Improved Estimation of Transmission Distortion for Error-resilient Video Coding

Zhifeng Chen[1], Peshala Pahalawatta[2], Alexis Michael Tourapis[2], and Dapeng Wu[1,*]

[1]Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611
[2]Dolby Laboratories, 3601 W Alameda Ave, Burbank, CA 91505

***Abstract*—This paper presents an improved technique for estimating the end-to-end distortion, which includes both quantization error after encoding and random transmission error, after transmission in video communication systems. The proposed technique mainly differs from most existing techniques in that it takes into account filtering operations, e.g. interpolation in subpixel motion compensation, as introduced in advanced video codecs. The distortion estimation for pixels or subpixels under filtering operations requires the computation of the second moment of a weighted sum of random variables. In this paper, we prove a proposition for calculating the second moment of a weighted sum of correlated random variables without requiring knowledge of their probability distribution. Then, we apply the proposition to extend our previous error-resilient algorithm for prediction mode decision without significantly increasing complexity. Experimental results using an H.264/AVC codec show that our new algorithm provides an improvement in both rate-distortion performance and subjective quality over existing algorithms. Our algorithm can also be applied in the upcoming high efficiency video coding (HEVC) standard, where additional filtering techniques are under consideration.**

***Index Terms*—Error-resilient Rate Distortion Optimization (ERRDO), filtering, subpixel-level distortion estimation, fractional motion estimation, ERMPC, mode decision, wireless video**

I. Introduction

In a typical video encoder, two kinds of compression techniques are usually involved, lossless and lossy compression. Lossless compression can be applied by reducing the redundancy between spatio-temporal neighboring pixels, i.e. through intra/inter prediction, and through the design and use of better codeword representations for a given probability distribution, i.e. using entropy coding techniques. Lossy compression is used to further compress the video by reducing fidelity, e.g. through quantization. In traditional video coding or video compression designs, rate-distortion (R-D) theory [1], [2] is proposed to study the relationship between bit rate and video distortion induced by the lossy quantization operation. With a R-D function, the redundancy of video sequences can be maximally exploited and distortion can be controlled by an acceptable quantization scheme through a R-D optimization (RDO) technique. However, in error-prone channels, reducing the redundancy also reduces the resilience to random transmission errors which may incur an increase in the end-to-end distortion. On the other hand, increasing error resilience, e.g. through random intra refreshing, may reduce compression efficiency. Error-resilient RDO (ERRDO), or sometimes called loss-aware RDO, is one method that can be used to alleviate the adverse effects of both bandwidth limitations and random transmission errors.

The problem of minimizing the distortion given a bit rate constraint can be formulated as a Lagrangian optimization. Due to its discrete characteristics, however, the rate distortion function is not guaranteed to be convex [3]. Therefore, in this case, the traditional Lagrange multiplier solution for continuous convex function optimization is infeasible. The discrete version of Lagrangian optimization was first introduced in Ref. [4], and then applied in a source coding application in Ref. [3]. Due to its simplicity and effectiveness, this optimization method is widely used in traditional video coding applications [5], [6], [7]. In the case of ERRDO, however, the end-to-end distortion is caused by both quantization and packet transmission errors. While the quantization error is known to the encoder, the transmission error depends on the particular channel realization and is therefore unknown to the encoder. Instead, ERRDO can use an estimate of the expected end-to-end distortion through characterizing the channel behavior, e.g. packet loss probability (PLP), to help the RDO process.

However, predicting end-to-end distortion is particularly challenging due to 1) the spatio-temporal correlation in the input video sequence, that is, a packet error will degrade not only the video quality of the current frame but also that of the subsequent frames due to error propagation; 2) the nonlinearity of both the encoder and the decoder, which makes the instantaneous transmission distortion not equal to the sum of distortions caused by individual error events; and 3) varying PLP in time-varying channels, which makes the distortion process into a non-stationary random process.

Some pixel-level end-to-end distortion estimation algorithms have been proposed to assist mode decision as in Ref. [8], [9], [10], [11]. Stockhammer et al. [8], [9] proposed a distortion estimation algorithm by simulating K independent decoders at the encoder during the encoding process and then averaging their simulated distortion. This algorithm, which we will refer to as LLN, is based on the Law of Large Numbers. That is, the estimated result will asymptotically approach the

*Correspondence author: Prof. Dapeng Wu, wu@ece.ufl.edu, http://www.wu.ece.ufl.edu. This work was supported in part by the US National Science Foundation under grant ECCS-1002214.

expected distortion when K goes to infinity. In the H.264/AVC JM reference software [7], this method is adopted to estimate the end-to-end distortion for mode decision. However, in the LLN algorithm more simulated decoders lead to higher computational complexity and larger memory requirements. Also for the same video sequence and the same PLP, different encoders may have different estimated distortions due to the randomly produced error events at each encoder.

In Ref. [10], the Recursive Optimal Per-pixel Estimate (ROPE) algorithm is proposed to estimate the pixel-level end-to-end distortion by recursively calculating the first and second moments of the reconstructed pixel value. However, non-linear clipping that contributes to the transmission distortion is neglected [12]. In addition, enhancing ROPE to support pixel averaging operations, e.g., interpolation filtering, requires intensive approximation computation of cross-correlation terms. In Ref. [13], the authors extend ROPE by using the upper bound, obtained from the Cauchy-Schwarz approximation, to approximate the cross-correlation terms. However, such an approximation requires very high complexity. For example, for an N-tap filter interpolation, each subpixel requires N integer multiplications[1] for calculating the second moment terms; $N(N-1)/2$ floating-point multiplications and $N(N-1)/2$ square root operations for calculating the cross-correlation terms; and $N(N-1)/2+N-1$ additions and 1 shift for calculating the estimated distortion. On the other hand, the upper bound approximation is not accurate for practical video sequences since it assumes that the correlation coefficient is 1, for any two neighboring pixels. In Ref. [14], the authors propose several models to approximate the correlation coefficient of two pixels as functions, e.g., an exponentially decaying function, of their distance. However, due to the random behavior of individual pixel samples, the statistical model does not produce an accurate pixel-level distortion estimate. In addition, such approximations incur extra complexity compared to the Cauchy-Schwarz upper bound approximation, since they need additional $N(N-1)/2$ exponential operations and $N(N-1)/2$ floating-point multiplications for each subpixel. Therefore, the complexity incurred is prohibitively high for real-time video encoders. On the other hand, since both the Cauchy-Schwarz upper bound and the correlation coefficient model approximations require floating-point multiplications, additional round-off errors are unavoidable, which further reduce their estimation accuracy.

In Ref. [12], we proposed a divide-and-conquer method to quantify the effects of four individual terms on transmission distortion, i.e. 1) residual concealment error, 2) Motion Vector (MV) concealment error, 3) propagation error and clipping noise, and 4) correlations between any two of them. Based on our theoretical results, we proposed the RMPC algorithm in Ref. [11] for error-resilient rate-distortion optimized mode decision with pixel-level end-to-end distortion estimation. In state-of-the-art video codecs, such as H.264/AVC [15] and HEVC [16], fractional pixel motion compensation with interpolation filtering can have a substantial R-D performance gain. However, distortion estimation for pixels or subpixels under filtering operations requires the computation of the second moment of a weighted sum of random variables. In this paper, we first theoretically derive a proposition for calculating the second moment of a weighted sum of correlated random variables using a closed-form function of the second moments of those individual random variables. This proposition is general in estimating the distortion for any pixels after a filtering operation. Then, we apply the proposition to extend our previous RMPC algorithm to subpixel-level distortion estimation for prediction mode decision without significantly increasing complexity. This algorithm is referred to as ERMPC. The ERMPC algorithm only requires N integer multiplications, $N-1$ additions, and 1 shift to calculate the second moment for each subpixel. Experimental results show that, ERMPC achieves an average PSNR gain of 0.25dB over the existing RMPC algorithm for the 'mobile' sequence when PLP equals 2%; and ERMPC achieves an average PSNR gain of 1.34dB over the the LLN algorithm for the 'foreman' sequence when PLP equals 1%.

The rest of this paper is organized as follows. Section II presents the system description and the necessary preliminaries of the RMPC algorithm, and serves as a starting point for understanding the following sections. In Section III, we first derive the general proposition for the second moment of a weighted sum of correlated random variables, and then apply this proposition to design a low-complexity and high-accuracy algorithm for mode decision. Section IV shows the experimental results, which demonstrate the advantages of the ERMPC algorithm over existing algorithms for H.264/AVC mode decision in error prone environments. Section V concludes the paper.

II. System Description and Preliminaries

A. Structure of a Wireless Video Communication System

Fig. 1 shows the structure of a typical hybrid video coding system. Note that in this system, both the residual and the MV channels are application-layer channels. These channels can be separated for more general applications, e.g. for slice data partitioning under Unequal Error Protection (UEP). If the residual and MV packets are transmitted in the same channel with the same error protection, this becomes a special case where the two channels have the same characteristics and are fully correlated.

In Fig. 1, for any pixel $\mathbf{u}$ in the k-th frame, $f_{\mathbf{u}}^k$ is the input pixel value at the encoder; $\hat{f}_{\mathbf{u}}^k$ and $\widetilde{f}_{\mathbf{u}}^k$ are the reconstructed pixel values at the encoder and decoder respectively. Suppose the pixel $\mathbf{u}$ in the k-th frame uses the $k-j$-th frame as a reference, where $j \in \{1,2,...,J\}$ and J is the number of reference frames. If motion estimation is performed before mode decision as in the JM16.0 reference software, the best MV value $\mathbf{mv}_{\mathbf{u}}^k$ for each prediction mode and the corresponding residual value $e_{\mathbf{u}}^k = f_{\mathbf{u}}^k - \hat{f}_{\mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k}^{k-j}$ for each prediction mode are known. As a result, after the transform, quantization, dequantization, and inverse transform processes at the encoder, the reconstructed residual $\hat{e}_{\mathbf{u}}^k$ and the reconstructed pixel value

[1]One common method to simplify the multiplication of an integer variable and a fractional constant is as follows: first scale up the fractional constant by a certain factor; round it off to an integer; perform integer multiplication; finally scale down the product.

