Probability Constrained Search Range Determination for Fast Motion Estimation

Hyun-Soo Kang, Si-Woong Lee, and Hamid Gholam Hosseini

In this paper, we propose new adaptive search range motion estimation methods where the search ranges are constrained by the probabilities of motion vector differences and a search point sampling technique is applied to the constrained search ranges. Our new methods are based on our previous work, in which the search ranges were analytically determined by the probabilities. Since the proposed adaptive search range motion estimation methods effectively restrict the search ranges instead of search point sampling patterns, they provide a very flexible and hardware-friendly approach in motion estimation. The proposed methods were evaluated and tested with JM16.2 of the H.264/AVC video coding standard. Experiment results exhibit that with negligible degradation in PSNR, the proposed methods considerably reduce the computational complexity in comparison with the conventional methods. In particular, the combined method provides performance similar to that of the hybrid unsymmetrical-cross multi-hexagon-grid search method and outstanding merits in hardware implementation.

Keywords: Motion estimation, H.264/AVC, adaptive search range.

Manuscript received Apr. 4, 2011; revised Nov. 21, 2011; accepted Dec. 6, 2011.

This work was conducted as a part of the research projects of "The Development of Security and Safety Systems based on Ubiquitous Technology for Shipping and Logistics" financially supported by the Ministry of Land, Transport and Maritime Affairs (MLTM) of Korea.

Hyun-Soo Kang (phone: +82 43 261 3488, hskang@cbnu.ac.kr) is with the College of Electrical and Computer Engineering, Chungbuk National University, Cheongju, Rep. of Korea.

Si-Woong Lee (swlee69@hanbat.ac.kr) is with the Division of Information Communication and Computer Engineering, Hanbat National University, Daejeon, Rep. of Korea.

Hamid Gholam Hosseini (hgholamh@aut.ac.nz) is with the Department of EE, Auckland University of Technology, Auckland, New Zealand.

http://dx.doi.org/10.4218/etrij.12.0111.0200

I. Introduction

Many video coding standards have widely adopted motion estimation (ME) to exploit temporal redundancy in video. However, that the ME process requires very intensive computation is a problem. This problem is more serious in video coding standards [1] that support variable block sizes, such as H.264/AVC. Among ME algorithms, full search algorithm (FSA) is most exhaustive in computation while it results in optimal motion vectors (MVs). To relieve the problem, many fast algorithms have been proposed, such as the three-step search [2], the four-step search [3], the diamond search [4], [5], the hexagon-based search [6]-[8], and fast FSAs [9], [10]. These algorithms reduce the computational complexity by means of search point sampling with their inherent search patterns. For the remainder of this paper, we call them search point sampling-based methods.

Even though the conventional algorithms alleviate the problem of the computational complexity, they often undergo quality degradation because of the local minimum problem. Moreover, they are not hardware-friendly because of their sequential behavior. That is, the searching process at the current step depends on the completion of the previous step. Particularly, some of these methods [7], [8] require the storing of the ME cost values of neighboring blocks, as well as the computing of the ME cost values for a few prediction positions, prior to actual searching operations, which causes further complexity in the hardware implementation. Such methods, which require conditional branch operations and serially repeated operations, should be avoided when considering hardware implementation. It is because hardware modules of these methods are designed to encompass the worst case scenarios, which require more computational power. For this

reason, the ME hardware modules have to read all pixels in the search area from frame memories, which requires considerable memory bandwidth.

In the meantime, adaptive search range (ASR) methods, which are also called dynamic search range (DSR) methods, are hardware-friendly, as they can be implemented by regular array structures [11]-[13]. Such regular structures facilitate parallel and pipeline operations by employing more processing elements. In addition, they have a merit in memory bandwidth required by pixel data reading operations. The major part of memory bandwidth is the number of clocks required for pixel data reading operations from external frame memories to internal memories of the ME module. For example, the reading operations occupy about 13.2% and 36.6% of the number of the clocks dedicated to the ME hardware module for the search ranges of 16 and 32, respectively. Here, it is assumed that the system specifications include: clock of 104 MHz, 64 bit data-bus, and inputs of 4CIF and 30 fps.[1)]

As shown in the example, the ASR methods may significantly improve the memory bandwidth problem as not all pixels in the search ranges are read. By saving the memory bandwidth, we have room for reducing the hardware complexity of the ME module. Furthermore, as the ASR methods provide rectangular-shaped search areas in which all pixels are valid, they can easily be combined with the search point sampling-based methods so that computational complexity may be additionally reduced.

The ASR methods introduced in [14]-[17] downsize the search ranges by using arithmetic computations of neighboring MVs. The search range of methods introduced in [16], [17] depends on the MVs rather than motion vector differences (MVDs). These methods employ thresholds to determine the search range, which makes them operate in a discrete manner. The search range for each block is determined such that it is proportional to the magnitude of the MVs of neighboring blocks of a current block. Our method is differentiated from these traditional methods in two aspects. Firstly, our method uses MVDs in determination of the search range, which is very effective for the image regions with consistent motions even for large motions. Secondly, it provides a mechanism that does not require any thresholds due to the increasing of MVDs. Lee and others [18] utilized motion estimation errors of neighboring blocks to restrict the search ranges. Chen and others [19] executed an MV estimation of a current block, followed by computation of the MVD between the estimated vector and a motion vector predictor (MVp). Then the search range was simply fixed to |MVD| + f where f is a positive value determined empirically.

While search point sampling-based methods, such as the three-step search algorithm, suffer from the local minimum problem, the ASR methods have the problem that search areas are incorrectly determined to cause performance degradation. In this paper, we aim to overcome the problem of the ASR methods. We made an effort to do this in our previous work [20], in which we presented an ASR determination method based on the probability model of MVDs. In this paper, we further improve our previous work and propose new methods based on it. The details are summarized as follows: 1) considering that search ranges are symmetric, the probability density function (PDF) of MVDs is modeled as the exponential distribution instead of the Laplace distribution in the previous work; 2) unlike the previous work, the search ranges of x and y directions are separately determined so that the performance may be improved, which results in a rectangular-shaped search area (that is, the search ranges of x and y directions are different from each other); 3) by additionally introducing a new set of samples for the estimation of the PDF's parameter, we propose the methods based on two sample sets to restrict the search ranges; and 4) to enhance the proposed methods in terms of the computational complexity, we newly introduce a combined method where the proposed methods based on the two sample sets are combined with the search point sampling-based methods.

