Energy-Aware MPEG-4 FGS Streaming

Kihwan Choi, Member IEEE, Kwanho Kim, Member IEEE, and Massoud Pedram, Fellow IEEE

Abstract *— In this paper, we propose an energy-aware MPEG-4 FGS video streaming system with client feedback. In this client-server system, the battery-powered mobile client sends its maximum decoding capability (i.e., its decoding aptitude) to the server in order to help the server determine the amount of data (in the form of enhancement layers on top of the base layer) per frame that it sends to the client, and thereby, set its data rate. On the client side, a dynamic voltage and frequency scaling technique is used to adjust the decoding aptitude of the client while meeting a constraint on the minimum achieved video quality. As a measure of energy efficiency of the video streamer, the notion of a normalized decoding load is introduced. It is shown that a video streaming system that maintains this normalized load at unity produces the optimum video quality with no energy waste. We implemented an MPEG-4 FGS video streaming system on an XScale-based test bed in which a server and a mobile client are wirelessly connected by a feedback channel. Based on actual current measurements in this testbed, we obtain 20% communication energy reduction at the client by making the MPEG-4 FGS streamer energy-aware.*

Index Terms — MPEG-4 FGS, video streaming, communication energy, computation energy.

I. INTRODUCTION

WITH the availability of mobile, communication and computing systems, we have seen an explosive demand for wireless multimedia, e.g., streaming video. This trend in turn poses two challenges: (1) establishing and maintaining a stable channel for real-time operation and (2) power-aware operation so as to increase the lifetime of the battery-powered mobile system while meeting a minimum quality of service (QoS) requirement. Furthermore, it is desirable to provide a mechanism for a graceful degradation in the video quality as the system trades off QoS for higher energy efficiency. *Fine Granularity Scalability* (FGS) coding technique [1], which was adopted as the standard in MPEG-4, indeed provides an effective mechanism for graceful video quality degradation based on its hierarchical layer structure, which consists of a base layer and one or more enhancement layers. Although extensive studies have been conducted on the hierarchical layer structure of MPEG-4 and its error resiliency under fluctuations in the channel bandwidth [2][3][4], energy efficiency in server-client system has received little attention. Indeed, this topic is the focus of the present paper, where we show how to use FGS feature of MPEG-4 standard to achieve high energy efficiency subject to meeting a minimum video quality constraint.

For video streaming application, there are two sources of energy consumption in a wireless mobile client: communication energy for transmitting and receiving data and computation energy for processing the received data at the client. These energy consumptions are proportional to the amount of data in a video frame.

Most of the previous work on wireless video streaming is concerned about how to achieve better QoS under a fluctuating channel bandwidth. There are also a few publications that consider energy-efficiency issues in this context. In [5] an approach to reduce power consumption in a video communication system was proposed, where the coded bits are allocated between the source and channel coders, according to wireless channel conditions and video quality requirements. Reference [6] introduced an approach to minimize the power consumption of a mobile transmitter due to source compression, channel coding and transmission subject to a fixed end-to-end source distortion. In these approaches, however, no variation in the *decoding aptitude* of a mobile client was considered.

The *decoding aptitude* of a mobile client is defined as the amount of data that its decoder can decode in a given deadline. It is frequently the case that the client can change its decoding aptitude by employing methods such as dynamic voltage and frequency scaling (DVFS) or dynamic power management (DPM) [7]. Many of the state-of-the-art processors that are designed for mobile application are equipped with DVFS for low-power operation [8].

When a mobile client is prepared to accept (or tolerate) a low video quality (a condition that may arise due to low levels of remaining energy in the client, lack of a need to receive high quality video due, etc.), then it should be able to communicate this looser performance requirement to the server, which will in turn reduce the amount of data that is sent for a given video frame. Consequently, a significant amount of energy can be saved in the mobile client who need not receive and decode additional packets sent by the server. An energy-efficient wireless video streaming system can thus be achieved by creating feedback from the mobile client to the server and by enabling the server to adjust the amount of information that it puts into the channel.

