EAVE: Error-Aware Video Encoding Supporting Extended Energy/QoS Tradeoffs for Mobile Embedded Systems[1]

KYOUNGWOO LEE
University of California, Irvine
NIKIL DUTT
University of California, Irvine
and
NALINI VENKATASUBRAMANIAN
University of California, Irvine

Energy/QoS provisioning is challenging for video applications over lossy wireless network with power-constrained mobile handheld devices. In this work, we exploit the inherent error-tolerance of video data to generate a range of acceptable operating points by controlling the amount of errors in the system. In particular, we propose an error-aware video encoding technique – *EAVE* – that intentionally injects errors while ensuring acceptable QoS. The expanded tradeoff space generated by EAVE allows system designers to comparatively evaluate different operating points with varying QoS and energy consumption by aggressively exploiting error-resilience attributes, and can potentially result in significant energy savings. The novelty of our approach resides in *active exploitation of errors* to vary the operating conditions for further optimization of system parameters. Moreover, we present the adaptivity of our approach by incorporating the feedback from the decoding side to achieve the QoS requirement under the dynamic network status. Our experiments show that *EAVE* can reduce the energy consumption for an encoding device by up to 37% for a video conferencing application over a wireless network without quality degradation, compared to a standard video encoding technique for test video streams. Further, our experimental results demonstrate that *EAVE* can expand the design space by 14 times with respect to energy consumption and by 13 times with respect to video quality, compared to a traditional approach without active error exploitation, on average over test video streams.

[1]This is an expanded version of a paper published in the Proceedings of the IFIP Working Conference on Distributed and Parallel Embedded Systems (DIPES) 2008. The current manuscript extends the previous paper by (i) generalizing our approach as an error-aware video encoding newly named EAVE in Section 4, (ii) evaluating two more error-aware video encodings based on PGOP and GOP in Section 4.3, (iii) proposing an intelligent frame dropping technique in Section 4.2.2 and a method to adjust an error rate for further energy/QoS tradeoffs in Section 4.2.1, (iv) presenting the comprehensive related work in Section 2, and (v) demonstrating the effectiveness of our proposals with comprehensive experimental results in Section 6.

Author's address: K. Lee, N. Dutt, and N. Venkatasubramanian, Department of Computer Science, University of California, Irvine, CA 92697.

Categories and Subject Descriptors: []: —;

General Terms:

Additional Key Words and Phrases:

1. INTRODUCTION

Due to the rapid deployment of wireless communications, video applications on mobile embedded systems such as video telephony and video streaming have grown dramatically. A major challenge in mobile video applications is how to efficiently allocate the limited energy resource in order to deliver the best video quality. A significant amount of power in mobile embedded systems is consumed by video processing and transmission. Also, error resilient video encodings demand extra energy consumption in general to combat the transmission errors in wireless video communications. Thus, it is challenging and essential for system designers to explore the possible tradeoff space and to increase the energy savings while ensuring the quality satisfaction even under dynamic network status. In this article, we introduce the notion of *active error exploitation* to effectively extend the tradeoff space between energy consumption and video quality, and present EAVE, an adaptive error-exploiting video encoding strategy to maximize the energy saving with minimal quality degradation. EAVE also enables design space exploration by generating multiple feasible design points with varying QoS and energy characteristics.

Tradeoffs between energy consumption and QoS (Quality of Service) for mobile video communications have been investigated earlier [Taylor et al. 2001; Eisenberg et al. 2002; Mohapatra et al. 2003; Yuan et al. 2003; Harris et al. 2005; Mohapatra et al. 2005]. It is interesting to observe that the delivered video data is *inherently error-tolerant*: spatial and temporal correlations between consecutive video frames are used to increase the compression efficiency, and result in errors at the reconstructed video data. Also, a high quantization scale causes a high loss of video data. Although naturally induced errors and losses from the encoding algorithms degrade the video quality, this loss of quality may not be perceived by the human eye. This inherent error-tolerance of video data can be exploited to reduce the energy consumption for battery-limited mobile embedded systems. For instance, relaxing the acceptable quality of the delivered video reduces the overhead for an exhaustive searching algorithm during encoding by exploring a partial area rather than the entire region. Further, we can exploit errors actively for the purpose of energy reduction. In our study, one way of active error exploitation is to *intentionally drop frames* before the encoding process. By dropping frames (a process similar to sampling in video processing), we eliminate the entire video encoding process for these frames and thereby reduce energy consumption while sacrificing some loss in the QoS of the delivered video stream. Note that the detrimental effects of dropping frames on the video quality are partially compensated by the inherent error-tolerance of video data.

To cope with transmission errors such as packet losses due to the congested routers and faded access points in wireless communication, error-resilient video encoding techniques [Wang et al. 2000; Zhang et al. 2000; Worrall et al. 2001; Cheng and Zarki 2004; Kim et al. 2006] have been investigated to reduce the effects of transmission errors on the QoS. Most existing error resilient techniques judiciously adapt their resilience levels considering the network status such as packet loss rates.

The joint approach we present in this work combines these error-resilient techniques with intentional dropping frames, presents several pros and cons. First, we can improve the video quality by applying error-resilient video encoding techniques to the video stream with frame drops disguised as network packet losses. Second, we can increase the error margins that video encoders can exploit for maximal energy reduction, i.e., we can drop more frames. On the other hand, the error-resilient techniques increase the size of the compressed video data in general, which raises the energy consumption for data transmission. Consequently, our joint approach that combines active error-exploitation approach with error-resilient techniques significantly enlarges the tradeoff space among energy consumption for compression, energy consumption for transmission, and QoS in mobile video applications. Furthermore, our error exploiting video encoding scheme extends the applicability of error resilient schemes, even when the network is error-free.

In this article, we propose a new tradeoff knob, *error injection rate* (*EIR*), that controls the amount of data to be dropped. This EIR knob can be used to explore the tradeoff space between the energy consumption and video quality, unlike in previous approaches. Specifically, we present a new error-aware video encoding scheme using existing error-resilient video encodings such as PBPAIR (Probability-Based Power-Aware Intra-Refresh) [Kim et al. 2006] and PGOP (Progressive Group-Of-Picture) [Cheng and Zarki 2004]. Our new approach, called *Error-Aware Video Encoding* or *EAVE*, is composed of two units: an error-injection unit and an error-canceling unit. The error-injection unit drops frames intentionally according to the EIR to save energy consumption; the error-canceling unit applies previously proposed error-resilient video encodings to compress video data resilient against intentional frame drops in an energy-efficient manner. Active error exploitation can reduce the overheads for transmission and even the decoding, and result in the end-to-end energy savings of all components in an encoding-decoding path in mobile video embedded systems. However, injecting errors very aggressively in EAVE can degrade the video quality significantly, creating a need to monitor the delivered video quality in distributed video applications and to adjust the error injection rate to ensure the satisfactory quality. Thus, we also present *adaptive EAVE*, which adapts the error injection rate based on the quality feedback from the decoding side while minimizing the energy consumption.

The main contributions of our work are listed below:

—We propose the notion of active error exploitation, that significantly extends the energy/QoS tradeoff space for video encodings on power-constrained mobile embedded systems.

—We present error-aware video encoding techniques such as *EA-PBPAIR*, *EA-PGOP*, and *EA-GOP* by dropping frames intentionally in accordance with existing video encodings such as PBPAIR, PGOP, and GOP.

—We present *adaptive EAVE*, a feedback-based quality adjustment technique that adapts the error injection rate to meet the quality constraint.

—We demonstrate the efficacy of our approach: as compared to a traditional video encoding based on H.263 [ITU-T 1996], our EA-PBPAIR technique can reduce the energy consumption of an encoding device by 37% on average over a set of video streams without quality degradation, and by 49% at the cost of 10% quality degradation.

—We demonstrate the ability to explore a large design space: as compared to a traditional video encoding, our error-aware video encoding can expand the design space by 14 times with respect to the energy consumption and by 13 times with respect to the QoS on average over test video streams.

2. RELATED WORK

Mobile video applications are challenging due to multiple constraints such as video quality, energy consumption, and error resilience. Researchers have studied the algorithms and parameters in video encoding processes, and devised *knobs* to satisfy those multi-dimensional *constraints*. Fig. 1 broadly classifies previously proposed video encodings into standard video encoders, energy-efficient video encoders, and error-resilient video encoders, and the knobs they have devised to satisfy the constraints they have considered. For instance, to satisfy the quality constraint, video encoding parameters such as resolution and quantization have been analyzed [Mohapatra et al. 2005]. And energy efficient encoding has been proposed using power management techniques to increase the energy reduction with minimal QoS degradation [Mohapatra et al. 2003; Yuan and Nahrstedt 2004]. Further, error-resilient video encodings have been studied by controlling the error robustness such as intra-coding refreshness in an energy-efficient manner [Kim et al. 2006].

In this section, we summarize the previously proposed approaches with respect to QoS, energy, and error-resilience for mobile video applications as presented in Fig. 1. As outlined in the following subsections, whereas a great deal of work has been done in these areas, previously proposed approaches have overlooked the opportunities to actively exploit errors for the purpose of energy reduction with minimal quality loss. Our main contribution is to actively exploit errors to maximize the resource efficiency (energy efficiency) while ensuring the video quality. Specifically, we present a novel knob – *active error exploitation* – to extend the tradeoff space between the energy consumption and the video quality.