II. Distribution of Motion Vector Differences and Search Range Determination

Given a search range SR as an input parameter of H.264/AVC, the MVD of a block must be in the range of $[-SR, +SR]$. As MVp corresponds to the center of a search area, the search area is given as MVp±SR. It may cause a waste of computing power for the blocks with small motions. Thus, if there is any evidence that the MVD is in the range of $[-k, +k]$, where $k<SR$, we can cut down the search area to MVp±k. For the evidence, we empirically investigate the distribution of the MVDs and propose the statistical model of it in this section. Based on the distribution, the effective search ranges are determined. For instance, if the variance of the distribution is small, we do not have to take a wide search range. Thus, the search ranges can be effectively managed by the variance. Alternatively, the search range can be controlled by the probability of an event related to an MVD. That is, we define the case in which the MVD is in the range of $[-k, +k]$, and we find the value of k such that the event occurs with more than a

1) 4CIF of 30 fps contains 47,520 macroblocks per second. Search ranges of 16 and 32 corresponds to (2∗16+1+15)∗(2∗16+1+15) pixels and (2∗32+1+15)∗(2∗32+1+15) pixels, where 15 is for inclusion of a macroblock in the search ranges, respectively. With 104 MHz clocks, about 2,188 clocks can be assigned to each macroblock, that is, 104M clocks/47,520 macroblocks. With 64 bit data-bus, one reading operation can read 8 pixels in a single clock. As a result, 288 clocks and 800 clocks are required to read all pixels of the search ranges of 16 and 32, respectively. These reach 13.2% and 36.6% of 2,188 clocks.

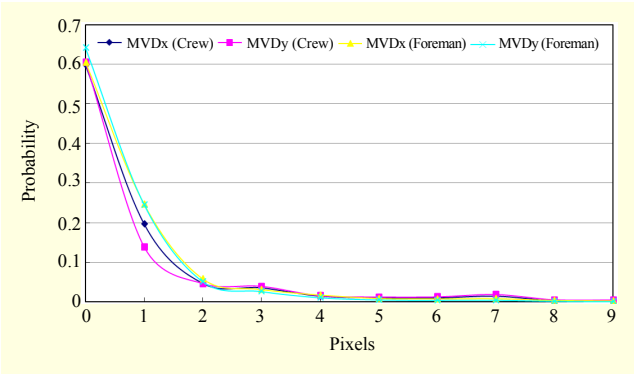

Fig. 1. Empirical distribution of absolute values of MVD components.

prefixed probability. Though the prefixed probability is fixed to a constant value for every block, the value of k may not be fixed because the parameter of the distribution can be different according to characteristics of the blocks.

In many cases, prediction error signals follow Laplace distributions or Gaussian distributions. By empirical results for MVDs, in this paper, we conclude that the MVDs follow Laplace distributions. Therefore, the absolute value of each component may abide by an exponential distribution. Figure 1 shows the distributions of the absolute values of x and y components of the MVDs resulting from the encoding of Foreman (CIF) 100 frames and Crew (706×576) 100 frames. Figure 1 reveals that exponential distributions can be good models to represent the MVD components. In addition, two components are very similar to each other in the distribution.

The exponential distribution of a continuous random variable (RV) Z is defined as

$$f_Z(z) = \alpha \cdot e^{-\alpha z} u(z), \tag{1}$$

where α is a positive constant, often called the rate parameter, and $u(\cdot)$ is the unit-step function.

However, the MVDs consist of discrete values, so each MVD must be considered a discrete RV. This says that the exponential distribution in (1) has to be modified such that the sum of probabilities over all $Z \in \mathbf{I}^+$, where $\mathbf{I}^+$ is the set of non-negative integers, is 1. As the sum in (1) is $\alpha(1-e^{-\alpha})^{-1}$ for $Z \in \mathbf{I}^+$, the exponential distribution of the non-negative integer valued MVDs should be

$$f_Z(z) = \beta \cdot e^{-\alpha z} u(z), \text{ where } \beta = (1-e^{-\alpha}). \tag{2}$$

In (2), it should be noted that the discrete random Z corresponds to the absolute value of the x or y component of the MVD vectors. Although the PDF is given by an integer pixel unit accuracy for convenience, it can be easily converted to a half pixel or a quarter pixel unit accuracy by scaling.

In the meantime, we assume that x and y components, which are denoted by MVDx and MVDy, are independent of each other. This assumption is reasonable because they are error signals given by the independent prediction process. When they are independent, the absolute values of them are also independent. Consequently, we have a joint PDF:

$$f_{XY}(x,y) = (1-e^{-\alpha_x})(1-e^{-\alpha_y}) \cdot e^{-(\alpha_x x + \alpha_y y)} u(x) u(y). \tag{3}$$