In this paper, we propose a client-feedback wireless video streaming system in which the decoding aptitude of the client is considered by the server when it determines the amount of data that is sent for a given frame. This then results in trading video quality for lower energy consumption in the client. Of course, the streaming system must also guarantee a minimum achieved video quality. The proposed technique is made possible by the FGS feature of the MPG-4 standard as will be described later. The proposed energy-saving technique has been implemented on XScale-based platform and has resulted in 20% reduction in communication energy at the client.

The remainder of this paper is organized as follows. Related works on energy consideration of streaming system and MPEG-4 FGS are described in Section II. In Section III, a client-feedback streaming system is presented. Details of the implementation, including both hardware and software, are described in Section IV. Experimental results and conclusion are given in Sections V and VI, respectively.

II. BACKGROUND

A. Fine Granularity Scalability (FGS)

To adapt to the time-varying channel capacity (which is due to dynamic channel condition change such as congestion or fading), scalable video coding schemes are used. Example techniques include SNR scalability, temporal scalability, and spatial scalability in MPEG-2 and MPEG-4. In these layered scalable coding technique, total encoded bit-streams consist of a base layer and several enhancement layers. The bit-rate of base layer is determined by the minimum channel bandwidth and is sufficient to ensure a minimum achievable video quality. The enhancement layer provides higher video quality when the channel has extra bandwidth for the transmission of this layer. The FGS video coding technique, which was adopted as the standard in MPEG-4, provides for a very smooth video quality variation compared to other scalable coding technique because any number of bits in the enhancement layers may be truncated according to the channel condition. In FGS, the bit-plane coding of the DCT coefficients difference between the original frame and the reconstructed frame using base layer only is used to generate an enhancement layer [1]. The maximum number of bit-planes in a frame is given by $log_2(|DCT_{coef}|_{max})+1$ where $DCTcoef_{max}$ is the maximum DCT coefficient difference in a frame. In wired (or wireless) video streaming with a server and a client, the data rate (in bits/sec), R_{send}, at the server is determined by the available bandwidth in the network (or channel) as follows:

$$R_{send} = R_b + R_e \tag{1}$$

where R_b denotes the base layer bit-rate and R_e denotes the enhancement layer bit-rate. R_b should be less than the minimum allowed bandwidth while R_e varies according to channel conditions. R_{send} is set to provide the optimum video quality by transferring the maximum amount of video data to the client subject to the existing channel conditions. The higher the bandwidth of the channel is, the better the quality of the video is.

Fig. 1 shows a typical FGS-based layer structure for frame sequence, IPPP. The gray boxes depict the total bits of the enhancement layer data that were transmitted for each frame whereas the white boxes show the portion of the enhancement layer data that was not transmitted due to channel conditions.

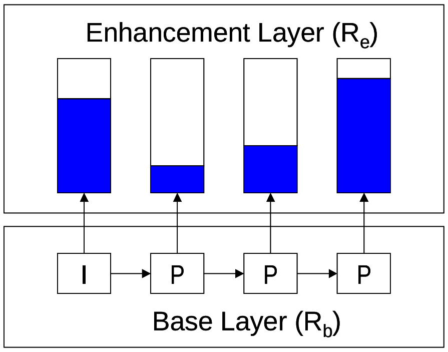


Fig. 1. FGS-based layer structure

B. Wireless Environments in Video Streaming

For energy-efficient video streaming in a wireless environment, two important factors should be considered: the energy consumption, which impacts the system lifetime and the channel bandwidth, which impacts the video quality.

1) Energy consumptions in wireless communication

Traditional low-power design techniques have mainly focused on minimizing the energy consumption due to charging/discharging of capacitive loads in a CMOS circuit that is performing computations or making decisions. As the battery-powered mobile devices became connected through a wireless network, the communication energy cost for transmitting and receiving data may dominate the computation energy cost. The video streaming application, which is one of the popular applications for mobile devices, results in rather large communication energy cost. To achieve a low energy video streaming system, more consideration must be given to optimizing the energy consumption due to the video data transmit and receive over the wireless channel. Fig. 2 shows a video streaming system in which a server and a mobile client are connected to one another by a wireless network.