2.1 Energy/QoS-aware Video Encoding

With the growing popularity of video applications on battery-operated mobile handhelds, energy-efficiency is an essential feature that mobile video applications consider along with QoS. A standard video encoder in Fig. 1 shows the basic flow of video compression algorithms consisting of ME (Motion Estimation), DCT (Discrete Cosine Transform), Q (Quantization), and VLC (Variable Length Coding). First, the video image is separated into a certain size of data blocks (e.g., 8×8 macro blocks or MB), and each data block is processed through a motion estimation algorithm, which exploits the spatial-temporal correlations between video data. After ME, each data block is transformed by a discrete cosine transform into its frequency domain equivalent. Then each frequency component is quantized (divided by a quantization scale value) to reduce the amount of data to be transmitted. Finally, these quantized data are encoded using a variable length coding technique. At each

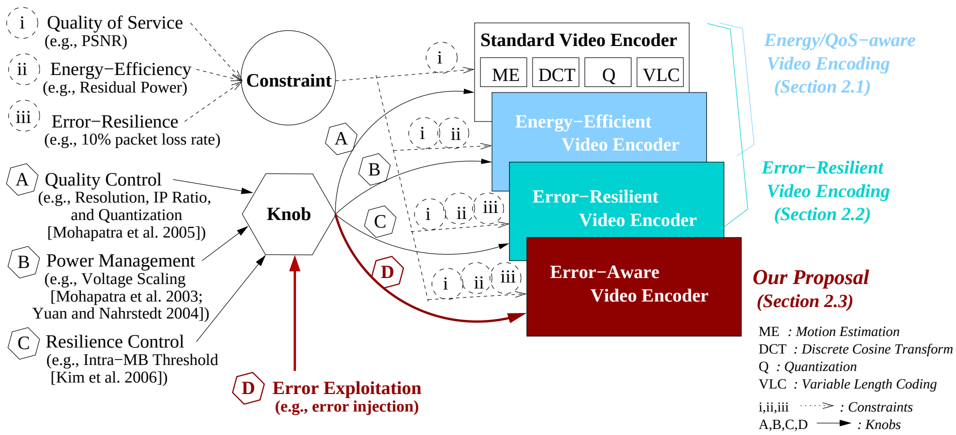


Fig. 1. Constraints and knobs considered by previous approaches and our proposal

compression process, several parameters need to be selected and each parameter affects the power and QoS. For example, full search and diamond search [Tourapis et al. 2000] are two candidates for ME, and they have tradeoffs between energy consumption for computation (diamond search is good since it searches for smaller area than full search), energy consumption for communication (full search is good since it can potentially find the reference data block with smaller difference than diamond search) and QoS (full search is good since it can deliver less difference potentially). Mohapatra et al. [Mohapatra et al. 2005] explored the effects of video encoding parameters such as quantization scale, IP-ratio, and motion estimation algorithms on energy consumption and QoS.

Energy and QoS aware adaptations have been studied for video applications on mobile handhelds in a cross-layer manner [Mohapatra et al. 2003; Yuan et al. 2003]. Mohapatra et al. [Mohapatra et al. 2003] proposed an integrated power management technique, which identifies interactive parameters among different system levels and tunes them to reduce the power consumption by middleware adaptations aware of system configurations. Similarly, Yuan et al. [Yuan et al. 2003] proposed a global cross-layer adaptation approach, which coordinates the CPU, operating system, and application to increase the energy efficiency. Yuan et al. also proposed a practical voltage scaling scheme to minimize the whole system energy of mobile devices while meeting the time constraints of multimedia applications. Eisenberg et al. [Eisenberg et al. 2002] considered the transmission power along with the video quality at the decoder. To limit the amount of distortion in the delivered video with minimal transmission energy, they exploited the knowledge of the concealment method at the decoder and the relationship between transmission power and the packet loss probability.

Related work in this area has mostly studied the tradeoff between energy consumption and QoS, but they did not take into account error resilience against unreliable transmission and they did not consider active error exploitation.

2.2 Error-Resilient Video Encoding

Video compression standards such as H.263 [ITU-T 1996] and MPEG [MPEG] increase the compression efficiency by exploiting the spatial and temporal correlations among consecutive frames with minimal quality loss. However, these compressed video data can be lost and eventually become error-inclusive at the decoding side through the unreliable channels due to congested routers, link failures, faded access points, etc. in wireless network. Thus, the effects of packet losses are propagated to the following frames due to the nature of spatial and temporal dependency in encoding techniques. To reduce these negative impacts on QoS, several techniques have been proposed and roughly classified into two groups: error-resilient techniques and error-concealment schemes [Cheng and Zarki 2004]. Typically, error-concealment techniques [Wang and Zhu 1998; Feamster and Balakrishnan 2002] are implemented at the decoder by recovering the lost data, and error-resilient techniques [Wang et al. 2000; Zhang et al. 2000; Worrall et al. 2001; Cheng and Zarki 2004; Kim et al. 2006] are designed at the encoder to increase the robustness against the transmission errors by adding redundancy.

One of the most effective methods for achieving error-resilient video is to introduce the intra-coded frame (I-frame) periodically since I-frames are decoded independently and protect the propagation of the transmission errors in previous frames. We call this video encoding technique as GOP-K (Group-Of-Picture), where K indicates the number of predictively-coded frames (P-frames) between I-frames. For instance, GOP-15 indicates a video encoding technique where one GOP consists of 1 I-frame and 15 P-frames. Recently, Yang et al. [Yang et al. 2007] reorganized the regular linear GOP structure to decrease the number of descendant frames using a double-binary tree structure and thus errors propagate to only a few frames. However, the transmission of I-frames causes delay and jitter due to their relatively large size compared to P-frames, and the loss of I-frames is more sensitive for QoS than P-frames [Cheng and Zarki 2004; Kim et al. 2006].

To mitigate both the propagation of the transmission errors and the overheads of large I-frames, intra-MB refresh approaches have been proposed [Worrall et al. 2001; Cheng and Zarki 2004; Kim et al. 2006]. Intra refresh techniques distribute intra-MBs among frames, and they not only remove the overheads of I-frames but also improve the error-resilience. Worrall et al. [Worrall et al. 2001] introduced the Adaptive Intra Refreshing (AIR), which updates the more important area of MBs more frequently. Cheng et al. [Cheng and Zarki 2004] allocated intra-MBs on a column-by-column basis in a progressive manner considering the residual error propagation, Progressive GOP (PGOP). While most intra-MB refresh techniques have been focused on alleviating the effects of the transmission errors on the video quality, Kim et al. [Kim et al. 2006] proposed an energy-efficient and error-resilient video encoding technique named PBPAIR, and presented tradeoffs among error resilience, encoding efficiency, and energy consumption for mobile handheld devices. Note that PBPAIR is not energy efficient in case of low packet loss rates since PBPAIR (as well as other intra refresh video encoding techniques) is designed to compress the video data as efficiently as a standard video encoding.

Most approaches above have focused on *passive error exploitation*, which means that errors are used for relaxing the constraint considering the feature of applications. On the contrary, *active* (or *aggressive*) *error exploitation* maximizes the feature of applications even by injecting errors intentionally, which to the best of our knowledge has not been applied to video encoding approaches.

2.3 Using Error-Awareness

While video encoding techniques did not consider error exploitation actively, system designers have considered error-awareness several ways. During system design – since error detection and correction schemes demand high overheads – they exploit the features of applications running on the system, and relax the error-correction requirements for the purpose of high yield rate and/or low energy consumption.

Kurdahi et al. [Kurdahi et al. 2007] proposed an error-aware design scheme for memory subsystems. They observed that strict 100% correctness is not required in some applications such as imaging, video, and wireless communications. They scaled down the voltage level aggressively to the point where the features of those applications can tolerate and let the memory system expose errors, and consequently achieve significant power savings due to the exponential relation between the supply voltage and the dynamic power dissipation.

At the network level, Harris et al. [Harris et al. 2005] exploited packet loss to increase energy-efficiency by discarding the subsequent packets, which compose a larger frame with the lost packet at the application layer (e.g., multimedia data) than a packet at the MAC (Media Access Control) layer. Previously, the frame-induced packet discarding mechanisms were applied to avoid the congestion collapse [Ramanathan et al. 1993], but even in the absence of congestion, they [Harris et al. 2005] aggressively used the framing-aware link layer mechanisms to reduce the energy consumption, which may be wasted by blindly processing each packet at the MAC layer from the transmission of unusable data at the end.

In general, the above approaches accept errors to their system design or network design; in contrast, our approach aggressively exploits the error tolerance of video data by introducing errors **intentionally**, and controls the error injection **adaptively** based on the feedback for the purpose of energy reduction with minimal quality loss for mobile video applications. By using errors actively to achieve the maximal energy gain while ensuring the QoS and resilience, our error-aware video encoding further opens opportunities to expand the tradeoff spaces as described in Fig. 1.

3. OUR APPROACH: ERROR-AWARE VIDEO ENCODING

In this section, we present the system model (Section 3.1), and fundamentals of our active error exploitation to expand the energy/QoS tradeoffs (Section 3.2).

3.1 System Model

Fig. 2 depicts our system model for mobile video conferencing applications. This mobile video conferencing system consists of two mobile devices (*Mobile 1* and *Mobile 2*) and the network environment (*Network*) between them as shown in Fig. 2. The Network consists of WAN (Wide Area Network) and two wireless access points, AP 1 and AP 2, each of which provides the wireless communication channel for each mobile device. Within the mobile devices, CPU and WNI (Wireless Network Interface) are two dominant contributors to power consumption [Mohapatra et al.

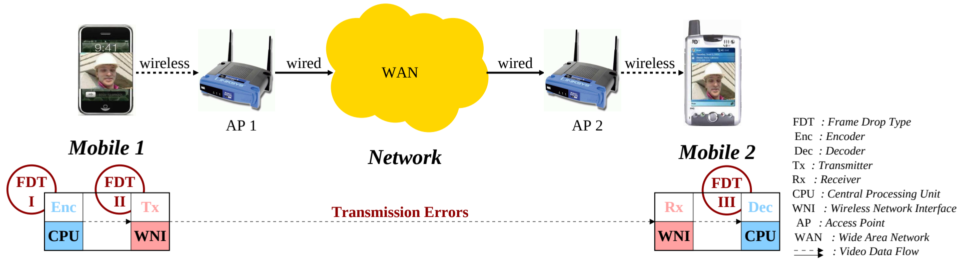


Fig. 2. System Model (Mobile Video Conferencing) and Frame Drop Types I/II/III for Active Error Exploitation

2003; Jiao and Hurson 2005; Guo et al. 2006]. Furthermore video processing and wireless communication are expensive in terms of power consumption. Thus to efficiently capture the energy consumption for computing and communication, each mobile device is modeled as a mobile station composed of CPU and WNI, where video data is encoded (or decoded) and transmitted (or received). Note that each mobile station of a video conferencing system is supposed to have both an encoder and a decoder. But for simplicity, this article considers one path from an encoder to a decoder. We analyze the quality of the delivered video at the decoding end, and study the energy consumption for each category such as the energy consumption for the encoding (*Enc EC*), transmission (*Tx EC*), the receiving (*Rx EC*), and the decoding (*Dec EC*) as summarized in Table I.

3.2 Fundamentals of Active Error Exploitation

Due to congestion, link failures, fading effects, etc., the transmission channel does not guarantee data delivery without packet losses and delays. Thus (as outlined in the previous section), error-resilient encoding techniques and error-concealment decoding schemes have been designed to combat transmission errors such as packet losses induced from an unreliable network.