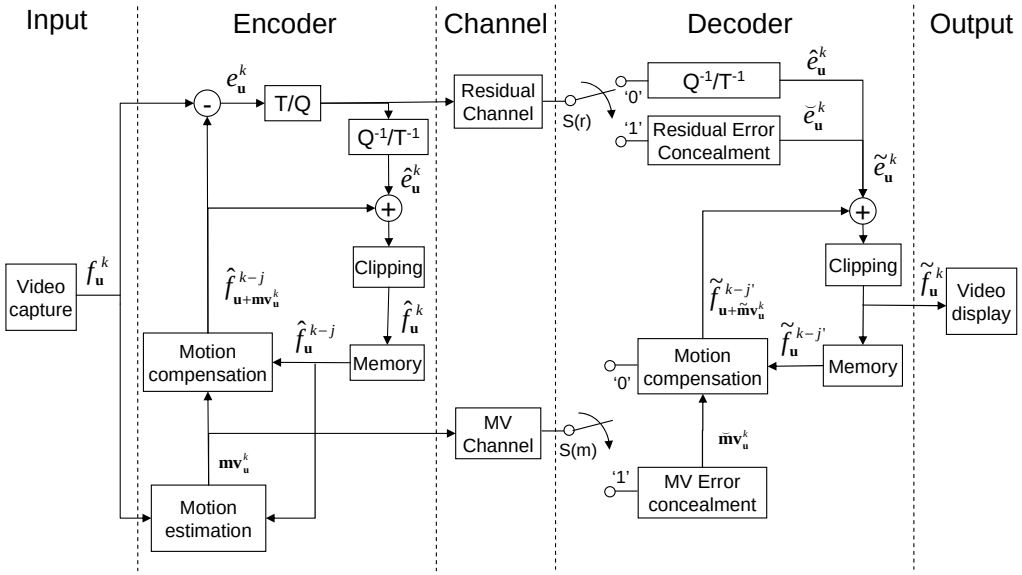


Fig. 1. System structure, where T, Q, Q^{-1}, and T^{-1} denote transform, quantization, inverse quantization, and inverse transform, respectively.

$\hat{f}^k_{\mathbf{u}} = \Gamma(\hat{f}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}} + \hat{e}^k_{\mathbf{u}})$ are known for each prediction mode. $\Gamma(\cdot)$ is the clipping function and is defined by

$$\Gamma(x) = \begin{cases} \gamma_L, & x < \gamma_L \\ x, & \gamma_L \le x \le \gamma_H \\ \gamma_H, & x > \gamma_H, \end{cases} \tag{1}$$

where γ_L and γ_H are low threshold and high threshold of pixel values. In this paper, we let $\gamma_L = 0$ and $\gamma_H = 255$.

Let us define $f^k_{\mathbf{u}} - \hat{f}^k_{\mathbf{u}}$ as quantization error and

$$\widetilde{\zeta}^k_{\mathbf{u}} \triangleq \hat{f}^k_{\mathbf{u}} - \widetilde{f}^k_{\mathbf{u}} \tag{2}$$

as transmission error. While the quantization error is known to the encoder after the encoding process, the transmission error is caused from any combinations of three error events, i.e. residual packet, MV packet, and propagated errors. In Fig. 1, the residual used for reconstruction in the decoder, i.e. $\widetilde{e}^k_{\mathbf{u}}$, may be either the quantized residual $\hat{e}^k_{\mathbf{u}}$ or the concealed residual $\check{e}^k_{\mathbf{u}}$ depending on the error status of residual packet $S(r)$; The MV used for motion compensation in the decoder, i.e. $\widetilde{\mathbf{mv}}^k_{\mathbf{u}}$, may be either the true MV $\mathbf{mv}^k_{\mathbf{u}}$ or the concealed MV $\check{\mathbf{mv}}^k_{\mathbf{u}}$ depending on the error status of MV packet $S(m)$; as a result, the reconstructed pixel value in the decoder is $\widetilde{f}^k_{\mathbf{u}} = \Gamma(\widetilde{f}^{k-j'}_{\mathbf{u}+\widetilde{\mathbf{mv}}^k_{\mathbf{u}}} + \widetilde{e}^k_{\mathbf{u}})$ where $\widetilde{f}^{k-j'}_{\mathbf{u}+\widetilde{\mathbf{mv}}^k_{\mathbf{u}}}$ is the reference pixel value for reconstruction at the decoder, which recursively depends on all error events in the reference trajectory till the intra coded pixel.

Depending on whether the MV is correctly received or not, the propagated error can be calculated, according to (2), by

$$\widetilde{\zeta}^{k-j'}_{\mathbf{u}+\widetilde{\mathbf{mv}}^k_{\mathbf{u}}} = \begin{cases} \widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}} = \hat{f}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}} - \widetilde{f}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}}, & \text{S(m)=0} \\ \widetilde{\zeta}^{k-j'}_{\mathbf{u}+\check{\mathbf{mv}}^k_{\mathbf{u}}} = \hat{f}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}} - \widetilde{f}^{k-j'}_{\mathbf{u}+\check{\mathbf{mv}}^k_{\mathbf{u}}}, & \text{S(m)=1}, \end{cases} \tag{3}$$

where the concealed MV may point to a different reference frame j' from the true reference frame j.

Clipping noise is defined as $\hat{\Delta}^k_{\mathbf{u}} \triangleq (\hat{f}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}} + \hat{e}^k_{\mathbf{u}}) - \Gamma(\hat{f}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}} + \hat{e}^k_{\mathbf{u}})$ at the encoder and as $\widetilde{\Delta}^k_{\mathbf{u}} \triangleq (\widetilde{f}^{k-j'}_{\mathbf{u}+\widetilde{\mathbf{mv}}^k_{\mathbf{u}}} + \widetilde{e}^k_{\mathbf{u}}) - \Gamma(\widetilde{f}^{k-j'}_{\mathbf{u}+\widetilde{\mathbf{mv}}^k_{\mathbf{u}}} + \widetilde{e}^k_{\mathbf{u}})$ at the decoder. That is, the clipping noise at the decoder is a function of the residual concealment, MV concealment, and propagated errors. Denote $\{\bar{r}, \bar{m}\}$ as the error event that both residual and MV are correctly received at the decoder for pixel $\mathbf{u}^k$; the clipping noise under this event will be $\widetilde{\Delta}^k_{\mathbf{u}}\{\bar{r}, \bar{m}\} = (\widetilde{f}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}} + \hat{e}^k_{\mathbf{u}}) - \Gamma(\widetilde{f}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}} + \hat{e}^k_{\mathbf{u}})$.

Some important notations used in this paper are listed in Table I. Throughout this paper, we add $\hat{}$ onto the reconstructed variables at the encoder; add $\check{}$ onto the concealed variables at the decoder; and add $\widetilde{}$ onto the variables, which are subject to the random channel error at the decoder.

B. Preliminaries about the RMPC algorithm

In Ref. [12], we take a divide-and-conquer approach to derive the second moment of $\widetilde{\zeta}^k_{\mathbf{u}}$, i.e. $E[(\widetilde{\zeta}^k_{\mathbf{u}})^2]$. We first divide the transmission reconstructed error into four components: three random errors (residual concealment, MV concealment, and propagated errors) due to their different physical causes, and clipping noise, which is a non-linear function of these three random errors. This error decomposition allows us to quantify the effects of below four terms on transmission distortion: (1) residual concealment error (R), (2) MV concealment error (M), (3) propagated error plus clipping noise (P), and (4) correlations between any two of the error sources (C). Based on this decomposition, we developed a practical algorithm, called RMPC algorithm, to estimate the pixel-level end-to-end distortion (PEED) for mode decision in Ref. [11].