Having the PDF of MVDs, we now describe how to find the parameters, α_x and α_y. To estimate these parameters, we consider N samples, $\mathbf{s}_1$, $\mathbf{s}_2$, ... , $\mathbf{s}_N$, which are independent and identically distributed. Employing the maximum likelihood estimation technique to obtain an estimate $\hat{\boldsymbol{\alpha}}$ of $\boldsymbol{\alpha}$, we have the following formula:

$$\begin{aligned} \hat{\boldsymbol{\alpha}} &= \max_{\boldsymbol{\alpha}} l(\boldsymbol{\alpha} \mid \mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_N) \\ &= \max_{\boldsymbol{\alpha}} \prod_{i=1}^{N} (1-e^{-\alpha_x})(1-e^{-\alpha_y}) e^{-(\alpha_x x_i + \alpha_y y_i)}, \end{aligned} \tag{4}$$

where $\hat{\boldsymbol{\alpha}} = (\hat{\alpha}_x, \hat{\alpha}_y)$, $\boldsymbol{\alpha} = (\alpha_x, \alpha_y)$, $\mathbf{s}_i = (x_i, y_i)$, and $l(\cdot)$ is a likelihood function. Then,

$$\begin{aligned} \hat{\boldsymbol{\alpha}} &= \max_{\boldsymbol{\alpha}} \ln\left[l(\boldsymbol{\alpha} \mid \mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_N) \right] \\ &= \max_{\boldsymbol{\alpha}} \left[N \ln(1-e^{-\alpha_x}) + N \ln(1-e^{-\alpha_y}) - \sum_{i=1}^{N} \left(\alpha_x x_i + \alpha_y y_i \right) \right]. \end{aligned} \tag{5}$$

Differentiating (5) with respect to α_x and α_y, this leads to the following equations:

$$\hat{\alpha}_z = \begin{cases} \ln(1+1/\hat{\mu}_z), & \hat{\mu}_z \neq 0, \\ 0, & \hat{\mu}_z = 0, \end{cases} \tag{6}$$

where $\hat{\mu}_z = \frac{1}{N} \sum_{i=1}^{N} z_i$ and $z \in \{x, y\}$.

Equation (6) says that the parameter estimation requires logarithmic computation, which causes computational complexity. An approximation technique to simplify the computation will be described in the remainder in this section.

Once the distribution is estimated by the equations above, the probability that an MVD falls within a given range can be found. Consider an event $\{X>k\}$ where the x component of an MVD is not included in a given range $\pm k$. We can define the same event $\{Y>k\}$ for the y component. To satisfy each of $P\{X>k\} \leq \varepsilon_x$ and $P\{Y>k\} \leq \varepsilon_y$, where ε_x and ε_y are called missing probability for each component, we have to choose k such that:

$$k_z \geq -\frac{\ln(\varepsilon_z)}{\alpha_z} - 1 \equiv k_{z\min}, \text{ where } z \in \{x, y\}. \tag{7}$$

Substituting the estimated parameters above into these equations, the search ranges, $[k_{x\min}, k_{y\min}]$, can be obtained when given the missing probabilities.

Combining (6) and (7), the relation between $k_{z\min}$ and $\hat{\mu}_z$ is

$$k_{z\min} = -\frac{\ln(\varepsilon/2)}{\ln(1+1/\hat{\mu}_z)} - 1, \text{ where } z \in \{x, y\}. \tag{8}$$

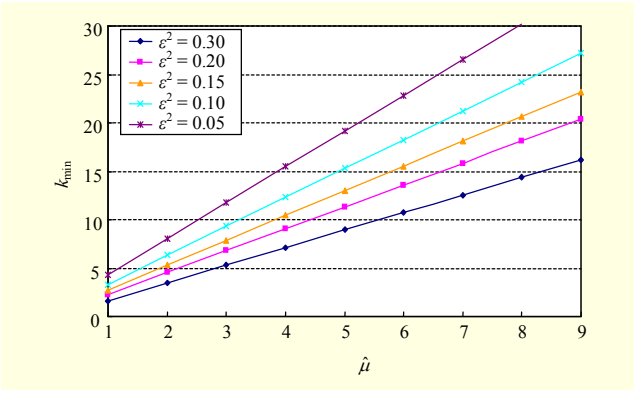

Fig. 2. Relation of search range $k_{\min}$ and absolute mean of samples $\hat{\mu}$ for different ε^2 values.

Table 1. Coefficients for first order polynomials.

Coefficients	ε^2				
	0.30	0.20	0.15	0.10	0.05
a	1.820	2.258	2.561	2.982	3.692
b	–0.206	–0.014	0.118	0.302	0.612

Figure 2 shows the relation of the search range $k_{z\min}$ to the absolute mean of samples $\hat{\mu}_z$ for different ε^2 values. For instance, if ε^2=0.1 and $\hat{\mu}_z$=3, then $k_{z\min}$=9.3. This explains that the optimal MV is in [MVp−9.3, MVp+9.3] with the probability of 90%. As shown in Fig. 2, (8) can be approximated by the first order polynomial function:

$$k_{z\min} = a\hat{\mu}_z + b. \tag{9}$$

The coefficients in (9) are given in Table 1, where the coefficient *a* plays a major role in determination of $k_{z\min}$. Therefore, the logarithmic computation of the search range $k_{z\min}$ based on (8) can be simplified using the approximation of (9). The search range $k_{z\min}$ can be simply obtained by adding the offset coefficient *b* to the multiplication of the coefficient *a* and the absolute mean of samples $\hat{\mu}_z$.

III. Proposed Method

We have described the PDF of MVD vectors so far. To estimate $\hat{\alpha}_x$ and $\hat{\alpha}_y$, *N* samples should be chosen for precise PDF estimation. There may be many options in choosing the samples. In our view, the method in [19] corresponds to the case where *N*=1. The estimated MVD in [19] may be sensitive to the single sample. To overcome this problem, we should adopt as many samples as possible even though all samples may not well represent the PDF. Taking into account memory requirement and empirical results, we carefully select two sets of four samples.

As the first set, we select:

$$U_1 \equiv \{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_4\} = \{MV_A\text{-}MVp, MV_B\text{-}MVp, MV_C\text{-}MVp, MV_{col}\text{-}MVp\}, \tag{10}$$

where MV_A, MV_B, MV_C, and MV_{col} denote the MVs of the neighboring blocks defined by H.264/AVC. As shown in Fig. 3, they are left block, the upper block, the upper-right block, and the collocated block in the previous frame, respectively. In the case of the image boundary where the block C is not available, the upper-left block D is used instead of the block C.