In our active error exploitation approach, we can inject errors **intentionally** at any point in the encoding to decoding path of our system model (encoder, transmitter, and decoder as shown in Fig. 2); these intentional errors are presented as transmission errors and these errors are gracefully canceled by error resilient techniques.

The primary goal of ative error exploitation (through intentional error injection)

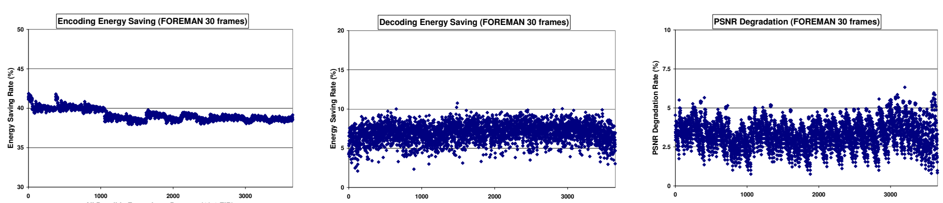
Table I. **Energy Consumption Category**

Type	Description
Enc EC	Energy consumed by CPU to encode a video stream
Tx EC	Energy consumed by WNI to transmit an encoded video stream
Source EC	Enc EC + Tx EC
Dec EC	Energy consumed by CPU to decode a received video stream
Rx EC	Energy consumed by WNI to receive a video stream
Destination EC	Dec EC + Rx EC

EC = Energy Consumption

is to achieve maximal energy reduction. For instance, the Decoder can drop the delivered video data to increase the energy reduction before the decoding process. Assume that the video encoder anticipates 10% packet losses in network and encodes the video data resilient against this 10% losses from the network (causing the increase of size in the compressed video data in general). But if the decoder receives all data without any losses, then it can intentionally drop 10% of the received data, saving the amount of energy which would be otherwise wasted for the decoding (*Frame Drop Type III* as in Fig. 2). Another example is the Transmitter dropping 10% of video data – saving the energy consumption for communication – with the error resilient video techniques taking care of the dropped data (*Frame Drop Type II*). Further, the Encoder can drop frames intentionally before the encoding process and encode only the rest of frames, making it robust against the dropped frames that will be considered as lost packets in the network (*Frame Drop Type I*). This intentional frame dropping scheme reduces the energy consumption by eliminating the encoding of the dropped frames. Note that the quality of service from intentional errors can be managed thanks to the features of error-resilient techniques and the inherent error-tolerance in video data. Of course, error-resilient video encoding techniques in general incur power consumption overheads for extra processing, and larger transmitted data size for the redundancy. Fortunately, there are video encoding techniques such as PBPAIR [Kim et al. 2006] that are not only error-resilient but also energy-efficient. Furthermore, the transmitted data size can be reduced by selectively dropping frames compared to the original error-resilient video encoders.

Note that dropping frames at the Encoder is most effective in terms of energy reduction since it affects the energy consumption across all the following components in an encoding-decoding path of Fig. 2, and the energy consumption for the encoding (Enc EC) is relatively high compared to those for the other components in our system model. Therefore, in this particular work, we only consider *Frame Drop Type I* for our active error-exploitation approach; *Types II* and *III* remain as our future work.



(a) Energy Reduction for Encoding (b) Energy Reduction for Decoding (c) Quality Degradation

Fig. 3. Energy Consumption Saving and Quality Degradation of Error-Aware Video Encoder (EA-PBPAIR) compared to a standard video encoder (GOP-15) at 10% EIR (Error Injection Rate)

To validate our idea of active error exploitation in video encodings, we performed a simple experiment by comparing our error-aware video encoder in terms of energy consumption and video quality to a standard video encoder. A standard video encoder in our study is defined as the GOP-15 video encoder based on H.263 with typical encoding parameters such that IP ratio is 15 (N_I:N_P = 1:15 where N_I and N_P denote the number of I frames and the number of P frames, respectively) [Wu et al. 2006; Yang et al. 2007], quantization scale is 10, and the resolution is QCIF (Quarter Common Intermediate Format: 176×144 pixels). We assume that the current network is error-free, i.e., 0% packet loss rate. We drop frames before the encoding at 10% error injection rate so that the error-resilient video encoder (PBPAIR) compresses the video data resilient against intentionally injected 10% errors (EIR = 10%), rather than against 0% packet loss rate (PLR). Thus, PBPAIR is configured with 10% PLR and 73% Intra_Threshold [Kim et al. 2006]. Since PBPAIR controls the error resilience at the cost of compression efficiency, it has a transmission overhead compared to a standard video encoder, which we will analyze in terms of energy consumption in our experiments. To observe the effects comprehensively, we explore all possible frame drop patterns when 3 frames are dropped intentionally out of 30 frames of a test video stream, *FOREMAN*, with its 10% EIR. Note that since the first frame is not dropped due to its critical effect on video quality, the actual number of all generated patterns is 3,654 (All possible combinations to choose 3 frames out of 29 frames: ${}_{29}C_3 = \frac{29!}{(29-3)!3!} = 3,654$). The simulation framework for this experiment will be presented in detail in Section 5. Fig. 3(a) and Fig. 3(b) show the effects of our active error exploitation on energy reduction at the Encoder and at the Decoder compared to energy consumption when a standard video encoder is applied. Fig. 3(c) plots the quality degradation measured in PSNR (Peak Signal-to-Noise Ratio), and mostly it is less than 5%. Fig. 3 shows that our active error exploitation can save energy consumptions by about 39% for the encoding, and by about 7% for decoding while QoS in PSNR degrades only by 3% on average, which demonstrates the promise of our approach.

4. EAVE: ERROR-AWARE VIDEO ENCODING

In order to extend the design space energy/Qos tradeoffs, we propose an error-aware video encoding (EAVE) technique. Video applications tolerate errors inherently. Further, error-resilient techniques make transmission errors negligible and error-concealment schemes (e.g., filtering and interpolating) decode the lost video data smoothly. Thus, errors induced intentionally (e.g., dropping frames) at the Encoder can be tolerated within an acceptable degree of QoS using error-tolerance, error-resilience, and error-concealment techniques.

In this section, we present the two-step architecture of EAVE (Section 4.1), introduce error control knobs and strategies for energy/QoS tradeoff extension (Section 4.2), evaluate several EAVEs based on previously proposed video encodings (Section 4.3), and present an adaptive EAVE (Section 4.4).

4.1 Two-Step EAVE

The error-aware video encoder is composed of two units, *error-injection unit* and *error-canceling unit*, as described in Fig. 4. *Error-injection unit* controls the amount of errors for the purpose of energy reduction, and *error-canceling unit* reduces the effects of the induced errors on the video quality using an error-resilient video encoder.

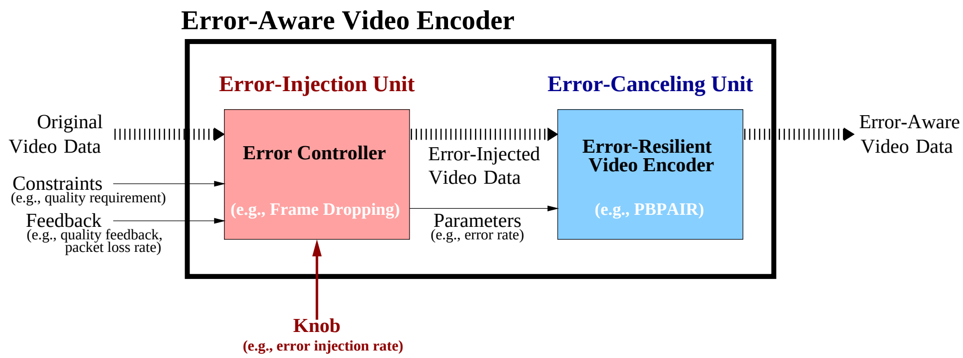


Fig. 4. Error-Aware Video Encoder (EAVE) is composed of Error-Injection Unit and Error-Canceling Unit.

Error–Injection Unit The *Error Controller* operates an *error-injection unit* to achieve tradeoffs between energy consumption and video quality using a newly introduced knob – error injection rate (EIR). The *Error Controller* accepts as input the constraint (e.g., required quality in PSNR) and the feedback information from the decoding side (e.g., reconstructed quality in PSNR) or from the network (e.g., packet loss rate) as illustrated in Fig. 4. The *Error Controller* preprocesses parameters (e.g., error rate) for the following video encoder in the *error-canceling unit*. For example, it sends the sum of packet loss rate and the error injection rate as an error-resilience parameter to the following error-resilient video encoder in the *error-canceling unit*. Note that "Frame Dropping" is one strategy for intentional error injection, and it will be detailed in Section 4.2.2.

Error–Canceling Unit The *Error-Resilient Video Encoder* in the *error-canceling unit* encodes the error-injected video data, rather than the original video data, from the *error-injection unit* as shown in Fig. 4. The *Error-Resilient Video Encoder* compresses the error-injected video data with error-resilience parameters (e.g., error rate = packet loss rate + error injection rate). Then *Error-Resilient Video Encoder* cancels the effects of injected errors on the video quality due to the error-resilient technique, and generates the error-aware video data, which is resilient not only against network errors but also against intentionally injected errors.

Our error-aware video encoders significantly extend the energy/QoS tradeoff space in several ways: (i) intentional error injection can tradeoff QoS for the encoding energy saving since it intentionally skips expensive video encoding for dropped frames, (ii) the energy consumption for video encoding can decrease since the error-resilient video encoder introduces more intra-MBs rather than inter-MBs due to the intentional frame drops while the energy consumption for video communication increases due to the increased size of compressed video data, and (iii) the error-resilient video encoder at *error-canceling unit* can adjust the resilience level of video data, which affects the energy consumption of video decoding and the delivered QoS.

4.2 Error Control Knobs and Strategies

The new feature of EAVE compared to the previously proposed video encoders is the *error-injection unit* with newly introduced knobs. Thus, the *Error Controller* mainly consists of the error injector and the parameter generator as shown in Fig. 5. We consider "frame dropping" as an error injection and "error rate" as a parameter in this work.

4.2.1 *Knobs.* EAVE introduces two knobs to extend energy/QoS tradeoff space: EIR (Error Injection Rate) and AER (Adjusted Error Rate).

EIR EIR indicates how many errors are intentionally injected at the *Error Controller*. By adjusting EIR, energy consumption and quality of service are traded off. A higher EIR increases the energy reduction for the encoding while decreasing quality of service (i.e., if it is beyond the point where error-resilient video encodings can manage the QoS). A lower EIR decreases the energy reduction for the encoding while decreasing the negative impact of intentional error injection on the video quality. EIR ranges from 0% to 100%: 0% EIR indicates no intentional error injection (i.e., EAVE is a conventional error-resilient video encoder), while N% EIR indicates N% of video data will be lost intentionally at the intentional error injection in Fig. 5. The EIR value is summed with other error factors (e.g., PLR and AER), and presented as a composite Error Rate (representing the desired level of error resilience) to the *Error-Resilient Video Encoder*, as shown in Fig. 5.