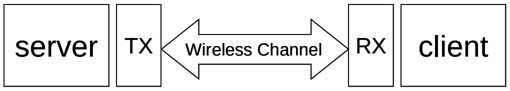


Fig. 2. Video streaming system with a server and a mobile client

The communication energy (E_{COMM}) is the sum of two energy dissipations: the transmit energy (E_{TX}) and the receive energy (E_{RX}), which are given by [9]:

$$E_{TX} = K_p \cdot (S \cdot \alpha_{TX} + \beta_{TX}), \qquad E_{RX} = K_p \cdot (S \cdot \alpha_{RX} + \beta_{RX}) \tag{2}$$

where K_p denotes the number of transferred packets, S is the packet size, and α_{TX}, β_{TX}, α_{RX}, and β_{RX} are regression coefficients for the transmit and receive energy dissipation costs, respectively. Notice that E_{TX} and E_{RX} in the above equations are proportional to the amount of data that is sent and received. E_{SERVER} and E_{CLIENT}, the sum of communication and computation energy (E_{COMP}) consumptions for server and mobile client during video streaming, can be written as:

$$\begin{aligned} E_{SERVER} &= E_{COMM\_SERVER} + E_{COMP\_SERVER} \\ &= K_p \cdot (S \cdot \alpha_{TX} + \beta_{TX}) + C_S \cdot V_S^2 \cdot f_{CPU\_S} \cdot T \end{aligned} \tag{3}$$

$$\begin{aligned} E_{CLIENT} &= E_{COMM\_CLIENT} + E_{COMP\_CLIENT} \\ &= K_p \cdot \left(S \cdot \alpha_{RX} + \beta_{RX} \right) + C_C \cdot V_C^2 \cdot f_{CPU\_C} \cdot T \end{aligned} \tag{4}$$

where C_S and C_C denote the switched capacitance per clock cycle time, V_S and V_C are the supply voltage level (assuming full swing transitions), f_{CPU_S} and f_{CPU_C} are the clock frequency of the server and client, and T is the total streaming time needed to transmit/receive K_p packets.

Let M denote the decoding aptitude of the client. M can be calculated as: $M = k{*}f_{CPU\_C}{*}D$, where k is a constant conversion factor and deadline D denotes the inverse of the target frame rate of the video. Assuming that f_{CPU_C} scales linearly with the operating voltage V_C, the computation energy consumption at the client is given as: $E_{COMP\_CLIENT} = \gamma{*}M^3$ where γ denotes a hardware-dependent constant. E_{CLIENT} is proportional to M in cubic way and a client can greatly reduce its energy consumption by decreasing its M value.

2) Throughput of wireless channel

In a wireless environment, the channel conditions vary widely over time and space in several ways due to factors such as noise, interference and multi-path fading. These factors can lead to errors in packets resulting in variation in throughput of a wireless channel. Commonly used methods for reliable packet transmission through a wireless channel are Automatic Retransmission reQuest (ARQ) and Forward Error Correction (FEC). ARQ basically means retransmission of corrupted packets where the expected number of transmissions for a packet is $1/(1\text{-}p)$ with p denoting the packet loss rate. In FEC, some redundant information about each packet is added by the encoder so that some of packets that incur errors may be recovered based on the redundant information.

In general ARQ is more suitable for uni-cast operation with low error rate whereas FEC is more effective in multi-cast operation with many nodes. By using these two methods, a certain degree of variation in channel throughput depending on error occurrence can be tolerated. Using ARQ, channel throughput variation due to packet error rate may be calculated based on IEEE 802.11b protocol that is widely used in wireless LAN [10]. Let t_{TX} denote the time to transmit a packet without error, then the expected time required for reliable transmission with packet error rate p is given by $t_{TX}/(1\text{-}p)$.