Strictly speaking, U_1 is not MVDs' set but the set of estimates of MVD. We can consider that MV_A, MV_B, MV_C, and MV_{col} are estimates of the motion vector of the current block X. That is, the four estimates are

$$\hat{MV}_X^{\,1} = MV_A, \quad \hat{MV}_X^{\,2} = MV_B, \quad \hat{MV}_X^{\,3} = MV_C, \quad \hat{MV}_X^{\,4} = MV_{col}. \tag{11}$$

With the estimates, we can write estimates of MVD:

$$\hat{MVD}_1 = \hat{MV}_X^{\,1} - MVp, \quad \hat{MVD}_2 = \hat{MV}_X^{\,2} - MVp, \quad \hat{MVD}_3 = \hat{MV}_X^{\,3} - MVp, \quad \hat{MVD}_4 = \hat{MV}_X^{\,4} - MVp. \tag{12}$$

After all, we can conclude that the elements of U_1 are the estimates of MVD of the current block. If the average value of the estimates are somewhat accurate, the search range given by (9) probably contains the optimal motion vector because the search range is determined to be larger than the average value of the estimates considering the value of the constants *a* and *b*. That is, $k_{z\min}$ resulting from each of the missing probabilities in Table 1 is always larger than the average value of the estimates. Thus U_1 can be a good set to determine the search range. As U_1 is empirically better in performance than another set U_2, which will be introduced next, we included U_1 set in our method. Though the elements of U_1 set do not accurately correspond to MVD, they may be considered as MVD and valuable in terms of experiment results.

Each element of U_1 can be considered as an estimated MVD if MV_A, MV_B, MV_C, and MV_{col} are assumed to be estimates of the current block (X).

The upper layer prediction [21] is used for U_1. For all sub-blocks except 16×16 blocks, one of MV_A, MV_B, and MV_C is replaced with the MV of the upper layer block. Here it should be noted that one of $\mathbf{s}_1$, $\mathbf{s}_2$, and $\mathbf{s}_3$ for each component is obviously zero because one of MV_A, MV_B, and MV_C is equal to MVp. For this reason, for 16×16 blocks, the denominator *N* in (6) is reduced to (*N*–1) to achieve unbiased estimation for $\hat{\mu}_x$ and $\hat{\mu}_y$, that is,

$$\hat{\mu}_x = \frac{1}{N-1}\sum_{i=1}^{N} x_i, \quad \hat{\mu}_y = \frac{1}{N-1}\sum_{i=1}^{N} y_i. \tag{13}$$

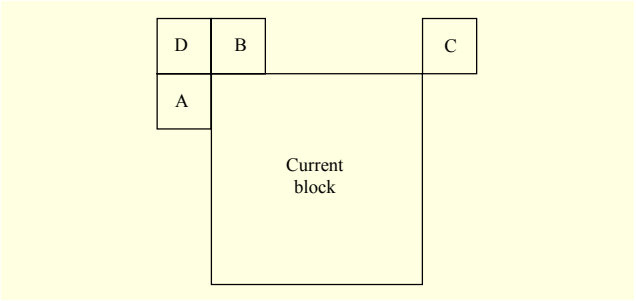

Fig. 3. Definition of neighboring blocks.

For sub-blocks other than 16×16 blocks, however, (6) is employed without modification since the MV of the upper layer block replaces the MV which causes the zero value for each component. Assuming the block mode of a current block is $\text{Mode}_{\text{curr}}$, the upper layer mode Mode_{up} is:

$$\text{Mode}_{\text{Up}} = \begin{cases} \text{unavailable}, & \text{if } \text{Mode}_{\text{curr}} = \text{Mode1}, \\ \text{Mode1}, & \text{if } \text{Mode}_{\text{curr}} = \text{Mode2, Mode3}, \\ \text{Mode2}, & \text{if } \text{Mode}_{\text{curr}} = \text{Mode4}, \end{cases} \quad (14)$$

where Mode1, Mode2, Mode3, and Mode4 are 16×16, 16×8, 8×16, and 8×8 block modes, respectively. The upper layer modes of the smaller sub-block modes follow the definition in [21].

As the second set, we selected:

$$U_2 \equiv \{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_4\} = \{\text{MVD}_A, \text{MVD}_B, \text{MVD}_C, \text{MVD}_{\text{col}}\}, \quad (15)$$

where MVD_A, MVD_B, MVD_C, and MVD_{col} are MVDs of A, B, C, and the collocated block, respectively. Similarly, if the block C is unavailable, it is replaced with the block D. This second set was adopted by our previous work [20] but we should note that it is applied to the exponential distribution model.

By intuition, it seems that U_2 set is not better than U_1 since MVDs are uncorrelated. However, this is not the estimation problem for the MVD of a current block, but for the distribution of it. For instance, consider the case where one estimates the distribution of a noise value at a time instant. In this case, we would choose the samples near the time instant for estimation, supposing a piece-wise stationary noise. In this manner, U_2 could also be a good candidate.

Meanwhile, there is an undesirable problem that occurs in estimation of $\hat{\alpha}_x$ and $\hat{\alpha}_y$. As a lot of MVD vectors are null, the absolute means of samples, $\hat{\mu}_x$ and $\hat{\mu}_y$, are often concluded to be zero, which causes $\hat{\alpha}_x = 0$ and/or $\hat{\alpha}_y = 0$ and hence $k_{x\min}$=0 and/or $k_{y\min}$=0. It means that the resultant search range corresponds to either a single search point at the center position of the search area or zero pixel wide search range including the center position. This is undesirable as it provides poor motion estimation. Hence, we should guarantee the minimal search range f for the case.

After all, the proposed method is summarized as below:

1) Set the missing probability ε^2.
2) Compute ($\hat{\mu}_x$, $\hat{\mu}_y$) and ($k_{x\min}$, $k_{y\min}$) using (6) and (9) if more than three out of $\mathbf{s}_1$, $\mathbf{s}_2$, $\mathbf{s}_3$, and $\mathbf{s}_4$ are available. Otherwise, set $k_{x\min}$ and $k_{y\min}$ to the original SR (SR_{org}) which is an input parameter to video encoder.
3) Obtain the final search range $k_x = \min(\max(k_{x\min}, f), SR_{\text{org}})$ and $k_y = \min(\max(k_{y\min}, f), SR_{\text{org}})$ which is to guarantee searching for at least $\pm f$, where f is a positive integer. By our experiments, f=2 or 3 is reasonable.
4) Go back to 2) for the next block.