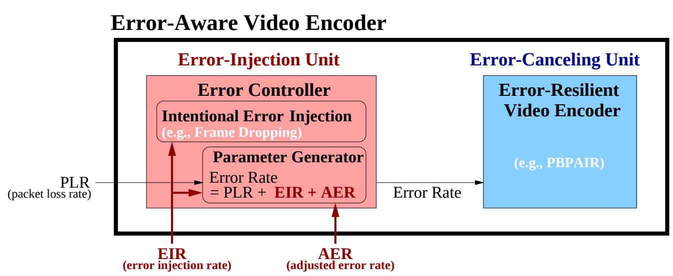


Fig. 5. EAVE introduces two knobs, EIR and AER, to Extend Energy/QoS Tradeoffs

AER EAVE also presents another knob, AER (Adjusted Error Rate), to tradeoff the energy consumption and the video quality. AER can be either negative or positive value in %, and the sum of PLR, EIR, and AER will be an error rate to the *Error-Resilient Video Encoder* as shown in Fig. 5. If we increase EIR, the degree of error-resilience increases and the size of error-aware compressed video data will increase, which causes high energy consumption overhead for the communication. To adjust this overhead, AER can reduce an error rate by setting a negative value in %, and reduce the compressed video data as output in the *Error-Resilient Video Encoder* at the cost of the video quality. For instance, if EIR and PLR are set to 20% and 10%, respectively, the error rate will be set to 20% when AER is set to -10%. This example is different from the case when an error rate is set to 30% with 20% EIR, 10% PLR, and 0% AER. Although they inject the same amount of errors (i.e., 20% EIR), the former (i.e., AER = -10%) encodes the video data with less resiliency, consumes more energy for the encoding, generates a smaller video output (due to the less intra-MB), consumes less energy for the communication, and degrades the video quality more than the latter (i.e., AER = 0%). Thus, a negative value of AER suppresses the increase of video output and reduces the transmission energy consumption while degrading the video quality. On the contrary, a positive value of AER can improve the video quality and the energy consumption for the encoding while increasing the transmission energy consumption. This AER is effective especially for conventional video encoders that do not implement a knob to control a finer degree of the error-resilience. Since these conventional video encoders are used in the *error-canceling unit*, we can use the AER to adjust the error rate for further extension of energy/QoS tradeoffs.

4.2.2 *Error Injection Strategies.* Recall that we achieve error injection through the dropping of frames. In this work, we consider two simple frame dropping approaches: PFD (Periodic Frame Dropping) and MDFD (Minimum Difference Frame Dropping).

PFD PFD periodically drops frames according to the error injection rate (EIR). For instance, PFD with 10% EIR drops every 10^{th} frame. PFD evenly distributes the effects of frame dropping on QoS over a video clip.

MDFD MDFD drops a frame if the difference in PSNR between the current frame and the previous frame is less than a threshold value. The intuition behind MDFD is that a smaller PSNR difference between frames indicates a smaller impact on QoS when the current frame is dropped. MDFD can keep dropping frames if consecutive frames have a smaller difference than the threshold value, which is very effective for energy reduction of video clips with low activity without significant loss of QoS. In this work, a threshold value and an EIR for MDFD will be selected based on the profiled results of video clips.

Note that our error-aware frame dropping strategies are different from traditional frame skipping. Traditional frame skipping techniques have studied the tradeoff between quality and bitrate [Song et al. 1999], and adapt the frame rate of video encoding to fit into the current network bandwidth while minimizing the quality degradation. The most effective strategy is to identify frames having high similarity with the reference frames and skip them to minimize the quality loss while satisfying bandwidth requirements. However, error-aware frame dropping in EAVE has a different strategy that does not need to consider the quality since the quality will be deliberately maintained by the nature of the error-resilient video encoding. Thus, error-aware frame dropping in EAVE can drop any frames within the guaranteed error rate that the original error-resilient video encoding can manage. For example, PFD with 10% EIR drops every 10^{th} frame after the first frame. Note that as shown in our experiments these simple frame dropping strategies are quite effective in conjunction with our active error-exploitation approach.

4.3 EAVE Evaluations

As shown in Fig. 5, our EAVE approach drops frames intentionally in the *error-injection unit*, and encodes video resiliently in the *error-canceling unit.* Thus, active error exploitation is orthogonal to any error-resilient and energy-efficient video encoding technique which adapts algorithmic parameters according to the network status such as packet loss rates. Thus, we study our active error exploitation for three error-resilient video encoding techniques: PBPAIR [Kim et al. 2006], PGOP [Cheng and Zarki 2004], and GOP-K in the following subsections. Accordingly, we evaluate *EA-PBPAIR*, *EA-PGOP*, and *EA-GOP* as error-aware video encodings in this work.

4.3.1 *EA-PBPAIR.* EA-PBPAIR uses PBPAIR [Kim et al. 2006] as the error-resilient video encoder. PBPAIR is an energy-efficient and error-resilient video encoder. PBPAIR has two parameters: the first parameter ($para_1 = Error\_Rate$) indicates the current network status (e.g., packet loss rate), and the second parameter ($para_2 = Intra\_Threshold$) represents the finer level of error resilience requested by designers.

EA-PBPAIR takes the sum of an EIR, an AER, and a current packet loss rate (PLR) in a network as $para_1$. For instance, the first parameter ($para_1$) is set to 15% when EIR is 10% and AER is 0% while PLR in a network is 5%. Note that the original PBPAIR would take 5% PLR as $para_1$. The $para_2$ is taken by original PBPAIR methodology. EA-PBPAIR is a PBPAIR with an intentional error injection. Thus, EA-PBPAIR is an energy-efficient, error-resilient video encoding technique with frames dropped intentionally before the encoding process to extend the energy/QoS tradeoff space for mobile video applications.

4.3.2 *EA-PGOP.* EA-PGOP uses PGOP (Progressive Group-Of-Picture) [Cheng and Zarki 2004] as the error-resilient video encoder. PGOP inserts a certain number of refresh columns per frame according to the network PLR. For example, PGOP introduces 3 refresh columns against 10% PLR. EA-PGOP takes the number of refresh columns for the sum of PLR, EIR, and AER by the PGOP own methodology [Cheng and Zarki 2004], and encodes the frame-dropped video data resilient not only against PLR in a network but also against the intentional error injection. Note that AER provides the finer level of error-resilience for the further extension of energy/QoS tradeoffs in EA-PGOP.

4.3.3 *EA-GOP.* EA-GOP uses GOP-K as the error-resilient video encoder. GOP-K inserts more I-frames if the network observes more packet losses. In this work, EA-GOP drops frames with an EIR, and encodes the video data with "K", the number of P-frames between two I-frames, based on the sum of PLR, EIR, and AER. To eliminate the impact of different sizes of compressed video data, "K" is selected for GOP-K to generate the similar size to that of PGOP. For example, GOP-3 is considered for 10% PLR since GOP-3 generates the compressed video data close to PGOP with the number of refresh columns 3, which is for 10% PLR [Cheng and Zarki 2004].

4.4 Adaptive EAVE

We now describe our Adaptive EAVE approach in more detail. Recall that we are intentionally injecting errors in the *error-injection unit*, labeled as "Error Controller" in Fig. 5. Fig. 6 describes the control flow of *Error Controller* in detail, and illustrates the video data flow and feedbacks from the Decoder and the network. For the purpose of illustration, we use EA-PBPAIR as EAVE in this section; similarly other error-aware video encodings (e.g., EA-PGOP, EA-GOP) can be used as the adaptive EAVE.

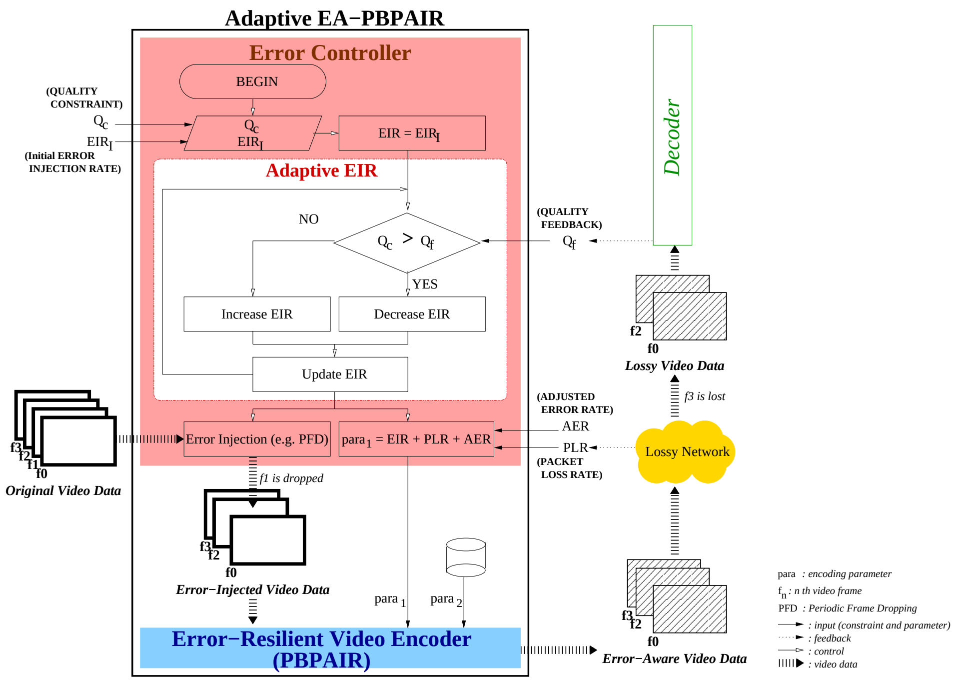


Fig. 6. Flowchart of Error Controller for an Adaptive EA-PBPAIR

One of simple approaches in EA-PBPAIR is to inject errors at a given EIR. Again, EIR is fixed and then the sum of EIR (from *Error Controller*) and the current packet loss rate (from the network) becomes an input $para_1$ for PBPAIR in the *error-canceling unit* with AER (Adjusted Error Rate) set to 0% as shown in Fig. 6.