In Ref. [11], the end-to-end distortion function for each pixel $\mathbf{u}$ in the k-th frame is defined as $D^k_{\mathbf{u},ETE} \triangleq E[(f^k_{\mathbf{u}} - \widetilde{f}^k_{\mathbf{u}})^2]$, and can be derived by

$$\begin{aligned} D^k_{\mathbf{u},ETE} &= E[(f^k_{\mathbf{u}} - \widetilde{f}^k_{\mathbf{u}})^2] = E[(f^k_{\mathbf{u}} - \hat{f}^k_{\mathbf{u}} + \widetilde{\zeta}^k_{\mathbf{u}})^2] \\ &= (f^k_{\mathbf{u}} - \hat{f}^k_{\mathbf{u}})^2 + E[(\widetilde{\zeta}^k_{\mathbf{u}})^2] + 2(f^k_{\mathbf{u}} - \hat{f}^k_{\mathbf{u}}) \cdot E[\widetilde{\zeta}^k_{\mathbf{u}}]. \end{aligned} \tag{4}$$

Define $\varepsilon^k_{\mathbf{u}} \triangleq \hat{e}^k_{\mathbf{u}} - \check{e}^k_{\mathbf{u}}$ as the residual concealment error when the residual packet is lost; define $\xi^k_{\mathbf{u}} \triangleq \hat{f}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}} - \hat{f}^{k-j'}_{\mathbf{u}+\check{\mathbf{mv}}^k_{\mathbf{u}}}$ as the MV concealment error when the MV packet is lost; and denote $P^k_{\mathbf{u}}$ as the PLP for pixel $\mathbf{u}$ in the k-th frame. Under the assumptions that $S(r)$ is independent of $e^k_{\mathbf{u}}$ and

TABLE I
NOTATIONS

$f_{\mathbf{u}}^k$	: Value of the pixel with position $\mathbf{u}$ in the k-th frame, i.e. pixel $\mathbf{u}^k$.
$e_{\mathbf{u}}^k$	: Residual value of the pixel $\mathbf{u}^k$.
$\mathbf{mv}_{\mathbf{u}}^k$	: MV of the pixel $\mathbf{u}^k$.
$\Delta_{\mathbf{u}}^k$	: Clipping noise of the pixel $\mathbf{u}^k$.
$\Delta_{\mathbf{u}}^k\{\bar{r},\bar{m}\}$	: Clipping noise of the pixel $\mathbf{u}^k$ under the condition that both residual and MV are correctly received at the decoder.
$\zeta_{\mathbf{u}}^k$	: Transmission error of the pixel $\mathbf{u}^k$, defined in (2).
$\varepsilon_{\mathbf{u}}^k$	: Residual concealment error of the pixel $\mathbf{u}^k$.
$\xi_{\mathbf{u}}^k$	: MV concealment error of the pixel $\mathbf{u}^k$.
$\widetilde{\zeta}_{\mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k}^{k-j}$	: 1) propagated error of the pixel $\mathbf{u}^k$ whose reference pixel is pointed to the $k-j$-th frame by the true MV $\mathbf{mv}_{\mathbf{u}}^k$;
	: 2) transmission error of the pixel with position $\mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k$ in the $k-j$-th frame according to (2).
$\widetilde{\zeta}_{\mathbf{u}+\check{\mathbf{mv}}_{\mathbf{u}}^k}^{k-j'}$	: 1) propagated error of the pixel $\mathbf{u}^k$ whose reference pixel is pointed to the $k-j'$-th frame by the concealed MV $\check{\mathbf{mv}}_{\mathbf{u}}^k$;
	: 2) transmission error of the pixel with position $\mathbf{u}+\check{\mathbf{mv}}_{\mathbf{u}}^k$ in the $k-j'$-th frame according to (2).

$S(m)$ is independent of $\xi_{\mathbf{u}}^k$, i.e. the packet transmission error is independent from the values of residual and MV, we proved in Refs. [12], [11] that without slice data partitioning, $E[\widetilde{\zeta}_{\mathbf{u}}^k]$ and $E[(\widetilde{\zeta}_{\mathbf{u}}^k)^2]$ can be calculated by (5) and (6) [2],

$$\begin{aligned} E[\widetilde{\zeta}_{\mathbf{u}}^k] &= P_{\mathbf{u}}^k \cdot (\varepsilon_{\mathbf{u}}^k + \xi_{\mathbf{u}}^k + E[\widetilde{\zeta}_{\mathbf{u}+\check{\mathbf{mv}}_{\mathbf{u}}^k}^{k-j'}]) \\ &\quad + (1-P_{\mathbf{u}}^k) \cdot E[\widetilde{\zeta}_{\mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k}^{k-j} + \widetilde{\Delta}_{\mathbf{u}}^k\{\bar{r},\bar{m}\}], \end{aligned} \tag{5}$$

$$\begin{aligned} E[(\widetilde{\zeta}_{\mathbf{u}}^k)^2] &= P_{\mathbf{u}}^k \cdot ((\varepsilon_{\mathbf{u}}^k + \xi_{\mathbf{u}}^k)^2 + 2(\varepsilon_{\mathbf{u}}^k + \xi_{\mathbf{u}}^k) \cdot E[\widetilde{\zeta}_{\mathbf{u}+\check{\mathbf{mv}}_{\mathbf{u}}^k}^{k-j'}] \\ &\quad + E[(\widetilde{\zeta}_{\mathbf{u}+\check{\mathbf{mv}}_{\mathbf{u}}^k}^{k-j'})^2]) + (1-P_{\mathbf{u}}^k) \cdot E[(\widetilde{\zeta}_{\mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k}^{k-j} + \widetilde{\Delta}_{\mathbf{u}}^k\{\bar{r},\bar{m}\})^2], \end{aligned} \tag{6}$$

where $E[\widetilde{\zeta}_{\mathbf{u}+\check{\mathbf{mv}}_{\mathbf{u}}^k}^{k-j'}]$ and $E[(\widetilde{\zeta}_{\mathbf{u}+\check{\mathbf{mv}}_{\mathbf{u}}^k}^{k-j'})^2]$ in the $k-j'$-th frame have been calculated by (5) and (6) and stored in memory during encoding of the $k-j'$-th frame; $E[\widetilde{\zeta}_{\mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k}^{k-j}+\widetilde{\Delta}_{\mathbf{u}}^k\{\bar{r},\bar{m}\}]$ and $E[(\widetilde{\zeta}_{\mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k}^{k-j} + \widetilde{\Delta}_{\mathbf{u}}^k\{\bar{r},\bar{m}\})^2]$ can be calculated by (7) and (8), where $E[\widetilde{\zeta}_{\mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k}^{k-j}]$ and $E[(\widetilde{\zeta}_{\mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k}^{k-j})^2]$ have been calculated and stored in memory during encoding of the $k-j$-th frame. Note that in this paper, for simplicity, we use $E(\cdot)$ to also represent the estimate of $E(\cdot)$, i.e. $\hat{E}(\cdot)$, in Ref. [11].

Existing pixel-level algorithms, e.g., the RMPC algorithm, are based on the integer pixel MV assumption to derive an estimate of $D_{\mathbf{u},ETE}^k$. In order words, in (5), (6), (7) and (8), $\mathbf{mv}_{\mathbf{u}}^k$ is with integer-pixel accuracy. Therefore, their application in state-of-the-art encoders is limited due to the possible use of fractional motion compensation. For subpixel motion compensation, $E[\widetilde{\zeta}_{\mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k}^{k-j}]$ and $E[(\widetilde{\zeta}_{\mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k}^{k-j})^2]$ need to be estimated based on the interpolated pixel values, i.e. a weighted sum of several neighboring pixels. That is, $E[(\widetilde{\zeta}_{\mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k}^{k-j})^2]$ requires the computation of the second moment of a weighted sum of correlated random variables and therefore the computation of several cross-correlation terms. This is also true for distortion estimation for pixels or subpixels under other filtering operations. In Section III, we will extend RMPC algorithm to solve this problem with low complexity.

III. The Extended RMPC Algorithm for Mode Decision

In this section, we first derive a general proposition for any second moment of a weighted sum of correlated random variables; then we apply it to extend RMPC to estimate the end-to-end distortion for subpixel motion compensation and design the algorithm for mode decision.

A. Subpixel-level Distortion Estimation for Error Resilient Video Encoding

Typically, the rate distortion optimized mode decision consists of two steps. First, the R-D cost is calculated by

$$J(\omega_m) = D^k(\omega_m) + \lambda \cdot R(\omega_m), \tag{9}$$

where $D^k = \frac{1}{|\mathcal{V}_l^k|}\sum_{\mathbf{u}\in\mathcal{V}_l^k} D_{\mathbf{u}}^k$; $\mathcal{V}_l^k$ is the set of pixels in the l-th MB (or sub-MB, i.e. any block size for a given prediction mode) of the k-th frame; ω_m is the prediction mode [17]; $R(\omega_m)$ is the encoded bit rate for mode ω_m, $\omega_m \in \Omega_m$ and Ω_m is the mode set for mode decision; λ is the preset Lagrange multiplier. Then, the optimal prediction mode that minimizes the rate-distortion (R-D) cost is found by

$$\hat{\omega}_m = \arg\min_{\omega_m\in\Omega_m}\{J(\omega_m)\}. \tag{10}$$

For the RMPC algorithms, if the MV of one block for encoding is fractional, the MV has to be rounded to the nearest integer. This block will use the reference block pointed to by the rounded MV as a reference. However, in state-of-the-art codecs, such as H.264/AVC and HEVC, interpolation filters are used to interpolate a reference block. Therefore, the distortion of nearest neighbor approximation is not optimal for such an encoder. In order to optimally estimate the distortion for pixels with interpolation filtering, or any other filtering in general, we need to extend the existing RMPC algorithm.

In H.264/AVC, motion compensation accuracy is in units of one quarter of the distance between luma samples. [3] The prediction values at half-sample positions are obtained by applying a one-dimensional 6-tap Finite Impulse Response (FIR) filter horizontally and vertically. The prediction values at quarter-sample positions are generated by averaging samples at integer- and half-sample positions [17]. Take $\widetilde{\zeta}_{\mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k}^{k-j}$ in (5), (6), (7) and (8) for example. Denote $\mathbf{v}^{k-j} = \mathbf{u}+\mathbf{mv}_{\mathbf{u}}^k$ and $\mathbf{v}$ is in a subpixel position in the $k-j$-th frame. All neighboring pixels in the integer position, used to interpolate the reconstructed pixel value at $\mathbf{v}$, are denoted by $\mathbf{v}_i$ and with a weight w_i, $i \in 1,2,...,N$, where $N=6$ for the half-sample

[2] In Ref. [12], [11], we assume that if $S(m)=1$, the reference pixel, pointed by the concealed MV $\check{\mathbf{mv}}_{\mathbf{u}}^k$, comes from the $k-1$-th frame; in this paper, we denote the $k-j'$-th frame for reference pixel pointed by the concealed MV $\check{\mathbf{mv}}_{\mathbf{u}}^k$ without such an assumption.