The search point sampling-based algorithms are repeatedly applied to all enabled block modes in the serial manner. When m block modes are enabled, the computational complexity is m times of that when a single block mode is enabled. Therefore, due to the sequential behavior of these algorithms, it is not possible to implement them in hardware with the parallel structure where the motion estimation operations for all sub-blocks are simultaneously executed by forming all the sub-blocks from 4×4 primitive sub-blocks [13]. It is because the sampled search points of the sub-blocks may be different from each other and hence it is difficult for the upper-layer block modes to use the computational results of the lower-layer block modes. However, as the ASR methods can have the same search points for all the block modes, it is possible to use the results of the lower-layer block modes. Therefore, they can be implemented in the parallel structure, considering the fact that each block mode uses a different motion vector predictor in H.264/AVC. In this regard, the proposed algorithm can be slightly modified for the parallel implementation. If all block modes are enforced to have the same search area, it can be implemented by the parallel structure as shown in [13]. In this case, it is also possible to employ the fast FSA of JM [22] of H.264/AVC, where the costs of all block modes are efficiently computed by selectively adding the costs of 4×4 blocks. Conclusively, the proposed algorithm is hardware-friendly so that motion estimation for all block modes could be realized in the parallel manner with some modifications of search ranges.

IV. Experiment Results

For experiments, JM16.2 [22] is used. The motion estimation module in JM16.2 is replaced by the proposed methods for performance evaluation. The test sequences are as follows: Hall Monitor (352×288), Coastguard (352×288), Foreman (352×288), Stefan (352×288), City (704×576), Crew (704×576), and Soccer (704×576). We set the encoding parameters of JM16.2 as follows: 100 frames for each sequence, 30 fps, rate-distortion optimization off, IPPP picture

Table 2. BDPSNR and computational complexity (CPX) of conventional DSR method [16] and hybrid unsymmetrical-cross multi-hexagon-grid search (UMHexagonS) method [7] against FSA (JM16.2).

Images	SR	DSR [16]		UMHexagonS [7]	
		BDPSNR	CPX (%)	BDPSNR	CPX (%)
Hall Monitor	16	–0.039	10.31	–0.033	8.58
Coastguard	16	–0.008	12.26	0.013	2.27
Foreman	16	–0.031	15.96	–0.005	3.78
Stefan	16	–0.063	21.27	–0.065	3.46
City	16	–0.016	21.85	0.005	2.03
	32	–0.014	10.55	0.005	1.07
Soccer	16	–0.043	56.47	–0.011	3.15
	32	–0.020	30.48	0.002	1.51
Crew	16	–0.011	33.85	0.005	7.50
	32	0.002	23.37	0.010	4.56
Average		–0.0243	23.637	–0.0074	3.79

Table 3. BDPSNR and computational complexity (CPX) of proposed method using U_1 against FSA (JM16.2).

Images	SR	PM1 (ε^2=0.3)		PM1 (ε^2=0.2)		PM1 (ε^2=0.1)		PM1 (ε^2=0.05)	
		BDPSNR	CPX (%)	BDPSNR	CPX (%)	BDPSNR	CPX (%)	BDPSNR	CPX (%)
Hall Monitor	16	–0.049	7.00	–0.043	7.16	–0.041	7.75	–0.034	8.37
Coastguard	16	–0.013	6.97	–0.015	7.01	–0.014	7.20	–0.011	7.32
Foreman	16	–0.082	7.10	–0.059	7.40	–0.046	8.39	–0.032	9.24
Stefan	16	–0.123	7.17	–0.083	7.51	–0.033	8.38	–0.033	9.06
City	16	–0.023	4.64	–0.019	4.69	–0.009	4.89	–0.007	5.05
	32	–0.025	2.96	–0.024	2.97	–0.014	3.04	–0.017	3.11
Soccer	16	–0.138	5.35	–0.102	6.30	–0.060	8.21	–0.051	9.66
	32	–0.130	3.14	–0.096	3.48	–0.053	4.40	–0.040	5.38
Crew	16	–0.054	7.55	–0.042	11.21	–0.027	17.04	–0.025	21.13
	32	–0.047	4.23	–0.038	6.31	–0.024	10.13	–0.020	13.45
Average		–0.068	5.61	–0.052	6.40	–0.032	7.94	–0.027	9.18

structure, and one reference frame. The minimal search range f is set to 2.

Choosing QP=8, 18, 28, 38 where the four values follow the ones used in [8] and QP denotes quantization parameter, BDPSNR [23] is measured. For comparison, in Table 2, we exhibit the performance of two conventional methods: the hybrid unsymmetrical-cross multi-hexagon-grid search (UMHexagonS) method [7] and Xu and He's DSR method [16], which is an enhanced version of Hong and others' method [14], [15].

Tables 3 and 4 show the performance of the proposed method for two different sets (U_1 and U_2), called PM1 and PM2, respectively, against the FSA on JM16.2. The computational complexities (CPX) represent the ratio of the number of search points of the proposed method to the number of search points of the FSA. For instance, the CPX of 10% means that the number of search points used by the proposed method is 10% of the number of search points adopted by FSA. For both PM1 and PM2, PSNR and complexity increase as the missing probability decreases. Accordingly, PSNR and complexity can be handled by the missing probability. In overall performance, PM1 is superior to PM2.

Tables 3 and 4 also show that the proposed methods, PM1 and PM2, outperform the conventional DSR method [16] for all sequences. The shortfall of the DSR method is that its CPX rapidly increases for image sequences with fast motions when

Table 4. BDPSNR and computational complexity (CPX) of proposed method using U_2 against the FSA (JM16.2).