In video streaming application having a deadline D for a frame, an increase in the data communication time due to potential retransmissions causes a reduction in the channel throughput and finally leads to degradation of the video quality. Let B and p denote the maximum bandwidth and the average packet error rate during D, respectively. If the server sends K packets during D, the number of correctly arrived packets at the client, A, is given by:

$$A = \begin{cases} K, & \left(\dfrac{t_{TX}}{1-p} \right) \cdot K \le D \\ B \cdot (1-p), & otherwise \end{cases} \tag{5}$$

where $B = \dfrac{D}{t_{TX}}$.

III. A CLIENT-FEEDBACK VIDEO STREAMING

When designing an energy-aware video streaming system, communication and computation energy and the video quality must be considered. We examine closely a video streaming system, in which a server and a mobile client exist and the client can control its decoding aptitude M according to its energy management policy. The server employs no such policy because we assume that it is AC-powered and therefore, it has an inexhaustible energy source. The mobile client is battery-powered, and therefore, E_{CLIENT} is dependent on the energy management policy of the client.

It is usual that a battery-operated client which is equipped with a DVFS functionality has a DPM scheme to increase its service lifetime. The power manager employs some policy by which the operating frequency (and possibly the voltage level) of the client is changed. For example, if the power manager detects that the remaining battery charge is lower than a predetermined threshold level, then it may reduce the operating frequency of the client so that the client's lifetime can be extended at the cost of some degradation in service quality. As another example, consider the fact the client must perform other functions and services (perhaps due some other user-initiated computations, various I/O operations, some sensing, etc.) while the video streaming is taking place. In that case, the video quality may be lowered (by decaling that a reduced decoding aptitude for the client) so that other (more critical) tasks can be completed more quickly.

So, depending on the requirements imposed, a client can vary its M value. In a typical video streaming system between a server and a mobile client, both high video quality and low energy consumption on the client side are required. Now when the client changes its M value, the ratio between A and M becomes quite important. When A is larger than M, the communication energy for handling a total of $A\text{-}M$ packets is wasted. On the other hand, when A is smaller than M, the server can send more packets to improve the video quality if the channel conditions permit.

One more thing we want to notice is that we have not considered the effect of wasteful decoding of other packets not meant for the client, like in the case of spread spectrum communication. If such wasteful decoding is high, then it reduces the effective M value of the client with respect to the packets that are in fact sent to it by the video server. Note, however, that the cost of decoding a packet header to realize whether the packet is intended for the client or not is much lower than the cost of decoding the whole packet. Therefore, as long as the ratio of the unintended packets to useful packets is not very high, we can ignore the impact of these unintended packets on the decoding aptitude of the client.

Definition: *Normalized decoding load at time instance i is defined for the* client *as the ratio* A_i/M_i, *and is denoted by* N_i.

Notice that N_i captures the amount of wasted energy at time instance i. For an energy-efficient streaming system, N_i

should be set to one such that the required video quality by the client can be provided with no energy wasted handling useless data in the client. Let us consider energy consumptions for three different cases:

- $N_i > 1$; communication energy is wasted in the client while the video quality is limited by M_i
- $N_i = 1$; no energy is wasted; the video quality is determined by A_i or M_i
- $N_i < 1$; no energy is wasted; the video quality is limited by A_i.

In all three cases, the achievable video quality is given as $min(A_i, M_i)$ since the number of the decoded bits determines the video quality. With $N_i > 1$, the number of incoming packets from the server is higher than the decoding aptitude, resulting in communication energy wasting, E_{waste_i}, of $(A_i - M_i)*[1/(1-p_i)]*(S*\alpha_{RX}+\beta_{RX})$. In case of $N_i < 1$, there is no communication energy waste, but the video quality desired by the client cannot be achieved. There is, however, some slack in the decoding aptitude of the client, so the server can send more packets to the client in order to improve the video quality. With $N_i = 1$ the required video quality by the client can be achieved without any energy waste.

To estimate how much energy is wasted in case of $N_i > 1$, we calculated the wasted energy while changing the M_i/B ratio from 0.1 to 1 for different p values: 0.02, 0.06, 0.1, 0.13 corresponding to bit error rates (BER) of 1×10^{-5}, 3×10^{-5}, 5×10^{-5} and 7×10^{-5}, respectively. We used the regression coefficients for communication energy calculation from reference [9] at 11Mbps data rate and assumed that 30 packets with size of 256-byte are sent in 100msec.