To keep the loss of QoS minimal, our approach is able to constrain the EIR based on the feedback from the decoding side. Fig. 6 describes this adaptive EIR feature in *Error Controller* for adaptive EA-PBPAIR. *Error Controller* takes two initial constraints such as Q_c (Quality Constraint) and EIR_I (Initial Error Injection Rate). And then it receives the feedback information such as Q_f (Quality Feedback) from the decoding side and PLR (Packet Loss Rate) from the current network as shown in Fig. 6. If the feedback of the quality (Q_f) is less than the given requirement (Q_c), the current EIR is bad in terms of QoS and so *Error Controller* decreases the EIR, and it otherwise increases the EIR (the flow of Adaptive EIR in Fig. 6). Based on EIR, the error injection module inserts errors intentionally (e.g., by dropping frame periodically). Thus, *Error Controller* passes the error-injected video data instead of the original video data to *Error Resilient Video Encoder* as drawn in Fig. 6 (in this example, $f1$ is dropped). And $para_1$ is delivered as an input parameter to the following *Error Resilient Video Encoder* as an *error* *canceling unit*, which encodes the error-injected video data resiliently in preparation for the amount of errors indicated as $para_1$. Now, the encoded video data is error-aware, which is cognizant of injected errors as well as anticipated packet losses as illustrated in Fig. 6. This adaptive video encoder adjusts EIR to meet the given quality constraint while minimizing the energy-consumption. So our adaptive approach can be effectively used to adjust our video encoder under the dynamic network environment for maximal energy reduction while ensuring the given quality. Note that AER is set to the default value (0%) in most situations, and AER can adjust $para_1$ for further tuning of energy/QoS tradeoff extension in this example as presented in Section 4.2.1.

5. EXPERIMENTAL SETUP

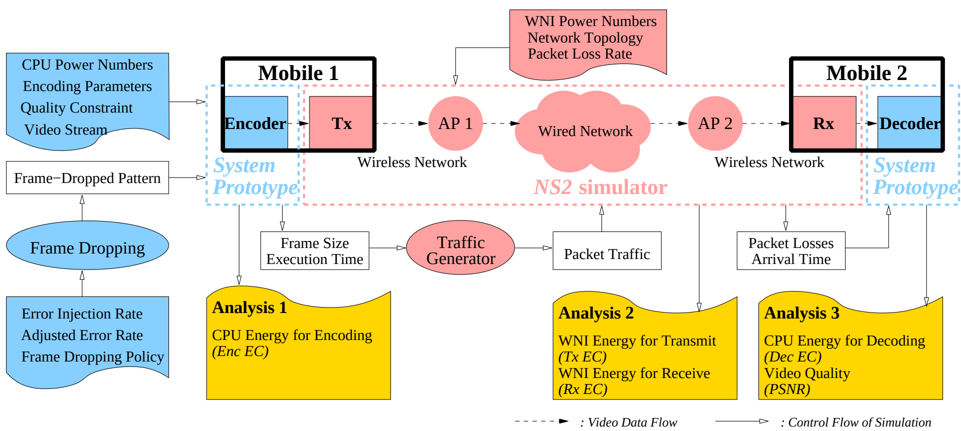


Fig. 7. Experimental Framework for Mobile Video Conferencing System - *System Prototype* + *NS2* Simulator

For interactive multimedia applications such as mobile video conferencing in distributed embedded systems, an end-to-end experimental system framework is a necessity since all components in a distributed system work interactively and affect other components in terms of energy consumption and performance. Thus, we evaluated EAVE on top of an end-to-end framework as shown in Fig. 7 consisting of a *System Prototype* [Lee et al. 2007] and *NS2* simulator [NS2] for mobile embedded system and network simulation. The *System Prototype* emulates a PDA and is detailed in our technical report [Lee et al. 2007].

The left side of Fig. 7 shows the preprocessing step, where a pattern of dropped frames is generated by a frame dropping policy according to an EIR and AER.

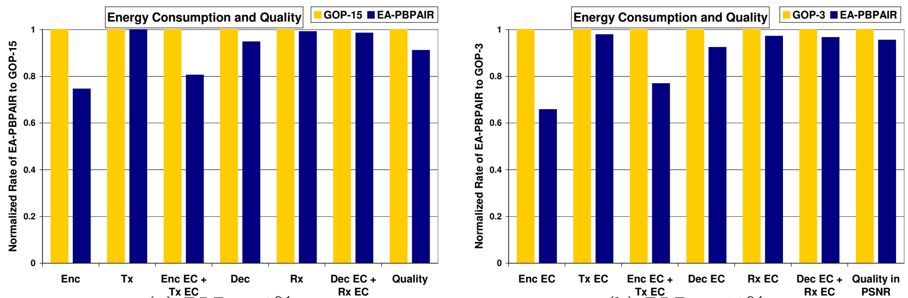
Table II. **Power Parameters and Transition Overhead**

	CPU			**WNI**			
Power Mode	Active	Idle	Sleep	Transmit	Receive	Idle	Sleep
Power (W)	0.411	0.121	0.001	1.425	0.925	0.80	0.045
Transition (msec)		1.00			0.75		

CPU power numbers, video encoder parameters, network status (PLR), and quality constraint are inputs to *System Prototype*, where a video encoder compresses a video stream. *System Prototype* analyzes the first set of results – Analysis 1 – such as the energy consumption for encoding (*Enc EC*), and calculates the encoded size and the encoding completion time of each video frame, which are used for generating the network traffic for the following network simulation. Analysis 1 succinctly shows the CPU energy for encoding at the sender. Next, *NS2* simulates the generated network traffic with a set of configurations including the network topology and WNI power values, and estimates the energy consumption (*Tx* and *Rx EC*) for WNIs – Analysis 2 – at Mobile 1 and Mobile 2 in our system model. Thus Analysis 2 captures the end-to-end networking effects, including those of the transmitter and the receiver. Finally at the receiver, the *System Prototype* decodes the transmitted video data based on generated packet losses and frame arrival times from *NS2*, and evaluates the energy consumption for decoding (*Dec EC*) and the video quality measured in PSNR (Peak Signal to Noise Ratio) in Analysis 3. Thus Analysis 3 captures the CPU energy for decoding at the receiver. Power consumption numbers for CPU [Intel Corporation] and WNI [Jiao and Hurson 2005] are configured as shown in Table II. By combining Analysis 1, Analysis 2 and Analysis 3, we are able to estimate the entire end-to-end energy savings for our proposed scheme. We now present further details of our experimental framework.

Using *NS2*, we simulate the network consisting of two IEEE 802.11 WLANs (Wireless Local Area Network) and a wired network connecting them as depicted in Fig. 7. Each WLAN is composed of one access point (AP 1 or AP 2), and one mobile device (Mobile 1 or Mobile 2). We exclude the effects of traffic from other mobile stations in this study since they affect the energy consumption of WNI in our mobile embedded systems. Instead, we limit the data rate of WNI, which constrains the encoded bit rate, and show clearly the effects of the varying data size generated by the Encoder. For wireless connection, we set the data rate to be 1 Mbps, considered to be an actual data rate [Guo et al. 2006; Meggers et al. 1996], and the link layer delay to be 25 μs. *NS2* generates packet losses for a given PLR. Each encoded video frame is composed of multiple packets if its size is larger than MTU (Maximum Transfer Unit), which is 1.5 KB in our simulation. A frame is considered lost if any packet of the frame is lost through the network simulation. For each scenario, we simulated more than 100 runs of *NS2* generated pseudo-random packet losses.

Recall that our EAVE approach combines an intentional frame dropping policy with an existing error-resilient video encoder (PBPAIR, PGOP, or GOP-K). PBPAIR takes two parameters, $para_1$ and $para_2$. We set $para_1$ (*Error_Rate*) as the sum of EIR, AER, and PLR. For comparison, $para_2$ (*Intra_Threshold*) is chosen for requested quality with the same compression efficiency as GOP-K (Group-Of-Picture with K) [Kim et al. 2006]. Similarly, parameters for GOP-K and PGOP are selected to configure themselves resilient against the sum of error rates. In this article, GOP-K based on H.263 [ITU-T 1996] is defined as a standard video encoder, where K indicates the number of P-frames between I-frames. In GOP-K, we change K for resilience against the transmission errors in network. For example, the number of refresh columns per frame for PGOP is selected as 3 according



(a) PLR = 0%   (b) PLR = 10%

Fig. 8. Energy Reduction and Quality Degradation of EA-PBPAIR compared to GOP-K (EIR = 10%, AER = 0%, FOREMAN 300 frames)

to [Cheng and Zarki 2003], and GOP-3 is selected as a baseline resilient for 10% PLR according to [Cheng and Zarki 2003; Kim et al. 2006].

As test video sequences, *AKIYO, FOREMAN,* and *COASTGUARD* in QCIF format (176×144 pixels) are used for our simulation study, and they are typical streams with low activity, medium activity, and high activity, respectively. Note that all video encoders generate a compressed stream at 5 fps (frames per second), which is the maximal frame rate [Lee et al. 2007] for a typical mobile handheld such as HP iPAQ h5555 [Hewlett Packard], and H.263 is designed for low data bandwidth [ITU-T 1996; Meggers et al. 1996] such as 64 kbps (kilobits per second). To constrain the bandwidth, we consider that the bitrate is 64 kbps and frame rate is 5 fps, which keeps the encoders from generating larger than 480 KB for 300 frames of test video sequences.

6. EXPERIMENTAL RESULTS

We present the efficacy of our approach through the following results: (i) the effectiveness of active error exploitation on energy reduction in EAVE (Section 6.1), (ii) the sensitivity of error injection rate and adjusted error rate in EAVE to energy reduction and QoS (Section 6.2), (iii) the expanded tradeoff space with EAVE over a set of video streams (Section 6.3), (iv) the energy reduction by MDFD in EAVE over a set of video streams (Section 6.4), and (v) the efficacy of the feedback-based adaptive EAVE (Section 6.5).

6.1 Energy Reduction from Active Error Exploitation

To show the effectiveness of our proposed technique, we present three sets of experiments with EA-PBPAIR, EA-PGOP, and EA-GOP, respectively.

Our first set of experiments evaluates EA-PBPAIR in comparison to GOP-15 with respect to energy consumption and quality. We use *FOREMAN* as a test sequence for this experiment. And this experiment considers that PLR is 0% in network to eliminate the impact of network losses, and GOP-15 is selected as a baseline encoder. Note that PBPAIR compresses the video data as efficiently as GOP-15 in case of the error-free network. EA-PBPAIR injects errors at 10% EIR with 0% of AER, and generates the similar size to that of GOP-15 to keep the transmission overhead close to GOP-15 by setting *parameters* for EA-PBPAIR.