[3] Note that considering the chroma distortion does not always improve the R-D performance but induces more complexity. Therefore, we only consider luma components in this paper.

$$E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}} + \widetilde{\Delta}^k_{\mathbf{u}}\{\bar{r},\bar{m}\}] = \begin{cases} \hat{f}^k_{\mathbf{u}} - 255, & E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}}] < \hat{f}^k_{\mathbf{u}} - 255 \\ \hat{f}^k_{\mathbf{u}}, & E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}}] > \hat{f}^k_{\mathbf{u}} \\ E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}}], & \hat{f}^k_{\mathbf{u}} - 255 \le E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}}] \le \hat{f}^k_{\mathbf{u}}, \end{cases} \tag{7}$$

$$E[(\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}} + \widetilde{\Delta}^k_{\mathbf{u}}\{\bar{r},\bar{m}\})^2] = \begin{cases} (\hat{f}^k_{\mathbf{u}} - 255)^2, & E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}}] < \hat{f}^k_{\mathbf{u}} - 255 \\ (\hat{f}^k_{\mathbf{u}})^2, & E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}}] > \hat{f}^k_{\mathbf{u}} \\ E[(\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}})^2], & \hat{f}^k_{\mathbf{u}} - 255 \le E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}}] \le \hat{f}^k_{\mathbf{u}}, \end{cases} \tag{8}$$

interpolation, and $N = 2$ for the quarter-sample interpolation in H.264/AVC. Therefore, the interpolated reconstructed pixel value at the encoder is

$$\hat{f}^{k-j}_{\mathbf{v}} = \sum_{i=1}^{N} w_i \cdot \hat{f}^{k-j}_{\mathbf{v}_i}, \tag{11}$$

and at the decoder

$$\widetilde{f}^{k-j}_{\mathbf{v}} = \sum_{i=1}^{N} w_i \cdot \widetilde{f}^{k-j}_{\mathbf{v}_i}. \tag{12}$$

From (2), we have

$$\begin{aligned} \widetilde{\zeta}^{k-j}_{\mathbf{v}} &= \sum_{i=1}^{N} w_i \cdot \hat{f}^{k-j}_{\mathbf{v}_i} - \sum_{i=1}^{N} w_i \cdot \widetilde{f}^{k-j}_{\mathbf{v}_i} \\ &= \sum_{i=1}^{N} w_i \cdot (\hat{f}^{k-j}_{\mathbf{v}_i} - \widetilde{f}^{k-j}_{\mathbf{v}_i}) = \sum_{i=1}^{N} w_i \cdot \widetilde{\zeta}^{k-j}_{\mathbf{v}_i}. \end{aligned} \tag{13}$$

Since $E[\widetilde{\zeta}^{k-j}_{\mathbf{v}_i}]$ has been calculated by the RMPC algorithm, $E[\widetilde{\zeta}^{k-j}_{\mathbf{v}}]$ can be very easily calculated by

$$E[\widetilde{\zeta}^{k-j}_{\mathbf{v}}] = \sum_{i=1}^{N} w_i \cdot E[\widetilde{\zeta}^{k-j}_{\mathbf{v}_i}]. \tag{14}$$

However, calculating $E[(\widetilde{\zeta}^{k-j}_{\mathbf{v}})^2]$ is not straightforward since

$$E[(\widetilde{\zeta}^{k-j}_{\mathbf{v}})^2] = E[(\sum_{i=1}^{N} w_i \cdot \widetilde{\zeta}^{k-j}_{\mathbf{v}_i})^2] \tag{15}$$

is in fact the second moment of a weighted sum of correlated random variables.

B. A Proposition for Calculating the Second Moment of a Weighted Sum of Correlated Random Variables

The Moment Generating Function (MGF) can be used to calculate the second moment for random variables [18]. However, to estimate the second moment of a weighted sum of random variables, the traditional moment generating function usually requires knowing their probability distribution and assumes they are independent. However, in a practical video codec, most random variables, e.g. those involved in the averaging operations, are not independent and their probability distributions are unknown. Therefore, some approximations, such as the Cauchy-Schwarz upper bound approximation [13] or the correlation coefficient model approximation [14], are usually adopted. However, those approximations are of high complexity. For example, for each subpixel, with the N-tap filter interpolation, the Cauchy-Schwarz upper bound approximation requires N integer multiplications for calculating the second moment terms, $N(N-1)/2$ floating-point multiplications and $N(N-1)/2$ square root operations for calculating the cross-correlation terms, and $N(N-1)/2 + N - 1$ additions and 1 shift for calculating the estimated distortion. The correlation coefficient model requires an additional $N(N-1)/2$ exponential operations and $N(N-1)/2$ floating-point multiplications when compared to the Cauchy-Schwarz upper bound approximation.

In a wireless video communication system, the computational capability of the real-time encoder is usually very limited, and floating-point processing is undesirable. Therefore, it is desirable to design a new algorithm for the calculation of the second moment in (15) using only integer operations. Proposition 1 is a result of this motivation.

Proposition 1: For any N correlated random variables $\{X_1, X_2, ..., X_N\}$ and $w_i \in \Re, i \in \{1, 2, ..., N\}$, the second moment of the weighted sum of these random variables is given by (16).

$$\begin{aligned} E[(\sum_{i=1}^{N} w_i \cdot X_i)^2] = &\sum_{i=1}^{N} w_i \cdot \sum_{j=1}^{N} [w_j \cdot E(X_j^2)] - \\ &\sum_{k=1}^{N-1} \sum_{l=k+1}^{N} [w_k \cdot w_l \cdot E(X_k - X_l)^2] \end{aligned} \tag{16}$$

The proof of Proposition 1 is provided below. Note that in H.264/AVC, most averaging operations, e.g., interpolation, deblocking, and bi-prediction, are special cases of Proposition 1 in that $\sum_{i=1}^{N} w_i = 1$. Therefore, we can extend the RMPC algorithm through the consideration of Proposition 1. In (16), since $E(X_j^2)$ has been estimated by the RMPC algorithm, the only unknown is $\sum_{k=1}^{N-1} \sum_{l=k+1}^{N} [w_k \cdot w_l \cdot E(X_k - X_l)^2]$. However, we will see that this unknown can be assumed to be negligible for the purposes of mode decision.

C. The Extended RMPC Algorithm for Mode Decision

1) Algorithm design: Replacing X_k and X_l in (16) by $\widetilde{\zeta}^k_{\mathbf{u}_i}$ and $\widetilde{\zeta}^k_{\mathbf{u}_j}$, and from (2) we have

$$\begin{aligned} E[(X_k - X_l)^2] &= E[(\widetilde{\zeta}^k_{\mathbf{u}_i} - \widetilde{\zeta}^k_{\mathbf{u}_j})^2] \\ &= E[\hat{f}^k_{\mathbf{u}_i} - \widetilde{f}^k_{\mathbf{u}_i} - (\hat{f}^k_{\mathbf{u}_j} - \widetilde{f}^k_{\mathbf{u}_j})]^2 \\ &= E[(\hat{f}^k_{\mathbf{u}_i} - \hat{f}^k_{\mathbf{u}_j}) - (\widetilde{f}^k_{\mathbf{u}_i} - \widetilde{f}^k_{\mathbf{u}_j})]^2. \end{aligned} \tag{17}$$

In (17), both $\hat{f}^k_{\mathbf{u}_i} - \hat{f}^k_{\mathbf{u}_j}$ and $\widetilde{f}^k_{\mathbf{u}_i} - \widetilde{f}^k_{\mathbf{u}_j}$ depend on the spatial correlation of the reconstructed pixel values in position $\mathbf{u}_i$ and $\mathbf{u}_j$. When $\mathbf{u}_i$ and $\mathbf{u}_j$ are located in the same neighborhood, they are very likely to be transmitted in the same packet. In other words, either both $\widetilde{f}^k_{\mathbf{u}_i}$ and $\widetilde{f}^k_{\mathbf{u}_j}$ use the true MV and residual for reconstruction, or both $\widetilde{f}^k_{\mathbf{u}_i}$ and $\widetilde{f}^k_{\mathbf{u}_j}$ use the concealed MV and residual for reconstruction. Therefore, $\widetilde{f}^k_{\mathbf{u}_i} - \widetilde{f}^k_{\mathbf{u}_j}$ will not change too much from $\hat{f}^k_{\mathbf{u}_i} - \hat{f}^k_{\mathbf{u}_j}$, and hence $E[(\widetilde{\zeta}^k_{\mathbf{u}_i} - \widetilde{\zeta}^k_{\mathbf{u}_j})^2]$

Proof:

$$\begin{aligned}
E[(\sum_{i=1}^{N} w_i \cdot X_i)^2] &= E[\sum_{j=1}^{N}(w_j^2 \cdot X_j^2) + \sum_{k=1}^{N}\sum_{\substack{l=1\\(l\neq k)}}^{N}(w_k \cdot w_l \cdot X_k \cdot X_l)] \\
&= E[\sum_{j'=1}^{N} w_{j'} \sum_{j=1}^{N}(w_j \cdot X_j^2) - \sum_{j=1}^{N}\sum_{\substack{j'=1\\(j'\neq j)}}^{N}(w_j \cdot w_{j'} \cdot X_j^2) + \sum_{k=1}^{N}\sum_{\substack{l=1\\(l\neq k)}}^{N}(w_k \cdot w_l \cdot X_k \cdot X_l)] \\
&= \sum_{i=1}^{N} w_i \sum_{j=1}^{N}[w_j \cdot E(X_j^2)] - E\{\sum_{k=1}^{N-1}\sum_{l=k+1}^{N}[w_k \cdot w_l \cdot (X_k^2 + X_l^2)] + \sum_{k=1}^{N-1}\sum_{l=k+1}^{N}(2 \cdot w_k \cdot w_l \cdot X_k \cdot X_l)\} \\
&= \sum_{i=1}^{N} w_i \sum_{j=1}^{N}[w_j \cdot E(X_j^2)] - \sum_{k=1}^{N-1}\sum_{l=k+1}^{N}[w_k \cdot w_l \cdot E(X_k - X_l)^2].
\end{aligned}$$