Images	SR	PM1 (ε^2=0.3)		PM1 (ε^2=0.2)		PM1 (ε^2=0.1)		PM1 (ε^2=0.05)	
		BDPSNR	CPX (%)	BDPSNR	CPX (%)	BDPSNR	CPX (%)	BDPSNR	CPX (%)
Hall Monitor	16	–0.039	7.11	–0.042	7.34	–0.035	8.11	–0.029	8.81
Coastguard	16	–0.010	7.00	–0.012	7.07	–0.009	7.33	–0.012	7.48
Foreman	16	–0.056	7.50	–0.037	8.11	–0.021	9.61	–0.016	10.82
Stefan	16	–0.082	7.55	–0.052	8.02	–0.020	9.04	–0.015	9.81
City	16	–0.015	4.73	–0.005	4.86	–0.001	5.30	0.000	5.72
	32	–0.018	2.98	–0.010	3.02	–0.007	3.19	–0.002	3.41
Soccer	16	–0.063	6.14	–0.037	7.29	–0.019	9.42	–0.010	11.00
	32	–0.051	3.41	–0.026	3.94	–0.014	5.14	–0.007	6.28
Crew	16	–0.028	9.99	–0.019	13.88	–0.012	19.74	–0.008	23.82
	32	–0.023	6.00	–0.014	8.57	–0.008	12.55	–0.009	15.81
Average		–0.039	6.24	–0.025	7.21	–0.015	8.94	–0.011	10.30

Table 5. BDPSNR and computational complexity (CPX) of PM1 combined with simple search point sampling method (PM1S) against FSA (JM16.2).

Images	SR	PM1S (ε^2=0.3)		PM1S (ε^2=0.2)		PM1S (ε^2=0.1)		PM1S (ε^2=0.05)	
		BDPSNR	CPX (%)	BDPSNR	CPX (%)	BDPSNR	CPX (%)	BDPSNR	CPX (%)
Hall Monitor	16	–0.037	2.82	–0.037	2.88	–0.036	3.06	–0.031	3.25
Coastguard	16	–0.013	2.79	–0.007	2.81	–0.009	2.86	–0.012	2.92
Foreman	16	–0.074	2.92	–0.068	3.09	–0.062	3.45	–0.066	3.82
Stefan	16	–0.062	2.94	–0.045	3.08	–0.022	3.34	–0.023	3.57
City	16	–0.014	2.20	–0.011	2.24	–0.007	2.33	–0.008	2.44
	32	–0.020	1.02	–0.019	1.03	–0.017	1.06	–0.011	1.11
Soccer	16	–0.053	2.56	–0.042	2.86	–0.029	3.40	–0.023	3.85
	32	–0.049	1.13	–0.030	1.26	–0.023	1.57	–0.020	1.88
Crew	16	–0.015	3.57	–0.011	4.59	–0.007	6.06	–0.005	7.22
	32	–0.017	1.81	–0.010	2.48	–0.005	3.48	–0.007	4.35
Average		–0.035	2.38	–0.028	2.63	–0.022	3.06	–0.021	3.44

compared with that of the proposed method. It yields low complexity for the Hall Monitor sequence, which has slow motions, but its complexity increases for faster motions, such as those in the Coastguard, Foreman, and Stefan sequences. This reveals that since the proposed methods use the MVDs, they effectively restrict the search range for the case where motions are fast but consistent. As the DSR method just uses MVs, it sets large search ranges when MVs of neighboring blocks are large, without consideration for the consistency of the MVs. As the center of motion search is at MVp, we don't have to set large search ranges for the image regions that have consistent motions even though the motions are large.

On the other hand, PM1 and PM2 underperform the UMHexagonS method. To improve their performance, they can be combined with the search point sampling-based methods [24]. The combining process is simple and straightforward as the search areas of the proposed methods are rectangular in shape and all pixels in the areas are available. However, we should consider that the main problem of the search point sampling-based methods is quality degradation caused by local minima. To overcome the problem, the proposed methods can be employed so that the original search area may be reasonably downsized to a small search area where the optimal MV may exist with a high probability. Then, the number of local minima decreases, and the danger of being trapped in local minima decreases, as well. To avoid the local

minimum problem even in the small area, we can consider a search point sampling method where the pixel points in the area are evenly sampled as search points rather than sparsely sampled.

As a result, we introduce a simple two-layer hierarchical searching method as the search point sampling-based method to be combined with the proposed methods. The first-layer searching step is performed at the positions with even number coordinates, $(2x, 2y)$ where $x, y \in \{\ldots, -2, -1, 0, 1, 2, \ldots\}$, followed by the second-layer searching step for nine points, $(x+a, y+b)$ where $x, y \in \{-1, 0, 1\}$ and (a, b) is the best position by the first searching step. At a glance, this simple method reduces the computational complexity to about a fourth of the original complexity when the second-layer searching process is ignored. The simple method has an attractive merit in hardware implementation. It can be realized by regular structures in accordance with a hardware-friendly approach of the proposed methods.

Table 5 shows the results when the simple method is performed for the search ranges determined by PM1. As shown in Table 5, combining PM1 with the simple method, which is called PM1S, gives significant reduction in the computational complexity with a negligible degradation in picture quality. Sometimes, it outperforms PM1 in image quality despite sampling the search points. It is probably because the search range is effectively extended by one pixel due to the second-layer searching step. The minimal search range adopted in the experiments is plus or minus two pixels wide, that is, f=2. The minimal search range is frequently taken since a lot of blocks have no motion. In this case, MVs can range up to plus or minus three pixels by the second-layer searching step.