Fig. 8(a) shows the effectiveness of an active error-exploiting approach in terms of the energy consumption for the source and the destination. This plots the normalized energy consumption for EA-PBPAIR to the energy consumption for GOP-15, and clearly shows that EA-PBPAIR is very effective in terms of each category of energy consumption, as compared to GOP-15. Especially, EA-PBPAIR consumes 25% less energy than GOP-15 with respect to the encoding since it drops 10% of frames and compresses more macro-blocks with less expensive intra encodings than predictive encodings. In terms of energy consumption of the transmitting video data at the source, EA-PBPAIR transmits similar amount of video data to GOP-15, and consumes the energy close to GOP-15. Thus, the energy consumption for the source, including the energy consumption for the encoding and for the transmission, is reduced by 20% with EA-PBPAIR at the cost of 9% quality degradation in PSNR. Note that 1% quality degradation indicates about 0.32 dB reduction from the PSNR value for GOP-15. At 0% PLR, we do not have enough margins of the transmission size to make up the quality loss from the intentional errors. So if a larger data size for transmission is allowed, EA-PBPAIR presents better quality with extra power overhead for transmission. At the destination side, EA-PBPAIR reduces the energy consumption by 5% for the decoding, which mainly results from dropping 10% frames at the source. Note that the more intra-encoded MB, the more energy consumption for the decoding but 10% frame dropping compensates for this effect. EA-PBPAIR consumes the energy for the receiving close to GOP-15. Thus, the energy consumption for the destination, including the energy consumption for the decoding and for the receiving, is reduced by 2% with EA-PBPAIR at the cost of 9% quality degradation in PSNR.

This result is very effective in terms of energy reduction at the cost of slight quality degradation, on the power-hungry mobile devices. For example, this active error-exploitation saves 25% energy for the encoding, which can be used to prolong the battery life time accordingly. Note that a passive error-exploitation video encoding such as a previously proposed error-resilient video encoding (PBPAIR) works similar to GOP-15 in case of an error-free network, which compresses video data as efficient as possible and incurs high energy consumption as compared to our proposal, an error-aware video encoding.

Similarly, we evaluate EA-PBPAIR with 10% EIR in comparison to GOP-3 for 10% PLR of network. Fig. 8(b) clearly shows that EA-PBPAIR reduces the energy consumption in all categories with only 4% quality loss. Energy saving for the encoding is about 34% by our proposed error-aware video encoding, EA-PBPAIR.

Throughout these experiments, the error injection before the encoding process in EA-PBPAIR can reduce the energy consumption for all categories, and it is a very effective method to tradeoff QoS for high energy reduction, which is not discovered with previously proposed video encodings.

The second set of experiments compares our EA-PGOP to PGOP in terms of energy consumption and video quality as shown in Fig. 9. For this experiment, PLR is considered at 10% and PGOP is configured with the number of refresh columns 3 resilient against 10% PLR [Cheng and Zarki 2004]. EA-PGOP is configured with the number of columns 4 against 10% PLR and 10% EIR. Note that the original

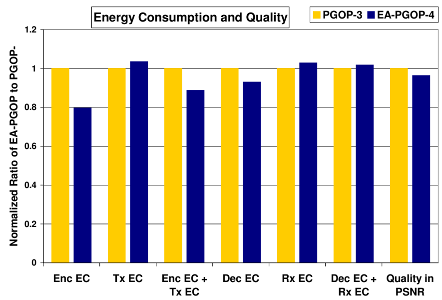


Fig. 9. Energy Reduction and Quality Degradation of EA-PGOP compared to PGOP (PLR = 10%, EIR = 10%, AER is set to configure EA-PGOP with the number of refresh columns 4, FOREMAN 300 frames)

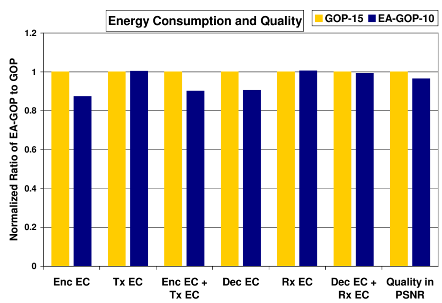


Fig. 10. Energy Reduction and Quality Degradation of EA-GOP compared to GOP (PLR = 0%, EIR = 20%, AER is set to configure EA-GOP-K with K = 10, FOREMAN 300 frames)

PGOP suggests 6 refresh columns per frame against 20% PLR but it substantially increases the portion of intra-MB and results in high energy consumption for the communication. Indeed, our preliminary experiments show that it incurs more than 20% energy overhead for both the transmission and the receiving. To minimize the energy overhead for the communication, we adjust AER (adjusting the error rate for error-resilient video encoding) and set the number of refresh columns as 4 rather than 6. This is the main reason why we present AER, which is able to increase the amount of errors we injected intentionally to save the energy consumption, especially for the communication energy. Fig. 9 shows that our EA-PGOP saves the energy consumption for the encoding by 20% and for the decoding by 7% while it incurs the energy consumption overhead for the transmission and for the receiving by about 3%, as compared to PGOP. Thus, the source energy consumption is reduced by 11% at the cost of the quality degradation by 4%, while the destination energy consumption is increased by only 2%.

This set of experiments demonstrates the effectiveness of active error exploitation in EA-PGOP in terms of the energy consumption for the encoding at the slight loss of video quality.

The third set of experiments evaluates the effectiveness of active error exploitation at a standard video encoding, GOP-K. We consider 0% PLR in the network and IP-ratio of GOP-K is set to 15 (K = 15). EA-GOP sets 20% EIR and it is configured to generate the similar amount of video data to GOP-15 by adjusting AER (about -15%). Thus, EA-GOP-10 is selected and the 20% amount of video frames are dropped with PFD. Fig. 10 shows that EA-GOP-10 saves the energy consumption for the source by about 10% and the energy consumption for the destination by about 1% at the cost of 4% video quality degradation. So this set of experiments demonstrates the effectiveness of active error exploitation with a standard video encoding such as GOP-K in terms of the energy consumption at the slight degradation of the QoS.

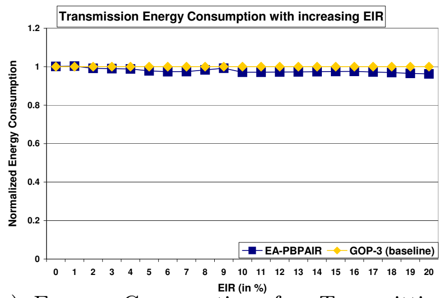
In summary, these experiments clearly demonstrate the effectiveness of our error-aware video encoding, EAVE, in terms of the energy consumption by trading off the video quality in several video encoding techniques.

6.2 Sensitivity of Error-Injection Rate and Adjusted Error Rate

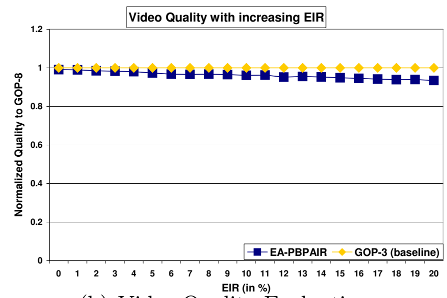
In this section, we present the experiments to show the sensitivity of EIR and AER in our error-controller algorithms as shown in Fig. 6.

EIR is an effective knob to increase the energy consumption at the slight cost of QoS. To observe the effects of EIR on the video quality and energy consumption, our first experiment compares EA-PBPAIR with GOP-3 by increasing EIR from 0% to 20% and keeping AER as 0%. For this experiment, we consider 10% PLR in network, on which *NS2* generates 10% packet losses. Since we adapt $para_2$ of PBPAIR to eliminate the transmission overhead as compared to GOP-3, Fig. 11(a) shows that the energy consumption for the data transmission of EA-PBPAIR with varying EIR is close to that of GOP-3. With an increase of EIR, still the video quality is managed within insignificant level of quality degradation as shown in Fig. 11(b). This is mainly because of the error-resilient feature of EA-PBPAIR. With 20% EIR, the loss of quality is about 7% in PSNR. However, Fig. 11(c) clearly shows that increasing the intentional EIR significantly saves the energy consumption for the encoding, and relatively small energy reduction is observed for the decoding as shown in Fig. 11(d). Since the portion of intra-MBs for each frame is increasing for increasing the error resilience, the energy consumption for the decoding is higher than GOP-3 with low EIR between 0% and 5%. However, with an increase of EIR, the number of frames to be decoded is decreasing (since we drop frames according to an increasing EIR) and thus the energy consumption decreases. With 20% EIR, we obtain 45% energy reduction for the encoding, and 17% for the decoding at the cost of 7% quality loss in PSNR.

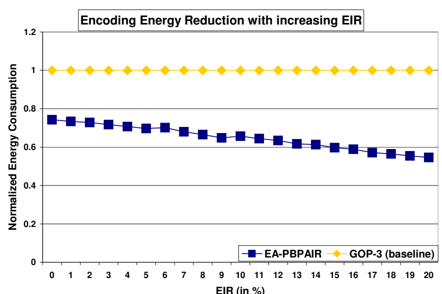
The next experiment compares EA-PBPAIR with intentional error-injection to the original PBPAIR in terms of the video quality and energy consumption. For this experiment, we fix all parameters except EIR and AER. We consider 5% PLR, and so 5% packets are lost in our network simulation. Note that EIR changes but the sum ($para_1$ as shown in Fig. 6) of EIR, PLR, and AER is fixed to 15% by adjusting AER for this experiment. For example, when EIR is set to 20% (PLR = 5%), AER is set to -10% to keep the sum of them as 15%. Fig. 12(a) clearly shows that EA-PBPAIR decreases the energy consumption at the source compared to PBPAIR (an error-resilient video encoding without intentional error-injection). With an increase of EIR, EA-PBPAIR saves more energy consumption while EA-PBPAIR degrades



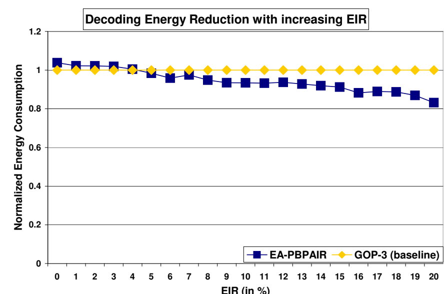
(a) Energy Consumption for Transmitting Data