■

is much smaller than $E[(\widetilde{\zeta}^k_{\mathbf{u}_i})^2]$ and $E[(\widetilde{\zeta}^k_{\mathbf{u}_j})^2]$ in (16). On the other hand, distortion is estimated for one MB or one sub-MB as in (9) for mode decision. When the cardinality $|\mathcal{V}^k_l|$ is large, $\sum_{\mathbf{v}\in\mathcal{V}^k_l}\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}[w_i \cdot w_j \cdot E(\widetilde{\zeta}^k_{\mathbf{u}_i} - \widetilde{\zeta}^k_{\mathbf{u}_j})^2]$ converges to a constant for all modes with high probability due to the summation over the same samples in each mode. For simplicity, we will call it "negligible term" in the following sections. Therefore, in (16) only the first term on the right-hand side needs to be calculated without too much loss in precision.

Since $\sum_{i=1}^{N} w_i = 1$, we calculate $E[(\widetilde{\zeta}^k_{\mathbf{v}})^2]$ for mode decision by

$$E[(\widetilde{\zeta}^k_{\mathbf{v}})^2] = \sum_{i=1}^{N}[w_i \cdot E(\widetilde{\zeta}^k_{\mathbf{u}_i})^2]. \tag{18}$$

With the N-tap filter interpolation, the complexity in (18) is dramatically reduced to only N integer multiplications, $N-1$ additions, and 1 shift. Here, we propose the following algorithm to extend the RMPC algorithm for mode decision.

Algorithm 1: Rate distortion optimized mode decision for an MB in the k-th frame ($k >= 1$).

1) **Input**: QP, PLP.
2) Initialization of $E[\widetilde{\zeta}^0_{\mathbf{u}}]$ and $E[(\widetilde{\zeta}^0_{\mathbf{u}})^2]$ for all pixel $\mathbf{u}$.
3) Loop for all available modes for each MB.
   3a) estimate $E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{m}\mathbf{v}^k_{\mathbf{u}}}]$ via (14) and $E[(\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{m}\mathbf{v}^k_{\mathbf{u}}})^2]$ via (18) for all pixels in the MB,
   3b) estimate $E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{m}\mathbf{v}^k_{\mathbf{u}}} + \widetilde{\Delta}^k_{\mathbf{u}}\{\bar{r},\bar{m}\}]$ via (7) and $E[(\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{m}\mathbf{v}^k_{\mathbf{u}}} + \widetilde{\Delta}^k_{\mathbf{u}}\{\bar{r},\bar{m}\})^2]$ via (8) for all pixels in the MB,
   3c) estimate $E[\widetilde{\zeta}^k_{\mathbf{u}}]$ via (5) and $E[(\widetilde{\zeta}^k_{\mathbf{u}})^2]$ via (6) for all pixels in the MB,
   3d) estimate $D^k_{\mathbf{u}}$ via (4) for all pixels in the MB,
   3e) estimate R-D cost for the MB via (9),
   End
4) Via (10), select the best mode with minimum R-D cost for the MB.
5) **Output**: the best mode for the MB.

Algorithm 1 is referred to as the Extended RMPC (ERPMC). Note that if an MV packet is lost, in order to reduce both the MV concealment and distortion estimation complexity, the concealed MV $\breve{\mathbf{m}}\mathbf{v}^k_{\mathbf{u}}$ does not necessary use fractional accuracy. That is, the ERMPC algorithm conceals the MV with integer accuracy. Therefore, $E[\widetilde{\zeta}^{k-j'}_{\mathbf{u}+\breve{\mathbf{m}}\mathbf{v}^k_{\mathbf{u}}}]$ and $E[(\widetilde{\zeta}^{k-j'}_{\mathbf{u}+\breve{\mathbf{m}}\mathbf{v}^k_{\mathbf{u}}})^2]$ in (5) and (6) do not require (18).

2) Complexity analysis: In Ref. [11], we compare the complexity of RMPC to that of ROPE and LLN in theory. In this section, we first compare the complexity of ERMPC to RMPC in theory. Then, we test the complexity of ERMPC/RMPC/ROPE in JM16.0 and compare them to the default ERRDO algorithm in JM16.0, i.e. LLN; we also test the RDO without error-resilient algorithms as a benchmark.

RMPC computational complexity is calculated from (4), (5), (6), (7) and (8), where the first moment and the second moment of the reconstructed error of the best mode should be stored after the mode decision (Note that the reconstructed error in previous frames could be regarded as the propagated error in the current frame, recursively). Therefore, 2 units of memory are required to store those two moments for each pixel. Note that the first moment $E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{m}\mathbf{v}^k_{\mathbf{u}}}]$ takes values in $\{-255,-254,...,255\}$, i.e., 8 bits plus 1 sign bit per pixel, and the second moment $E[(\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{m}\mathbf{v}^k_{\mathbf{u}}})^2]$ takes values in $\{0,1,...,255^2\}$, i.e., 16 bits per pixel. From the discussion above, we know that the difference, in computational complexity, of ERMPC compared to RMPC is the estimation of $E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{m}\mathbf{v}^k_{\mathbf{u}}}]$ in (7) and $E[(\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{m}\mathbf{v}^k_{\mathbf{u}}})^2]$ in (8). To be more specific, in ERMPC $E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{m}\mathbf{v}^k_{\mathbf{u}}}]$ and $E[(\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{m}\mathbf{v}^k_{\mathbf{u}}})^2]$ include a fractional MV while in RMPC $E[\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{m}\mathbf{v}^k_{\mathbf{u}}}]$ and $E[(\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{m}\mathbf{v}^k_{\mathbf{u}}})^2]$ include only an integer MV. Therefore, the additional complexity of ERMPC compared to RMPC is the calculation of (14) and (18).

Note that (14) and (18) are not needed in ERMPC if: 1) the prediction mode is intra; 2) the MV is an integer MV. For MVs pointing to half-pixel positions, the values of $\{w_i\}$ are $\{1,-5,20,20,-5,1\}$ in (14) and (18); i.e., there are $N=6$ integer multiplications, $N-1=5$ additions, and 1 right shift with 5 bits to make sure $\sum_{i=1}^{N} w_i = 1$. By using the same simplification of the JM interpolation implementation, (18) can be calculated by $E(\widetilde{\zeta}^k_{\mathbf{u}_{3.5}})^2 = \{20 \cdot [E(\widetilde{\zeta}^k_{\mathbf{u}_3})^2 + E(\widetilde{\zeta}^k_{\mathbf{u}_4})^2] - 5 \cdot [E(\widetilde{\zeta}^k_{\mathbf{u}_2})^2 + E(\widetilde{\zeta}^k_{\mathbf{u}_5})^2] + [E(\widetilde{\zeta}^k_{\mathbf{u}_1})^2 + E(\widetilde{\zeta}^k_{\mathbf{u}_6})^2] + 16\} >> 5$. In fact, we may further simplify the implementation of (18) by replacing the integer multiplication operations by additions

and shifts as

$$E(\widetilde{\zeta}^k_{\mathbf{u}_{3.5}})^2 = \{[E(\widetilde{\zeta}^k_{\mathbf{u}_3})^2 + E(\widetilde{\zeta}^k_{\mathbf{u}_4})^2] \ll 4 + [E(\widetilde{\zeta}^k_{\mathbf{u}_3})^2 + E(\widetilde{\zeta}^k_{\mathbf{u}_4})^2] \ll 2 - [E(\widetilde{\zeta}^k_{\mathbf{u}_2})^2 + E(\widetilde{\zeta}^k_{\mathbf{u}_5})^2] \ll 2 - [E(\widetilde{\zeta}^k_{\mathbf{u}_2})^2 + E(\widetilde{\zeta}^k_{\mathbf{u}_5})^2] + [E(\widetilde{\zeta}^k_{\mathbf{u}_1})^2 + E(\widetilde{\zeta}^k_{\mathbf{u}_6})^2] + 16\} \gg 5. \quad (19)$$

Note that in (19), $E(\widetilde{\zeta}^k_{\mathbf{u}_3})^2 + E(\widetilde{\zeta}^k_{\mathbf{u}_4})^2$ and $E(\widetilde{\zeta}^k_{\mathbf{u}_2})^2 + E(\widetilde{\zeta}^k_{\mathbf{u}_5})^2$ are invoked two times, but are counted only once since the temporary result is saved in the CPU register. As a result, the half-pixel position needs 8 additions (ADDs) and 4 shifts only.

For quarter-pixel positions additional computations will need to be performed. We observe that there are two practical implementations for ERMP, each one having different computational complexity and memory requirements. In the first, we can store the first and second moments of the transmission error for half-pixel positions, which will require 3 times the memory compared to storing only the integer pixel moments. However, this will reduce the complexity of calculating the moments for quarter-pixel positions. In the second, the moments for all fractional positions are calculated on the fly. This is also the method used in the LLN algorithm available in the JM. The complexity of this method is analyzed below. Note that the computational complexity of calculating (14) and (18) for different subpixel positions is different.