We now compare PM1S with the UMHexagonS method. In CPX, PM1S is lower than the UMHexagonS method, on average. In measuring the complexity, the early termination process in the UMHexagonS method is not included in complexity. Note that the early termination process causes irregular termination of the ME process. In BDPSNR, PM1S is slightly lower than the UMHexagonS method; for instance, it is 0.014 dB lower at ε^2=0.05. However, the degradation in BDPSNR may be negligible. Though PM1S is slightly inferior in BDPSNR, it has an outstanding advantage in that it is much more hardware-friendly than the UMHexagonS method, as its algorithm is simple and regular. Specifically, as the UMHexagonS method contains the early termination process, it is hard to predict when the early termination is in effect. In addition, the UMHexagonS method has different search patterns for different searching steps and the starting point of a current searching step is dependent on the result of the previous searching step, which makes it impossible to achieve parallelization in hardware implementation. In contrast, PM1S is very regular, and there is no termination in the process of motion estimation. Accordingly, it can be easily implemented with a parallel structure, and its termination time is predictable. In addition, the UMHexagonS method requires more memory bandwidth for the reading operation of the pixels in the search area. That is, all the pixels in the search area have to be read by the motion estimation module from frame memory because searching patterns become highly flexible as searching steps proceed. According to the experiment results shown in Tables 3 and 4, the proposed method performs motion estimation for about 10% of all search points, on average. Roughly speaking, the proposed method requires that only 10% of the pixels in the original search area are read. This means that the proposed method can save 90% of the memory bandwidth. Conclusively, when taking overall aspects into account, PM1S is competitive enough with the UMHexagonS method.

In the complexity measure of the proposed method, we do not include the complexity required for the search range computation of (9) as the complexity is insignificant. Once the constants a and b in (9) are fixed for a given missing probability, we do not have to compute them whenever motion estimation is performed. That is, they can be saved in the memory in advance. Furthermore, in the computation of the average value of four samples, we do not require a division operation by four when the original value of the constant a is divided by four beforehand, so that the division in the average value computation is contained in a. That is, (9) can be rewritten as

$$\begin{aligned} k_{z\min} &= \frac{a}{4}(z_1 + z_2 + z_3 + z_4) + b \\ &= a'(z_1 + z_2 + z_3 + z_4) + b, \text{ where } a' = a/4. \end{aligned} \quad (18)$$

As preprocessing for motion estimation, four additions and one multiplication are required for each component. They are not significant, compared to SAD computations in motion estimation.

Meanwhile, a question about how to determine ε^2 still remains. The value of ε^2 should be chosen according to the characteristics of the input images since the same ε^2 yields different performances for different input images. As a solution, we can fix ε^2 for the next frame by observing PSNR and complexity resulting from encoding a frame. In determination of ε^2, additionally, we should take the quantization parameter into account. Figure 4 shows the complexities according to QPs. The complexities are inversely proportional to QP values while the slopes are dependent on ε^2 and input images. A small value of ε^2 also induces low variation of the complexities. Consequently, we should pay more attention to the cases in which there is a small value of ε^2 and a small value of QP.

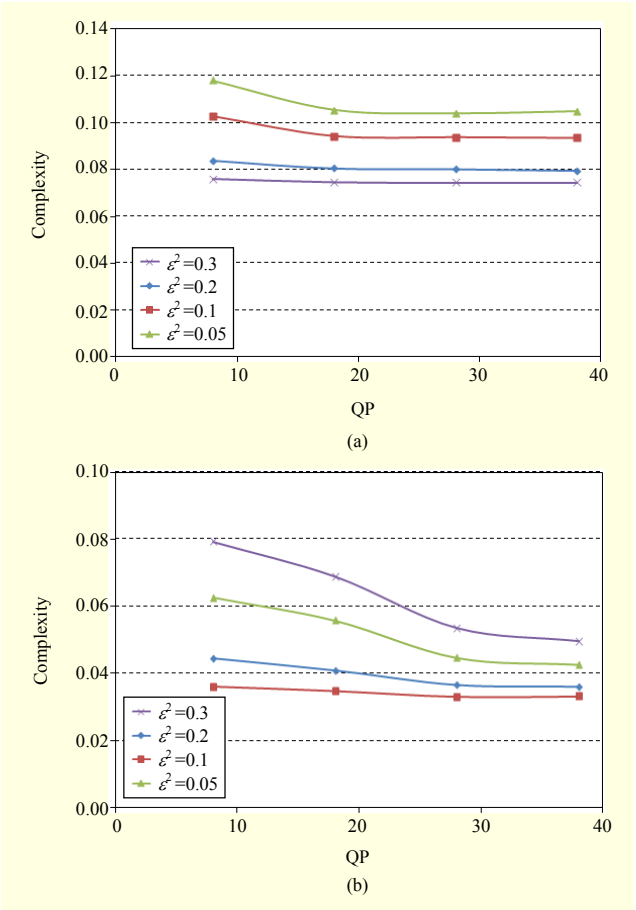

Fig. 4. Complexity versus QP: (a) Foreman CIF (SR=16) and (b) Soccer 4CIF (SR=32).

Since the complexity of motion estimation in our method is dependent upon the value of ε^2, more investigation into the determination of ε^2 is required as further work.

V. Conclusion

We proposed new ASR methods wherein the search ranges are constrained by the probabilities of MVDs and a search point sampling technique is applied to the constrained search ranges. Our previous work in which the search ranges were analytically determined by the probabilities was improved and new methods based on it were proposed. The PDF of MVDs was modeled as the exponential distribution and then its parameter was estimated. To relieve the complexity to compute search ranges, we introduced a formula showing that the search ranges are linearly proportional to the absolute mean of the MVD samples. For the parameter estimation, two sets of samples U_1 and U_2 were introduced. The first set was taken by estimating the MVDs of the current block using motion information of neighboring blocks. The second set of samples was selected from the MVDs of neighboring blocks. In addition, we proposed a combined method with a simple search point sampling-based method. The simple method was also selected due to its hardware-friendly aspect. An evaluation for the two sets of the proposed method (PM1 and PM2) and for the combined method was performed with the test sequences. In particular, the combined method provided performance similar to that of the UMHexagonS method, having the outstanding advantage in hardware implementation, that is, parallelism, memory bandwidth, processing termination time predictability, and so on.