(b) Video Quality Evaluation



(c) Encoding Energy Consumption



(d) Decoding Energy Consumption

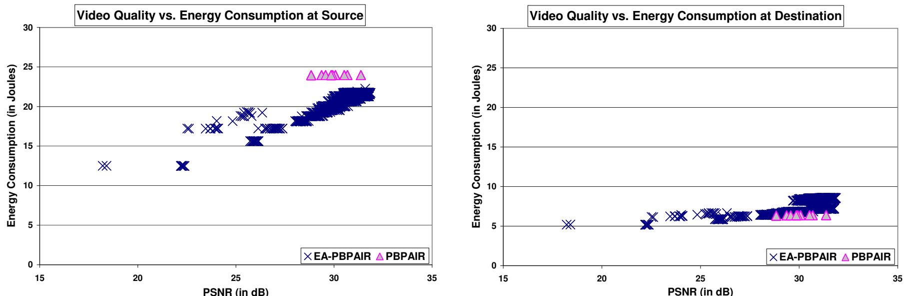
Fig. 11. Effects of Error Injection Rate on Energy Consumption and Video Quality (PLR = 10%, AER = 0%, FOREMAN 300 frames, Each encoding is constrained with bandwidth)

the quality of service since AER keeps the error rate for the resilience level that EA-PBPAIR can manage. The important observation we can make from Fig. 12(a) is that EA-PBPAIR discovers superior points with smaller energy consumption and better quality than PBPAIR. Fig. 12(b) shows the energy consumption vs. the quality at the destination. EA-PBPAIR consumes more energy than PBPAIR due to higher energy for the receiving since EA-PBPAIR encodes the video data robust not only against actual PLR but also against intentional EIR, which causes more intra-MB, and so increases the compressed data size. However, the energy reduction at the source is comparatively large enough to compensate for the slight energy increase at the destination. The most interesting observation we can make from Fig. 12(a) and Fig. 12(b) is that error-injection expands the operating points we can select to satisfy a given quality requirement and/or to meet a given power budget.

In summary, our proposed knob, EIR, with AER can effectively balance the energy consumption and the video quality.

6.3 Energy/QoS tradeoff

To achieve maximal energy reduction with minimal quality loss, we performed simulations by changing the intentional EIR, AER, and the error resilience ($para_2$). We consider 5% PLR in the network. We configure EA-PBPAIR to increase EIR



(a) Video Quality vs. Source Energy Consumption (Source EC)

(b) Video Quality vs. Destination Energy Consumption (Destination EC)

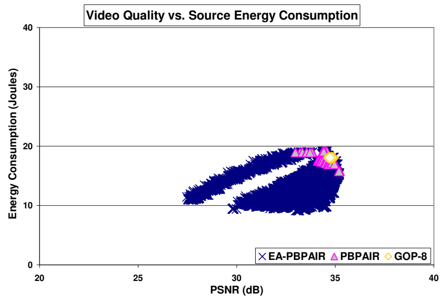
Fig. 12. Energy Consumption vs. Video Quality of EA-PBPAIR compared to PBPAIR (EIR = 1% to 50%, PLR = 5%, AER is set to $para_1 = 15\%$, FOREMAN 300 frames)

at the Encoder from 1% up to 50% in 1% increments, to adjust AER to keep $para_1$ as 15%, and to increase $para_2$ from 0% to 100% in 10% increments, and observe the effects on the energy consumption and video quality. Thus, applying these knobs generates interesting tradeoff space among quality, compression-efficiency, and energy-efficiency. Note that H.263 is designed for low data bandwidth [ITU-T 1996; Meggers et al. 1996] such as 64 kbps. To constrain the bandwidth (since the lower EIR and higher $para_2$ increase the encoded data output), we assume that the bitrate is 64 kbps and frame rate is 5 fps, which keep encoders from generating larger than about 480 KB for 300 frames of test video sequences.

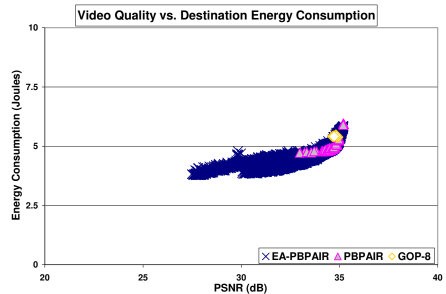
To demonstrate that EA-PBPAIR extends a constrained space of operating points to the larger space with quality and energy tradeoffs by exploiting errors actively, Fig. 13(c) and Fig. 13(d) draw the plots of energy consumption vs. video quality of EA-PBPAIR compared to PBPAIR and GOP-8 for 300 frames of a test bitstream, *FOREMAN*. They clearly show that EA-PBPAIR explores much larger points featured with perspectives of energy consumption and quality for the source and the destination. As compared to GOP-8, our EA-PBPAIR siginificantly expands the tradeoff space by about 15 times with respect to the energy consumption and by about 16 times with respect to the video quality as shown in Fig. 13(c). As compared to PBPAIR, our EA-PBPAIR extends the energy consumption space by two times, and the video quality space by about two times as shown in Fig. 13(c).

With an increase of EIR, we save additional energy consumption at the cost of QoS degradation. Interestingly, a video clip with lower activity such as *AKIYO* increase the energy reduction more effectively while minimizing the quality loss even at the destination as shown in Fig. 13(a) and Fig. 13(b). On the other hand, in high activity of bitstreams such as *COASTGUARD*, energy consumption is reduced but quality is distributed widely as shown in Fig. 13(e) and Fig. 13(f), since dropped frames can propagate errors dramatically due to high correlation among consecutive frames compared to *AKIYO* and *FOREMAN*.

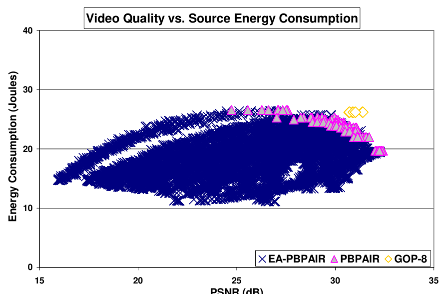
In summary, our EAVE can significantly expand the interesting tradeoff space for the energy consumption and the QoS for video applications running on battery-



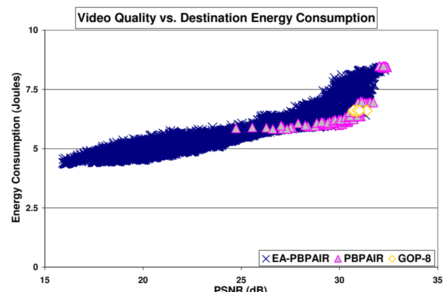
(a) Video quality vs. Source Energy Consumption (AKIYO)



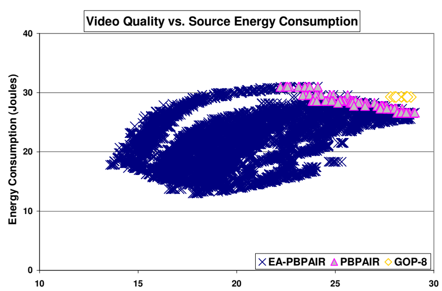
(b) Video quality vs. Destination Energy Consumption (AKIYO)



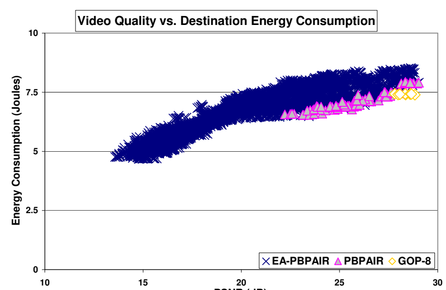
(c) Video quality vs. Source Energy Consumption (FOREMAN)



(d) Video quality vs. Destination Energy Consumption (FOREMAN)



(e) Video quality vs. Source Energy Consumption (COASTGUARD)



(f) Video quality vs. Destination Energy Consumption (COASTGUARD)

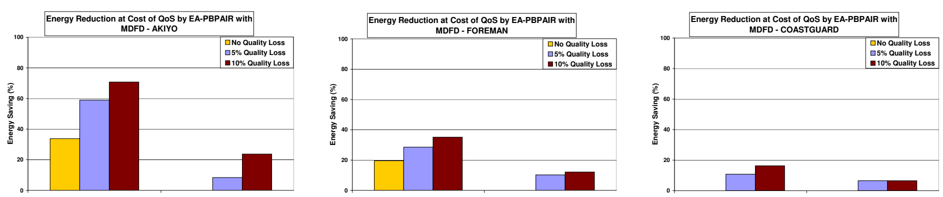
Fig. 13. Video Quality vs. Energy Consumption (Each encoding is constrained with bandwidth, EIR = 1% to 50%, PLR = 5%, AER is set to $para_1 = 15\%$, $para_2$ is varying, FOREMAN 300 frames)

limited mobile embedded systems, as compared to previously proposed video encodings. Our EA-PBPAIR shows 14 times extension in terms of the energy consumption and 13 times extension in terms of the video quality on average over test video streams, as compared to a conventional video encoding approach, GOP-8.

6.4 Effectiveness of MDFD in EAVE

Based on the profiled experiments from Fig. 13, EA-PBPAIR can save the energy consumption of the source by up to 37%, while the small amount of energy saving is observed (3%) at the destination without degrading quality. EA-PBPAIR reduces the energy consumption at the source including CPU power for encoding and WNI power for transmitting by up to 49%, and at the destination by up to 11%, at the cost of 10% loss in quality. Note that these are possible reductions with the best case of periodic frame dropping in EA-PBPAIR.

Fig. 14 plots how much energy can be reduced by EA-PBPAIR in conjunction with MDFD (Minimal Difference Frame Dropping) over a set of video clips with different levels of activity. Fig. 14(b) shows that EA-PBPAIR with MDFD can save the energy consumption of the source by about 20% without any degradation of the video quality. However, we do not observe any energy reduction at the destination without any quality loss. When we increase the threshold value of the minimum difference in MDFD (causing more frame drops), the source energy consumption can be reduced by 35% and the destination energy consumption can be reduced by 12% at the cost of 10% quality degradation.