In this paper we only show the complexity analysis for those half-pixel positions interpolated by integer pixels. It is very easy to extend this analysis for all fractional positions. Since (14) and (18) are calculated on the fly, there is no additional memory requirement for ERMPC. Note that in most CPUs, a shift can be integrated into an ADD, therefore not impacting complexity. As a result, the complexity of ERMPC can be found in Table II [4]. For LLN, half-pixel position motion compensation requires 8 ADDs and 4 shifts more than LLN with integer-pixel position motion compensation for each simulated decoder; that is LLN is $8N_d$ ADDs more than the complexity of that in Ref. [11], where N_d means the number of simulated decoders at the encoder; the default value in JM is $N_d = 30$. We also cite the complexity of ROPE and LLN with integer MV accuracy from Ref. [11] in Table II for reference.

Since the theoretical complexity comparison only accounts for half-pixel positions, it would be beneficial to evaluate complexity in a real encoding environment. In reality, MVs could point to any position, integer or fractional. The nearest integer MV is used to approximate the fractional MV in RMPC and ROPE. NO_ERRDO means the normal RDO mode decision process, without any error resiliency considerations available in the JM. Table III shows the results for the mobile sequence at CIF resolution with $PLP = 5\%$. The experimental setup is the same as those in Section IV. A system based on an AMD Opteron(tm) 2356 processor at 2.29GHz was used. It can be seen that the executation time of ERMPC/RMPC/ROPE is only slightly higher of that of ERRDO. However, LLN requires considerable more execution time than these schemes. Other sequences and channel conditions show similar results.

[4]Note that in H.264/AVC seven inter prediction modes are supported, i.e., 16×16, 16×8, 8×16, 8×8, 8×4, 4×8, and 4×4. Nine intra 4×4 and 8×8 modes, as well as four 16×16 modes for luma intra prediction are supported. Total complexity is calculated for all prediction modes.

D. Merits and Limitations of ERMPC Algorithm

1) Merits: Since both the Cauchy-Schwarz upper bound approximation [13] and the correlation coefficient model approximation [14] induce floating-point multiplications, round-off error is unavoidable. The algorithm by Yang et al. [14] needs extra complexity to mitigate the effect of round-off error during distortion estimation. In contrast, one of the merits of Proposition 1 is that it only needs integer multiplications and additions. In H.264/AVC and HEVC, w_i (and $w_i \cdot w_j$) can be scaled to an integer value without any round-off error for all coding modes. As a result, round-off error can be avoided in the ERMPC algorithm.

In Ref. [19], the authors prove that a low-pass interpolation filter will decrease the frame-level propagated error under some assumptions. In fact, it is easy to prove that when $\sum_{i=1}^{N} w_i = 1$ and $|\mathcal{V}_l^k|$ is large, the negligible term is larger than or equal to zero. Even at the MB-level, the negligible term is larger than or equal to zero with very high probability. From (16), we see that the block-level distortion decreases, with very high probability, after the interpolation filtering.

One additional benefit of (16) is to guide the design of the interpolation filter. Traditional interpolation filter design aims to minimize the prediction error. With (16), we may design an interpolation filter by maximizing $\sum_{k=1}^{N}\sum_{l=k+1}^{N}[w_k \cdot w_l \cdot E(X_k - X_l)^2]$ under the constraint of $\sum_{j=1}^{N}[w_j \cdot E(X_j^2)]$.

2) Limitations: In Algorithm 1, the second moment of propagated error $E[(\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}})^2]$ is estimated by ignoring the negligible term to reduce the complexity. A more accurate alternative method is to estimate $E(\widetilde{\zeta}^k_{\mathbf{u}_i} - \widetilde{\zeta}^k_{\mathbf{u}_j})^2]$ recursively by storing the value in memory. This will be considered in our future work.

IV. Experimental Results

In this section, we compare the R-D performance and subjective performance of the ERMPC algorithm with that of the RMPC and the LLN algorithms for mode decision in H.264/AVC. Since the original ROPE does not support the interpolation filtering operation and its extensions [13], [14] induce many floating-point operations and round-off errors, we only use the same nearest integer MV approximation to show how its R-D performance differs from ERMPC, RMPC, and LLN. To compare all algorithms under multi-reference picture motion compensated prediction, we also enhance the original ROPE algorithm [10] with multi-reference capability.

A. Experimental Setup

The JM encoder and decoder were used in the experiments. The first 100 frames from several CIF resolution, 30fps test video sequences were tested under different PLP settings from 0.5% to 5%. The co-located pixel copy from the previous frame method was used for error concealment in all algorithms. The first frame is assumed to be correctly received. The High profile of H.264/AVC, using CABAC for entropy coding but without B slices, with 3 slices per picture and 3 reference frames, was used. Constrained intra prediction was also enabled. In the LLN algorithm, the number of simulated decoders is 30.

TABLE II
COMPLEXITY COMPARISON IN THEORY

Algorithms		computational complexity	memory requirement
ERMPC (half-pixel)	inter mode	25 ADDs, 8 MULs	25 bits/pixel
	intra mode	7 ADDs, 6 MULs	
	total complexity	266 ADDs, 134 MULs	
RMPC	inter mode	9 ADDs, 8 MULs	25 bits/pixel
	intra mode	7 ADDs, 6 MULs	
	total complexity	154 ADDs, 134 MULs	
LLN (half-pixel)	inter mode	$10N_d$ ADDs, N_d MULs	$8N_d$ bits/pixel
	intra mode	N_d ADDs, N_d MULs	
	total complexity	$83N_d$ ADDs, $20N_d$ MULs	
LLN (integer pixel)	inter mode	$2N_d$ ADDs, N_d MULs	$8N_d$ bits/pixel
	intra mode	N_d ADDs, N_d MULs	
	total complexity	$27N_d$ ADDs, $20N_d$ MULs	
ROPE	inter mode	7 ADDs, 8 MULs	24 bits/pixel
	intra mode	4 ADDs, 7 MULs	
	total complexity	101 ADDs, 147 MULs	

TABLE III
COMPLEXITY COMPARISON IN EXPERIMENT

Algorithm	ERMPC	RMPC	LLN	ROPE	NO_ERRDO
Time in second	105.828	105.126	137.393	104.766	102.719

B. R-D Performance

Due to space limitations, we only show the plots of PSNR vs. bit rate for video sequences 'foreman' and 'mobile' under $PLP = 0.5\%$ and $PLP = 2\%$ in Figs. 2 and 3 respectively. The experimental results show that ERMPC achieves the best R-D performance; RMPC achieves the second best R-D performance; ROPE achieves better performance than LLN in some cases such as at high rate in Fig. 2, but worse performance than LLN in other cases such as in Fig. 3 and at the low rate in Fig. 2.

It is interesting to see that for some sequences and channel conditions, ERMPC achieves a notable PSNR gain over RMPC. This is, for example, evident in 'mobile' and 'foreman'. For some other cases, however, ERMPC only achieves a marginal PSNR gain over RMPC (e.g., 'coastguard' and 'football'). From the analysis in Section III-A, we know that the only difference between RMPC and ERMPC is the estimate of the propagated error $\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}}$ in (7) and (8). Therefore, the performance gain of ERMPC over RMPC only comes from inter modes, since they both use exactly the same estimates for intra modes. Thus, the higher percentage of intra modes in 'coastguard' and 'football' may result in a marginal PSNR gain of ERMPC over RMPC.

For most sequences and channel conditions, we observe that in most cases the higher the bit rate for encoding, the more the PSNR gain of ERMPC over RMPC, such as in Fig. 2 and Fig. 3(a). In (4), the end-to-end distortion consists of both quantization and transmission distortion. The ERMPC algorithm gives a more accurate estimation of propagated error in transmission distortion than the RMPC algorithm. When the bit rate for source encoding is very low, with rate control the controlled Quantization parameter (QP) is large, and hence the quantization distortion becomes the dominant part in the end-to-end distortion. Therefore, the PSNR gain of ERMPC over RMPC is marginal. On the contrary, when the bit rate for source encoding is high, the transmission distortion becomes the dominant part in the end-to-end distortion. Therefore, the PSNR gain of ERMPC over RMPC is notable. However, this is not always true as observed in Fig. 3(b). The reason is as follows. In the JM, the Lagrange multiplier in (9) is a function of QP. A higher bit rate or smaller QP also causes a smaller Lagrange multiplier; therefore, the rate cost in (9) becomes smaller, which may produce a higher percentage of intra modes. In such a case, the PSNR gain of ERMPC over RMPC decreases. As a result, different sequences give different results depending on whether the effect of increase of intra modes dominates over the effect of decrease of quantization distortion.

LLN has poorer R-D performance than ERMPC. This may be since 30 simulated decoders are still not enough to achieve a reliable distortion estimate. Meanwhile, complexity increase is considerable compared to ERMPC. It is also interesting to see that the integer MV approximation for ROPE is only valid for some sequences, such as 'foreman', while this approximation gives poor R-D performance for some other sequences, such as 'mobile'. However, the nearest neighbor approximation for RMPC in all sequences achieves good performance. This is because RMPC approximates the first and second moments of the propogated error $\widetilde{\zeta}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}}$ by the rounded MV, while ROPE approximates the first and second moments of the reference pixel value $\widetilde{f}^{k-j}_{\mathbf{u}+\mathbf{mv}^k_{\mathbf{u}}}$ by the rounded MV. Since the propagated errors are much smaller and more stable than the reference pixel values, RMPC shows better and more stable performance using integer MV approximation.