Figure 3 shows how much energy is wasted for different M_i/B_i ratios and four different packet error rates, p, (so it is in fact a 3-D plot.) It thus conveys more information than a plot of energy waste versus A_i/M_i (which is only a 2-D plot). Note that A_i reduces the effect of M_i/B_i and p to a single term.

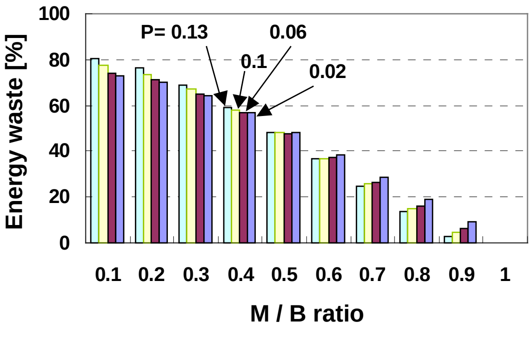


Fig. 3. Communication energy waste when $N_i > 1$

The mismatch between A_i and M_i usually arises from lack of information exchange between the server and the mobile client. To gain better video quality without wasting energy, a server ought to send the optimal amount of data in a deadline considering both the channel throughput and the decoding aptitude of the client. However, while channel throughput can be determined by using some network parameters such as the packet loss rate, BER, and transmission delay [12], it is difficult for a server to figure out the decoding aptitude of the client by itself. Clearly, if the server knows the client's decoding aptitude and the channel conditions, then the optimal data rate can be chosen without any energy waste. One way to achieve this goal is that the client periodically sends a small packet that contains information about its internal state (e.g., high performance or low power) to the server. In Fig. 4 a video streaming system with a feedback path from the mobile client to the server is shown. A status packet is periodically sent to the server at regular time intervals.

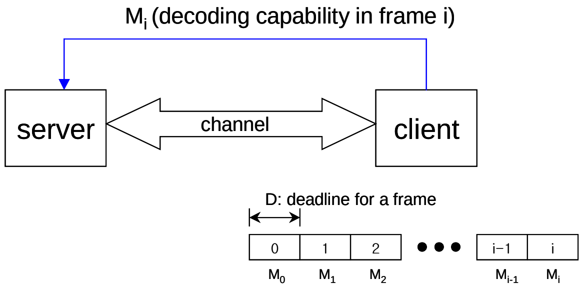


Fig. 4. Video streaming system with a feedback

If the client's decoding aptitude is changed during the i^th^ frame, the server should also change A_{i+1} to $min[M_i, B*(1-p_i)]$ such that energy waste can be avoided while enabling the client to process the data in the i+1st frame. A_{i+1} is set to M_i when there is enough bandwidth in the channel, but if M_i is larger than the current channel maximum bandwidth, then A_{i+1} is set to $B*(1-p_i)$. A_{i+1} calculated based on M_i has some error since M_i from the client cannot give exact values for the i+1st frame time. If M_i varies a lot, as a heuristic to reduce the average error, we can predict M_{avg} based on the history. Using an exponentially weighted average technique with a weighting factor α, A_{i+1} is calculated as follows:

$$A_{i+1} = \min\left[M_{avg}, B\cdot(1-p_{i+1})\right] \tag{6}$$

where $M_{avg} = \alpha\cdot\sum_{\tau=0}^{i}(1-\alpha)^{\tau}\cdot M_{i-1}$

In order to verify the effectiveness of our proposed streaming policy we performed a simulation. The wireless channel is modeled as a Gilbert-Elliot model [11], with two states, good and bad, which represents the channel conditions during data transmission. BER of good and bad state are set to $1e^{-5}$ and $1e^{-4}$, respectively. Packet size of 256 bytes and frame count of 500 are assumed. For the maximum bandwidth of the channel model, we assumed 11 Mbps IEEE 802.11b network protocol. For varying M, we generated a trace in which M is set to 0.8, 0.4, 1.2, 0.5, 0.7, and 1.4 of the maximum bandwidth, B. Fig. 5 shows variation in N for different M values over the 500 frames. From this figure, feedback from mobile client is essential to avoid redundant energy wasting by keeping N close to 1. N changes abruptly as M changes since M_{avg} is predicted using the weighted average method. If there is no severe fluctuation in M, calculating A_{i+1} using M_i may be better as explained before. Energy waste with M variation is summarized in Table.1. When $M < B$, significant amount of energy is wasted while there is no video quality improvement because M limits the video quality. Although these results can be different for different M values and channel conditions, they tell the story.

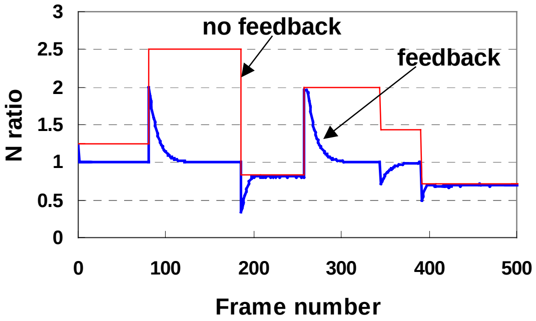


Fig. 5. ***N*** **ratio comparison with and without client feedback**

TABLE 1
ENERGY WASTE COMPARISON

M	Energy waste	
	No FB	FB
0.4B	57.35%	2.57%
0.5B	48.49%	6.27%
0.7B	28.69%	0%
0.8B	18.74%	0.21%
1.2B	0%	0%
1.4B	0%	0%

IV. IMPLEMENTATION

We implemented an MPEG-4 FGS streaming system on a high-performance test bed. The hardware used is the Intel's XScale processor which supports 9 different frequencies from 200MHz to 733MHz. A D/A converter was used as a variable operating voltage generator to control the reference input voltage to a DC-DC converter that supplies operating voltage to the CPU. Inputs to the D/A converter were generated using customized CPLD logic. When the CPU clock speed is changed, a minimum operating voltage level should be applied at each frequency to avoid a system crash due to increased gate delays. In our implementation, these minimum voltages are measured and stored in a table so that these values are automatically sent to the variable voltage generator when the clock speed changes. Voltage levels mapped to each frequency are distributed from 0.9V @333 MHz to 1.5V @733 MHz. For the software work, Microsoft reference MPEG-4 FGS encoder/decoder was modified to fit our purpose. Two generated bit-streams of QCIF video sequence with 150 frames, a base layer and a FGS enhancement layer with 5 bit-planes (bp0~bp4), are split into packets with size of 256-byte. Table II shows the average statistics for a frame of generated MPEG-4 layers.

In general, packet size is affected by many factors such as the packet fragmentation, header overhead, and channel conditions. For example, too small a packet size can cause a larger number of bytes to be transmitted, whereas too large a packet size is very susceptible to channel noise, causing either more frequent packet retransmissions (higher ARQ) or the necessity for more redundant channel coding (more FEC). In our application, different packet sizes may be assigned to different layers. For example, as shown in Table II, a small packet size can be used for the "base", "FGS header", and "bp0" layers whereas a larger one can be used for "bp1", etc. In our experiments, however, we chose to ignore this degree of flexibility, and used 256-byte packet size for all layers.

RTP/RTCP on UDP was used as a network protocol between server and client. We set our Linux machine as the AC-powered server and our test bed as the mobile battery-powered client, which is equipped with an IEEE 802.11b wireless LAN (WLAN) card. Energy consumptions of the WLAN interface during receiving packets were measured using our data acquisition (DAQ) system. The results are reported in Fig. 6.

