 그림 2에서는 기존의 SDSR 기법이 *MVmax*의 분산의 크기에 따라 *D*의 평균값이 비슷하게 변화하는 현상을 볼 수 있었지만, 그림 3에서는 ASDSR 기법이 *MVmax*의 분산의 크기와는 상관없이 *D*의 평균값이 거의 일정하게 나타내고 있다. 따라서 본 논문에서 제안하는 ASDSR 기법이 움직임의 복잡도가 큰 부분에서 훨씬 성능 개선이 있음을 알 수 있다.

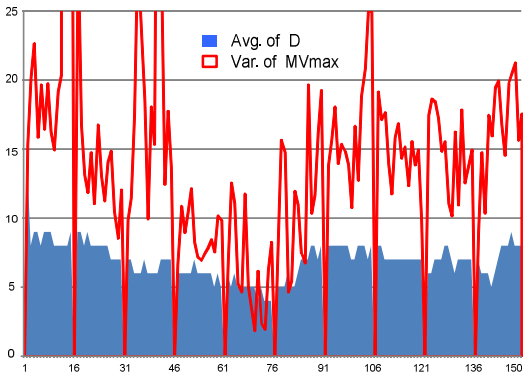

그림 3. Bus CIF 동영상에 대한 ASDSR기법의 *D*의 평균과 *MVmax*의 분산
Fig. 3 Avg. of *D* & Var. of *MVmax* for Bus CIF sequence

Ⅳ. 결 론

SDSR 기법을 비롯한 기존의 탐색 범위 조절 기법들은 동영상의 움직임이 복잡할수록 탐색 범위 조절 성능이 저하되는 단점을 보였다. 그러나 SDSR 기법을 기본으로 하고 시간적으로 인접된 주변 블록들의 움직임 정보를 추가로 이용한 ASDSR 기법은 움직임의 복잡도가 큰 동영상에서도 성능이 우수함을 나타내었다. 따라서 본 논문에서 제안한 ASDSR 기법은 움직임 추정 성능의 저하를 가져오지 않으면서 탐색 범위의 축소를 통하여 고속의 움직임 추정을 가능하게 하였다.

감사의 글
본 논문은 2007~2008년도 동양대학교 연구년 지원에 의해 이루어졌습니다.

참고문헌

[1] L. W. Lee, J. F. Wang, J. Y. Lee, and J. D. Shie, "Dynamic search-window adjustment and interlaced search for block-matching algorithm," IEEE Trans. Circuits Syst. Video Technol., vol. 3, no. 1, pp. 85-87, Jan. 1993.
[2] P. I. Hosur, "Motion adaptive search for fast motion estimation," IEEE Trans. Consumer Electron., vol. 49, no. 4, pp. 1330-1340, 2003.
[3] S. Saponara and L. Fanucci, "Data-adaptive motion estimation algorithm and VLSI architecture design for low-power video systems," IEE Proc. Comput. Digital Techniques, vol. 151, no. 1, pp. 51-59, 2004.
[4] Y. W. Chen, M. H. Hsiao, H. T. Chen, C. Y. liu, and S. Y. Lee, "Content-Aware Fast Motion Estimation Algorithm," J. Vis. Commu. Image R., vol. 19, pp. 256-269, Feb., 2008.
[5] H.264/AVC reference software, <http://iphome.hhi.de/suehring/tml/>.

저자 소개

이상학(Sang-hak Lee)
1984년 2월 : 경북대학교 전자공학과 (공학사)
1986년 2월 : 경북대학교 전자공학과 (공학석사)
2001년 8월 : 경북대학교 전자공학과 (공학박사)
현재 : 동양대학교 정보통신공학부 부교수
※ 주 관심분야 : HDTV, 영상신호처리