Table IV shows the average PSNR gain (in dB) of ERMPC over RPMC, LLN, and ROPE for different video sequences and different PLP. The average PSNR gain is obtained using the BD-PSNR method in Ref. [20], which measures the average distance (in PSNR) between two R-D curves. From Table IV, we see that ERMPC achieves an average PSNR gain of 0.25dB over RMPC for the sequence 'mobile' under $PLP = 2\%$; it achieves an average PSNR gain of 1.34dB over LLN for the 'foreman' sequence under $PLP = 1\%$; and it achieves an average PSNR gain of 3.18dB over ROPE for the 'mobile' sequence under $PLP = 0.5\%$.

C. Subjective Performance

Since PSNR may not be as meaningful for error concealment, subjective performance is also evaluated. Fig. 4 shows the subjective quality of the 84-th frame and the 99-th frame of 'foreman' sequence under a PLP of 1% and a bit rate of

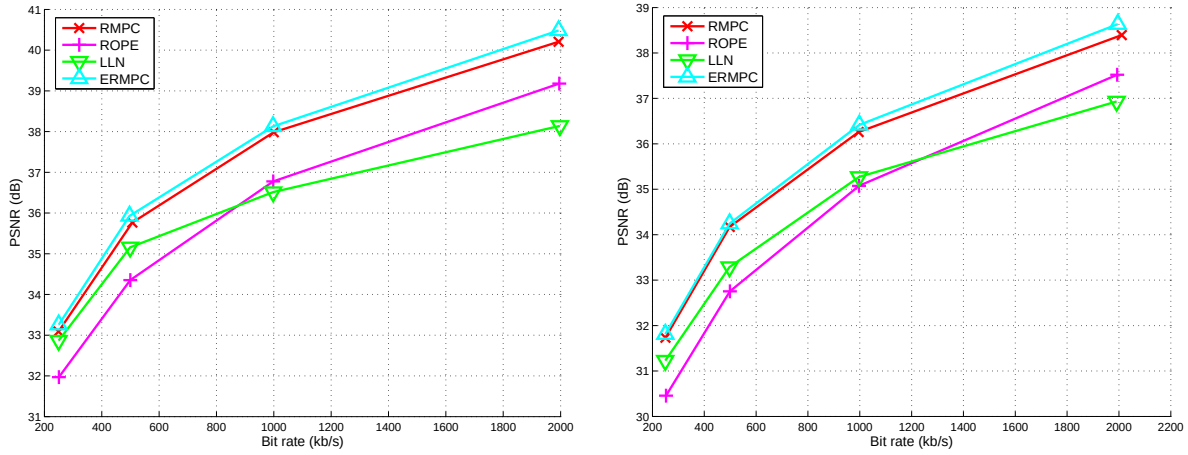


Fig. 2. PSNR vs. bit rate for 'foreman': (a) PLP=0.5%, (b) PLP=2%.

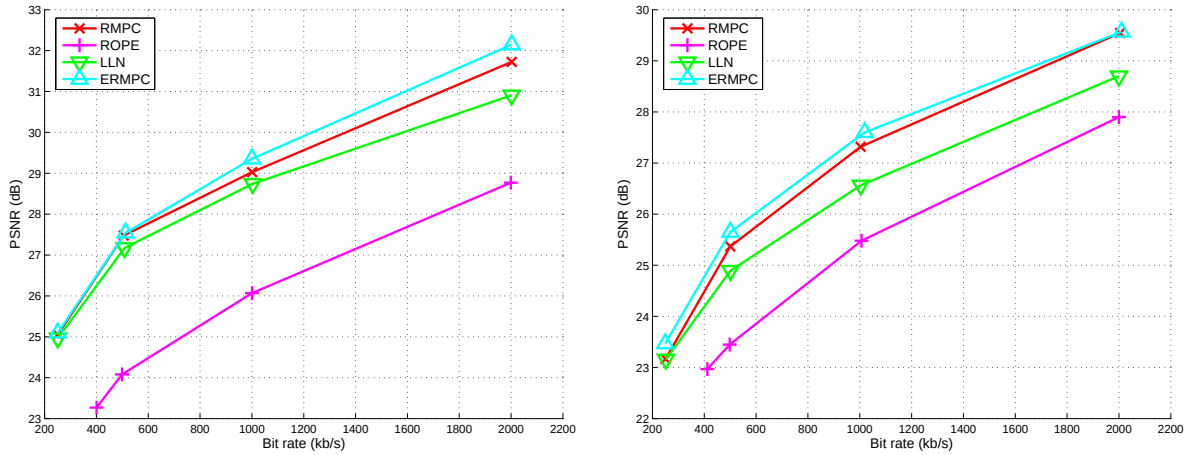


Fig. 3. PSNR vs. bit rate for 'mobile': (a) PLP=0.5%, (b) PLP=2%.

TABLE IV
AVERAGE PSNR GAIN (IN DB) OF ERMPC OVER RMPC, LLN AND ROPE

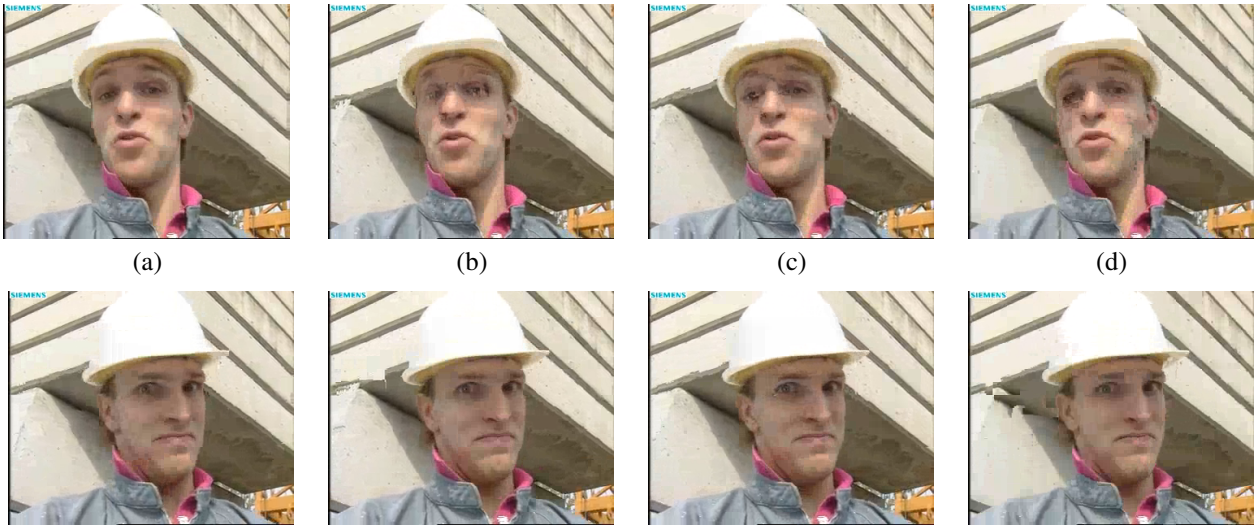
Sequence	coastguard				football				foreman				mobile			
PLP	5%	2%	1%	0.5%	5%	2%	1%	0.5%	5%	2%	1%	0.5%	5%	2%	1%	0.5%
ERMPC vs. RMPC	0.09	0.08	0.08	0.06	0.01	0.01	0.01	0.03	0.08	0.13	0.21	0.17	0.20	**0.25**	0.21	0.21
ERMPC vs. LLN	0.32	0.36	0.46	0.37	0.28	0.39	0.36	0.26	0.64	1.07	**1.34**	1.24	0.50	0.82	0.56	0.54
ERMPC vs. ROPE	0.58	0.46	0.52	0.62	0.47	0.25	0.27	0.33	1.59	1.37	1.41	1.42	1.11	1.89	2.79	**3.18**

250kbps. These results suggest a similar performance as those presented in Section IV-B. We can thus conclude that ERMPC achieves the best performance.

D. Discussion

1) Effect of clipping noise on the mode decision: Since ROPE does not consider the effect of clipping noise on the transmission distortion, it over-estimates the end-to-end distortion for inter modes. Hence, ROPE would tend to select intra modes more often than ERMPC, RMPC, and LLN, which will lead to higher encoding bit rates. To verify this conjecture, we tested all sequences under the same QP settings, from 20 to 32, without rate control. We observed that the ROPE algorithm always produced a higher bit rate than other schemes as shown in Fig. 5 and Fig. 6.