(a) AKIYO (b) FOREMAN (c) COASTGUARD

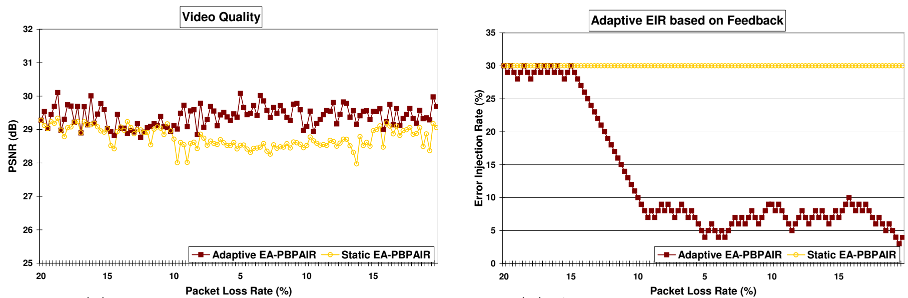
Fig. 14. Energy Saving at the Cost of Video Quality by EA-PBPAIR with MDFD (Minimal Difference Frame Dropping)

Fig. 14(a) and Fig. 14(c) present that dropping frames in bitstreams with high activity should be implemented wisely to obtain the high energy reduction while ensuring the quality requirement. While EA-PBPAIR achieves the saving by up to about 33% of the energy consumption at the source without losing QoS for *AKIYO*, no energy reduction is observed for *COASTGUARD*. These results are because *AKIYO* has low difference, i.e., high correlation, between frames, and thus it helps EA-PBPAIR be able to drop frames as many as possible within quality constraints. The difference between frames in high activity video streams such as *COASTGUARD* is comparatively high, and small threshold values cannot drop frames under no quality degradation.

In summary, our EAVE with MDFD (e.g., EA-PBPAIR with MDFD) can effectively reduce energy consumption at the cost of video quality.

6.5 Adaptive EA-PBPAIR under Dynamic Network Status

To show the effectiveness of our adaptive EA-PBPAIR by adapting EIR, we consider a scenario of dynamic network and compare adaptive EA-PBPAIR to static EA-PBPAIR with a fixed EIR. For this experiment, PLR begins with 20% and decreases



(a) Delivered Video Quality (b) Adaptive Error Injection Rate

Fig. 15. Adaptive EA-PBPAIR Robust to Varying PLR under Dynamic Network Status

by 5% every 20 runs and after 5% PLR it increases by 5% until it reaches 15%, and AER is set to the default value (0%). Horizontal axes in Fig. 15 represent this PLR scenario. Static EA-PBPAIR encodes the video data with fixed EIR = 30% for a given PLR. The quality constraint is 29.3 dB in PSNR, which is about 10% quality degradation from GOP-8 without any errors and losses. Adaptive EA-PBPAIR decreases the EIR by 1% if the feedback of the quality is less than the requirement. Otherwise, it increases the EIR by 1% to save the energy consumption as described at Adaptive EIR in Fig. 6. Fig. 15(a) draws the PSNR values for adaptive EA-PBPAIR in comparison to static EA-PBPAIR with fixed 30% of EIR, and shows that the delivered quality of adaptive EA-PBPAIR is consistently better than that of EA-PBPAIR with the fixed EIR. The important observation we can make from Fig. 15(a) is that adaptive EA-PBPAIR can adjust EIR dynamically to keep the quality considering the minimal energy consumption. EA-PBPAIR adapts the EIR according to the feedback with respect to the quality as shown in Fig. 15(b).

In summary, this EIR adaptive technique with EA-PBPAIR adjusts the quality of service based on the feedback while minimizing the energy consumption under dynamic network status.

7. SUMMARY

Energy reduction is challenging for video conferencing applications on battery-constrained mobile devices due to high processing power for compression algorithms and transmission of a large volume of video data. Since wireless networks are prone to errors and losses, energy-efficient and error-resilient video encoding techniques have been investigated intensively. It is interesting that video applications can tolerate errors inherently, and further error-resilient and error-concealment techniques can reduce the effects of losses induced from unreliable networks on the delivered quality.

We propose EAVE, a new video encoding technique which actively exploits these error-awareness such as error-tolerance, error-resilience, and error-concealment for maximal energy reduction with acceptable quality degradation at the cost of compression-efficiency. We show that EAVE with intentional error injection (simple frame drops) and error-resilient video encoders (PBPAIR, PGOP, and GOP) can reduce significantly the energy consumption from an encoding-decoding path for video conferencing applications on resource-limited mobile embedded systems. Further, we propose an adaptive EAVE by controlling a newly proposed knob, error-injection rate, in order to keep the delivered quality satisfied according to the feedback under the dynamic network status. Our experimental results show that EAVE can significantly expand the tradeoff space by 14 times with respect to the energy consumption and by 13 times with respect to the video quality on average over test video streams, as compared to a conventional video encoding. This largely expanded design space allows multimedia system designers to explore different design points and tradeoffs considering power and QoS as primary metrics.

Our future work includes intelligent frame dropping techniques combined power management techniques using slack times from frame drops for further energy reduction with minimal quality degradation. We also plan to extend our active error exploitation approach to the system level combined with error-aware architecture and/or with error-aware network protocols for further optimization.

REFERENCES

Cheng, L. and Zarki, M. E. 2003. An adaptive error resilient video encoder. In *SPIE Visual Communication and Image Processing*.

Cheng, L. and Zarki, M. E. 2004. PGOP: An error resilient techniques for low bit rate and low latency video communications. In *PCS*.

Eisenberg, Y., Luna, C., Pappas, T., Berry, R., and Katsaggelos, A. 2002. Joint source coding and transmission power managment for energy efficient wireless video communications. *IEEE Trans. Circuits Syst. Video Technology*.

Feamster, N. and Balakrishnan, H. 2002. Packet loss recovery for streaming video. In *12th International Packet Video Workshop*.

Guo, L., Ding, X., Wang, H., Li, Q., Chen, S., and Zhang, X. 2006. Exploiting idle communication power to improve wireless network performance and energy efficiency. In *IEEE INFOCOM*.

Harris, A., Sengul, C., Kravets, R., and Ratanchandani, P. 2005. Energy-efficient multimedia communications in lossy multi-hop wireless networks. *IFIP Mobile and Wireless Communication Networks 162*, 461–472.

Hewlett Packard. *HP iPAQ h5555 Series - System Specifications*. Hewlett Packard, http://www.hp.com.

Intel Corporation. *Intel PXA255(R) Processor: Developer's Manual*. Intel Corporation, http://www.intel.com/design/pca/applicationsprocessors/manuals/278693.htm.

ITU-T. 1996. H.263 Draft: Video Coding for Low Bitrate Communication. Draft ITU-T recommendation H.263, ITU-T. May.

Jiao, Y. and Hurson, A. R. 2005. Adaptive power management for mobile agent-based information retrieval. In *IEEE AINA*.

Kim, M., Oh, H., Dutt, N., Nicolau, A., and Venkatasubramanian, N. 2006. PBPAIR: An energy-efficient error-resilient encoding using probability based power aware intra refresh. *ACM SIGMOBILE Mobile Computing and Communications Review*.

Kurdahi, F., Eltawil, A., Djahromi, A. K., Makhzan, M., and Cheng, S. 2007. Error-aware design. In *Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD)*.

Lee, K., Kim, M., Dutt, N., and Venkatasubramanian, N. 2007. *System Prototype: Mobile device emulator for flexible and fast simulations.* Tech Rep. (http://www.ics.uci.edu/~kyoungwl/systemprototype/).

Meggers, J., Bautz, G., and Park, A. S.-B. 1996. Providing video conferencing for the mobile user. In *IEEE Conference on Local Computer Networks*.

MOHAPATRA, S., CORNEA, R., DUTT, N., NICOLAU, A., AND VENKATASUBRAMANIAN, N. 2003. Integrated power management for video streaming to mobile handheld devices. In *ACM Multimedia*.

MOHAPATRA, S., CORNEA, R., OH, H., LEE, K., KIM, M., DUTT, N., GUPTA, R., NICOLAU, A., SHUKLA, S., AND VENKATASUBRAMANIAN, N. 2005. A Cross-Layer Approach for Power-Performance Optimization in Distributed Mobile Systems. In *Next Generation Software Program in conjunction with IPDPS*.

MPEG. http://www.cselt.it/mpeg/.

NS2. Network Simulation version 2, http://www.isi.edu/nsnam/ns/.

RAMANATHAN, S., RANGAN, P. V., AND VIN, H. M. 1993. Frame-induced packet discarding: An efficient strategy for video networking. In *Network and Operating System Support for Digital Audio and Video*. 173–184.

SONG, H., KIM, J., AND KUO, C.-C. J. 1999. Real-time encoding frame rate control for h.263+ video over the internet. *Signal Processing: Image Communication*.

TAYLOR, C. N., DEY, S., AND PANIGRAHI, D. 2001. Energy/latency/image quality tradeoffs in enabling mobile multimedia communication. In *Proc. of Software Radio: Technologies and Services*. Springer Verlag, 55–66.

TOURAPIS, A., SHEN, G., LIOU, M., AU, O., AND AHMAD, I. 2000. A new predictive diamond search algorithm for block based motion estimation. In *Proc. of Visual Comm. and Image Proc. (VCIP)*.

WANG, Y., WENGER, S., WEN, J., AND KATSAGGELOS, A. K. 2000. Review of error resilient coding techniques for real-time video communications. *IEEE Signal Processing Magazine*.

WANG, Y. AND ZHU, Q.-F. 1998. Error control and concealment for video communication: A review. In *Proceedings of The IEEE*.

WORRALL, S., SADKA, A., SWEENEY, P., AND KONDOZ, A. 2001. Motion adaptive error resilient encoding for mpeg-4. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*.

WU, H., CLAYPOOL, M., AND KINICKI, R. 2006. Guidelines for selecting practical MPEG group of pictures. In *Proceedings of the 24th IASTED international conference on Internet and multimedia systems and applications (IMSA)*. 61–66.

YANG, K.-C., HUANG, C. M., AND WANG, J.-S. 2007. Error resilient GOP structures on video streaming. *Journal of Visual Communication and Image Representation 18*, 151–161.

YUAN, W. AND NAHRSTEDT, K. 2004. Practical Voltage Scaling for Mobile Multimedia Devices. In *Proceedings of the 12th annual ACM international conference on Multimedia*. 924–931.

YUAN, W., NAHRSTEDT, K., ADVE, S. V., JONES, D. L., AND KRAVETS, R. H. 2003. Design and evaluation of a cross-layer adaptation framework for mobile multimedia systems. In *Proceedings of SPIE/ACM Multimedia Computing and Networking Conference (MMCN)*.

ZHANG, R., REGUNATHAN, S. L., AND ROSE, K. 2000. Video coding with optimal inter/intra-model switching for packet loss resilience. *IEEE Journal on Selected Areas in Communications 18,* 6 (June).