TABLE II
STATISTICS OF GENERATED MPEG-4 LAYERS

	Size(bytes)	Packet number
base	76	1
FGS Header	9	1
bp0	18	1
bp1	278	2
bp2	1007	4
bp3	2022	8
bp4	3358	14

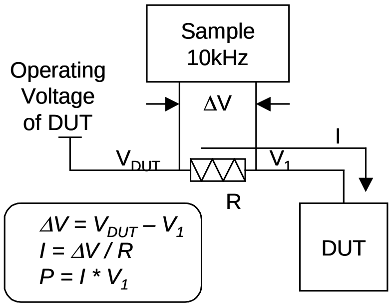


Fig. 6. Data acquisition system

V. EXPERIMENTAL RESULTS

We considered six different scenarios as follows:

(I) base only
(II) base + bp0
(III) base + bp0 + bp1
(IV) base + bp0 + bp1 + bp2
(V) base + bp0 + bp1 + bp2 + bp3
(VI) base + bp0 + bp1 + bp2 + bp3 + bp4

where bpX denotes bit-plane X. Each test case represents a different decoding aptitude for the client. In Fig. 7, the video qualities for each case were shown using average peak signal noise ratio (PSNR), which is a widely used estimate of the quality of a reconstructed image compared with the original image. The video quality increases as the number of decoded bit-planes increases. In theory, FGS provides continuous levels in image quality by truncating any number of bits in an enhancement layer, but a bit-plane is considered one level of image quality for practical purpose.

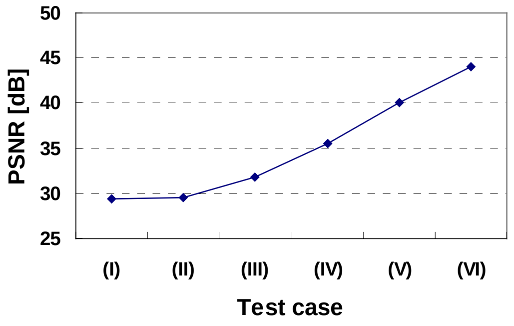


Fig. 7. Video quality with different bit-plane number decoded

Fig. 8 shows the current consumption drawn from 5V operating voltage of the WLAN when receiving packets for a frame. If more data are sent in a given deadline, more receiving energy is required. For example, with frame rate of 10, 125.2mJ and 133.4mJ for a frame were consumed for cases (V) and (VI), respectively.

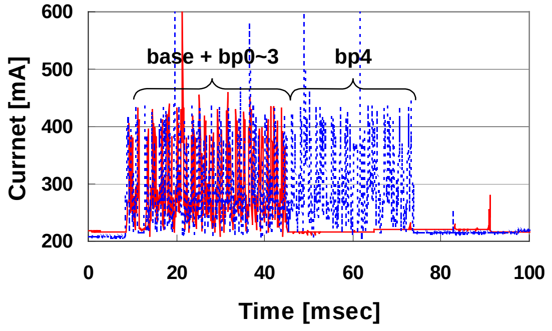


Fig. 8. Current consumption of WLAN interface in receiving packets

Fig. 9 shows the energy consumptions of the WLAN card and the CPU, which capture the bulk of the communication and computation energies in the client, respectively. We measured decoding aptitude variation depending on the CPU's operating frequency, but less than 5 fps rate was achieved even at the highest CPU frequency (733MHz). This is because the MPEG-4 decoder that we used is not optimized to run on the XScale hardware. So we measured the CPU energy consumption at 733MHz and normalized it to 10 fps rate. Fig. 10 shows energy waste in each test case.

To quantify the impact of the proposed technique on the battery lifetime, not the energy reduction, we need a realistic usage scenario for the client as well as a charge gauge circuit to report the remaining battery life. In addition we need to know how much of the video quality a client is willing to accept as other tasks arise or the remaining battery energy level drops. So, for the experimental results, it is assumed that the client accepts six different video qualities (from test case I to VI) and no other task is running except video streaming.

Let $VQ_{(i)}$ denote the video quality for case (i). $VQ_{(I)}$ is set by the client; in this case, for example, the wasted communication energy is 25.4mJ, which is about 40% of the CPU energy for $VQ_{(I)}$. Using our proposed method, about 20% of the communication energy reduction can be achieved.