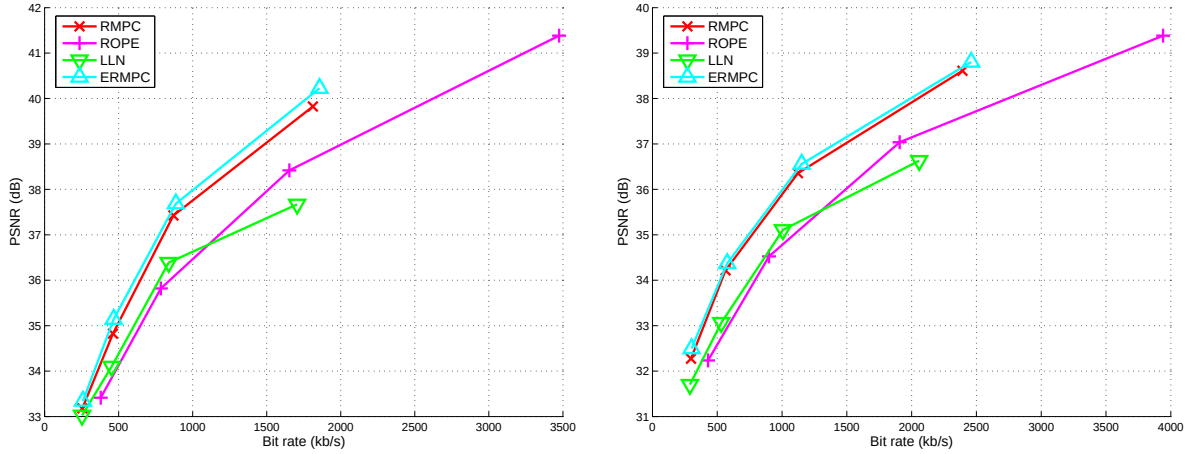
2) Effect of transmission errors on mode decision: One can observe three characteristics for ERMPC/RMPC/LLN/ROPE algorithms vs NO_ERRDO. 1) The number of intra MBs increases since the transmission error is accounted for during mode decision; 2) The number of skip mode MBs also increases, since the transmission error will increase the transmission distortion in all other modes except for this mode; 3) if we allow the first frame to be erroneous, the second frame will have high percentage of intra MBs. This is because only the value of 128 can be used to conceal the reconstructed pixel values if the first frame is lost, while if other frames are lost the collocated pixel in the previous frame can be used to conceal the reconstructed pixel values. Therefore, the propagated error from the first frame will be much higher than the error from other frames. As a result, if the first frame is allowed to be lost with a certain probability, the second frame will contain a high percentage of intra MBs due to ERRDO.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 4. (a) ERMPC at the 84-th frame, (b) RMPC at the 84-th frame, (c) LLN at the 84-th frame, (d) ROPE at the 84-th frame, (e) ERMPC at the 99-th frame, (f) RMPC at the 99-th, (g) LLN at the 99-th frame, (h) ROPE at the 99-th frame.



(a) (b)

Fig. 5. PSNR vs. bit rate for 'foreman': (a) PLP=0.5%, (b) PLP=2%.

V. Conclusion

In this paper, we proved a new proposition for calculating the second moment of a weighted sum of correlated random variables without requiring knowledge of the random variable probability distributions. Then, we apply this proposition to extend our previous RMPC algorithm in estimating the fractional-level end-to-end distortion for prediction mode decision without significantly increasing complexity. Experimental results show that ERMPC achieves on average a PSNR gain of 0.25dB over the existing RMPC algorithm for the 'mobile' sequence when PLP equals 2%; ERMPC achieves an average PSNR gain of 1.34dB over the the LLN algorithm for the 'foreman' sequence when PLP equals 1%. Experimental results also show that subjective quality was also improved.

References

[1] C. E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," *IRE Nat. Conv. Rec. Part*, vol. 4, pp. 142–163, 1959.

[2] T. Berger, *Rate distortion theory: A mathematical basis for data compression*. Prentice-Hall, Englewood Cliffs, NJ, 1971.

[3] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers." *IEEE Trans. Acoust. Speech Signal Process.*, vol. 36, no. 9, pp. 1445–1453, 1988.

[4] H. Everett III, "Generalized Lagrange multiplier method for solving problems of optimum allocation of resources," *Operations Research*, vol. 11, no. 3, pp. 399–417, 1963.

[5] A. Ortega and K. Ramchandran, "Rate-distortion methods for image and video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 23–50, 1998.

[6] G. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 74–90, 1998.

[7] "H.264/AVC reference software JM16.0," Jul. 2009. [Online]. Available: http://iphome.hhi.de/suehring/tml/download

[8] T. Stockhammer, M. Hannuksela, and T. Wiegand, "H. 264/AVC in wireless environments," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 657–673, 2003.

[9] T. Stockhammer, T. Wiegand, and S. Wenger, "Optimized transmission of h.26l/jvt coded video over packet-lossy networks," in *IEEE ICIP*, 2002.

[10] R. Zhang, S. L. Regunathan, and K. Rose, "Video coding with optimal inter/intra-mode switching for packet loss resilience," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, pp. 966–976, Jun. 2000.

[11] Z. Chen and D. Wu, "Prediction of Transmission Distortion for Wireless Video Communication: Algorithm and Application," *Journal of Visual Communication and Image Representation*, vol. 21, no. 8, pp. 948–964, 2010.

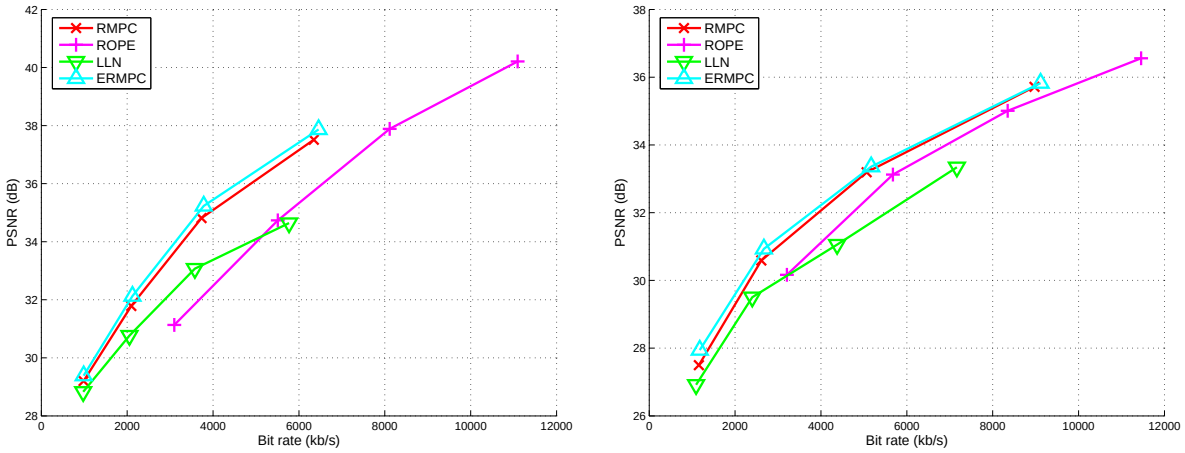


Fig. 6. PSNR vs. bit rate for 'mobile': (a) PLP=0.5%, (b) PLP=2%.

[12] ——, "Prediction of Transmission Distortion for Wireless Video Communication: Part I: Analysis," *IEEE Transactions on Image Processing*, 2011, accepted.

[13] A. Leontaris and P. Cosman, "Video compression for lossy packet networks with mode switching and a dual-frame buffer," *IEEE Transactions on Image Processing*, vol. 13, no. 7, pp. 885–897, 2004.

[14] H. Yang and K. Rose, "Advances in recursive per-pixel end-to-end distortion estimation for robust video coding in H. 264/AVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 7, p. 845, 2007.

[15] *ITU-T Series H: Audiovidual and Multimedia Systems, Advanced video coding for generic audiovisual services*, Nov. 2007.

[16] T. Wiegand, W.-J. Han, B. Bross, J.-R. Ohm, and G. J. Sullivan, *WD1: Working Draft 1 of High-Efficiency Video Coding*, Guangzhou, Oct. 2010, JCTVC-C403, 3rd JCT-VC Meeting.

[17] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[18] G. Casella and R. L. Berger, *Statistical Inference*, 2nd ed. Duxbury Press, 2001.

[19] K. Stuhlmuller, N. Farber, M. Link, and B. Girod, "Analysis of video transmission over lossy channels," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 1012–1032, Jun. 2000.

[20] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves, 13th VCEG-M33 Meeting," *Austin, USA*, 2001.

Zhifeng Chen received Ph.D. degree in Electrical and Computer Engineering from the University of Florida, Gainesville, Florida, in 2010. From 2002 to 2003, he was an engineer in EPSON (China), and from 2003 to 2006, he was a senior engineer in Philips (China), both working in mobile phone system solution design. He joined Interdigital Inc. in 2010, where he is currently a staff engineer working on video coding research.

Peshala V. Pahalawatta received his Ph.D. degree in electrical engineering from Northwestern University, Evanston, IL, in 2007. He is currently a Staff Engineer with the Image Technology group at Dolby Laboratories Inc., Burbank, CA. His research interests include image and video compression and transmission, image and video quality evaluation, and computer vision.

Alexis M. Tourapis received the Diploma degree in Electrical and Computer Engineering from the National Technical University of Athens (NTUA), Greece, in 1995 and the Ph.D. degree in Electrical and Electronic Engineering from the Hong Kong University of Science & Technology, HK, in 2001. Alexis has held in the past various research and development positions with companies such as Microsoft, Thomson, DoCoMo Labs USA, and Dolby Laboratories. He is currently with Magnum Semiconductor Inc. as a Senior Director of Video Algorithm Engineering focusing on the development of next generation video processing and compression hardware system designs.

Alexis is a senior member of the IEEE, and a member of the ACM, SPIE, and SMPTE. In 2000 he received the IEEE HK section best postgraduate student paper award for his work, and in 2006 he was acknowledged as one of 10 most outstanding reviewers by the IEEE Transactions on Image Processing. Alexis currently holds 7 US patents and has more than 90 US and international patents pending. Alexis has made several contributions to several video coding standards, and in particular to H.264/MPEG-4 AVC, on a variety of topics, such as motion estimation and compensation, rate distortion optimization, rate control and others, and currently serves as a co-chair of the development activity on the H.264 Joint Model (JM) reference software.

Dapeng Wu (S'98–M'04–SM'6) received Ph.D. in Electrical and Computer Engineering from Carnegie Mellon University, Pittsburgh, PA, in 2003. Currently, he is a professor of Electrical and Computer Engineering Department at University of Florida, Gainesville, FL.