References

[1] ITU-T VCEG and ISO/IEC MPEG, "Advanced Video Coding for Generic Audiovisual Services," ITU-T Recommendation H.264 and ISO/IEC 14496-10 (MPEG-4 AVC), May 2003.

[2] J. Jain and A. Jain, "Displacement Measurement and Its Application in Interframe Image Coding," *IEEE Trans. Commun.*, vol. 29, no. 12, Dec. 1981, pp. 1799-1808.

[3] L.-M. Po and W.-C. Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 3, June 1996, pp. 313-317.

[4] S. Zhu and K.-K. Ma, "A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation," *IEEE Trans. Image Process.*, vol. 9, no. 2, Feb. 2000, pp. 287-290.

[5] C. Cheung and L. Po, "A Novel Cross-Diamond Search Algorithm for Fast Block Motion Estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 12, Dec. 2002, pp. 1168-1177.

[6] C. Zhu, X. Lin, and L.P. Chau, "Hexagon-Based Search Pattern for Fast Block Motion Estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 5, May 2002, pp. 349-355.

[7] Z. Chen, P. Zhou, and Y. He, "Fast Integer and Fractional Pel Motion Estimation for JVT," JVT-F017r, Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG, Dec. 2002.

[8] X. Yi et al., "Improved and Simplified Fast Motion Estimation for JM," JVT-P021, Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG, July 2005.

[9] W. Li and E. Salari, "Successive Elimination Algorithm for Motion Estimation," *IEEE Trans. Image Process.*, vol. 4, no. 1, Jan. 1995, pp. 105-107.

[10] M. Brunig and W. Niehsen, "Fast Full-Search Block Matching," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 2, Feb. 2001, pp. 241-247.

[11] T. Komarek and P. Pirsch, "Array Architectures for Block Matching Algorithms," *IEEE Trans. Circuits Syst.*, vol. 36, no. 10, Oct. 1989, pp. 1301-1308.

[12] Y. Jehng, L. Chen, and T. Chiueh, "An Efficient and Simple VLSI Tree Architecture for Motion Estimation Algorithms," *IEEE Trans. Signal Process.*, vol. 41, no. 2, Feb. 1993, pp. 889-900.

[13] C. Ou, C. Le, and W. Hwan, "An Efficient VLSI Architecture for H.264 Variable Block Size Motion Estimation," *IEEE Trans. Consum. Electron.*, vol. 51, no. 4, Nov. 2005, pp. 1291-1299.

[14] M.-C. Hong and H.H. Oh, "Range Decision for Motion Estimation of VCEG-N33," JVT-B022, Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG, Feb. 2002.

[15] M.-C. Hong, C.-W. Kim, and K.S. In, "Further Improvement of Motion Search Range," JVT-D117, Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG, July 2002.

[16] X. Xu and Y. He, "Modification of Dynamic Search Range for JVT," JVT-Q088, Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG, Oct. 2005.

[17] T. Song et al., "Adaptive Search Range Motion Estimation Algorithm for H.264/AVC," *Proc. Int. Symp. Circuits Syst.*, 2007, pp. 3956-3959.

[18] S.W. Lee, S.M. Park, and H.S. Kang, "Fast Motion Estimation with Adaptive Search Range Adjustment," *Optical Engineering*, vol. 46, no. 4, April 2007, pp. 040504-1-040504-3.

[19] Z. Chen et al., "A Macroblock Level Adaptive Search Range Algorithm for Variable Block Size Motion Estimation in H.264/AVC," *Proc. Int. Symp. Intell. Signal Process. Comm. Sys.*, 2007, pp. 598-601.

[20] H.S. Kang and J.H. Park, "Fast Motion Estimation Based on Search Range Adjustment Using Neighboring MVDs," *Lecture Note in Computer Science*, vol. 6455, 2010, pp. 239-248.

[21] K.-P. Lim, G. Sullivan, and T. Wiegand, "Text Description of Joint Model Reference Encoding Methods and Decoding Concealment Methods," JVT-N046, Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG, Jan. 2005.

[22] H.264/AVC Reference Software JM16.2. http://iphome.hhi.de/suehring/tml/download/

[23] K. Andersson, R Sjöberg, and A. Norkin, "Reliability Measure for BD Measurements," ITU-T SG16 Q.6 Document, VCEG-AL22, July 2009.

[24] S. Lee, S. Park, and J. Park, "270 MHz Full HD H.264/AVC High Profile Encoder with Shared Multibank Memory-based Fast Motion Estimation," *ETRI J.*, vol. 31, no. 6, Dec. 2009, pp. 784-794.

Hyun-Soo Kang received his BS degree in electronic engineering from Kyoungpook National University, Daegu, Rep. of Korea, in 1991 and his MS and PhD in electrical and electronics engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Rep. of Korea, in 1994 and 1999, respectively. From 1999 to 2005, he worked for Hynix Semiconductor Co., Ltd., ETRI, and Chungang University. He joined the College of Electrical and Computer Engineering of Chungbuk National University, Chungbuk, Rep. of Korea, in March 2005. His research interests include image compression and image processing.

Si-Woong Lee received his BS in electronic engineering from Kyoungpook National University, Daegu, Rep. of Korea, in 1991 and his MS and PhD in electrical and electronics engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Rep. of Korea, in 1993 and 1997, respectively. From 1995 to 2000, he was with Samsung Electronics Company, Ltd., where he was engaged in digital IC design as a senior engineer. He joined the Division of Information Communication and Computer Engineering of Hanbat National University, Daejeon, Rep. of Korea, in April 2000. His research interests include digital video compression and image processing algorithms.

Hamid Gholam Hosseini received his MSc in electrical and electronic engineering from Tehran University, Tehran, Iran, in 1985 and his PhD in biomedical engineering from Flinders University, Australia, in 2002. He is presently a senior research lecturer at the School of Engineering and a research group leader at the Engineering Research and Innovation Cluster (ERIC), Auckland University of Technology, Auckland, New Zealand. His research interests include biomedical signal and image processing, electronic noise, embedded and reconfigurable systems, and remote monitoring systems.