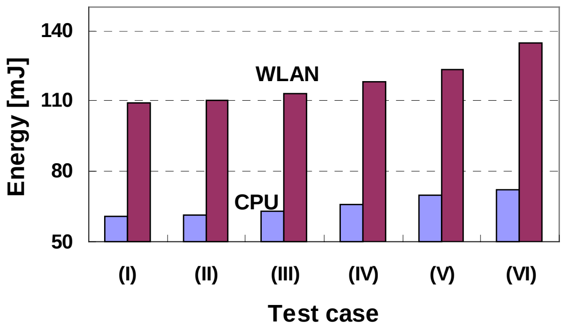


Fig. 9. Energy consumption of the client

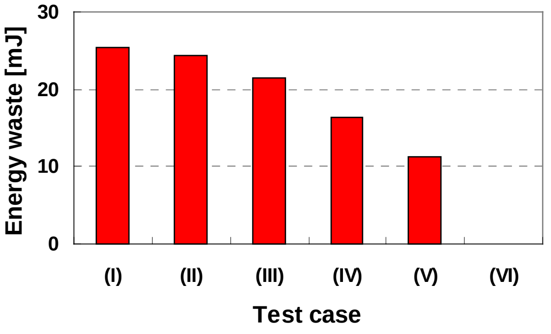


Fig. 10. Energy waste with different decoding aptitude

VI. CONCLUSION

A client feedback scheme is proposed to reduce redundant energy consumption in a wireless video streaming system consisting of a server and a client. The server does not send more data to the client when it cannot handle more than a given data rate, which is in turn determined by the client based on its own local energy management policy. From analysis and based on actual measurements under different channel conditions and decoding aptitudes for the mobile client, it was demonstrated that a video streaming system that maintains its normalized decoding load at unity produces the optimum video quality with no energy waste.

REFERENCES

[1] W. Li, "Overview of fine granularity scalability in MPEG-4 video standard," IEEE Trans. On Circuits and Systems for Video Technology, Vol.11, No. 3, March 2001

[2] M. van der Schaar, H. Radha, and C. Dufour, "Scalable MPEG-4 video coding with graceful packet-loss resilience

over bandwidth-varying networks," ICME 2000, vol.3, pp.1487-1490, 2000

[3] R. Cohen and H. Radha, "Streaming fine-grained scalable video over packet-based networks," GLOBECOM'00. IEEE, vol.1, pp.288-292, 2000

[4] R. Yan, F. Wu, S. Li, and R. Tao, "Error resilience methods for FGS video enhancement bitstream," The First IEEE Pacific-Rim Conference on Multimedia (IEEE-PCM 2000), Dec. 13-15, 2000 Sydney, Australia.

[5] Q. Zhang, Z. Ji, W. Zhu and Y. Q. Zhang, "Power minimized bit allocation for video communication over wireless channels," IEEE transactions on Circuits and Systems for video technology, vol. 6, pp. 398-410, jun. 2002

[6] X. Lu, E. Erkip, Y. Wang, D. Goodman, "Power efficient multimedia communication over wireless channels," IEEE Journal on Selected Areas in Communications, Vol. 21, No. 10, Dec. 2003

[7] T. Pering, T. Burd, and R. Brodersen, "The simulation and evaluation of dynamic voltage scaling algorithms," International Symposium on Low Power Electronics and Design, pp.76-81, 1998

[8] http://www.transmeta.com

[9] L. M. Feeney and M. Nilson, "Investigating the energy consumption of a wireless network interface in an ad hoc networking environment," IEEE INFOCOM 2001, vol.3, pp.1548-1557, 2001

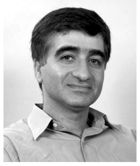
[10] IEEE, IEEE Std 802.11b – Wireless LAN medium access control (MAC) and physical layer (PHY) specifications: high speed physical layer (PHY) in the 2.4 GHz Band, 1999

[11] J. R. Yee and E. J. Weldon, "Evaluation of the performance of error correcting codes on a Gilbert channel," IEEE trans. Comm. vol.43, no.8, pp.2316-2323, 1995

[12] G. Wang, Q. Zhang, and Wenwu Zhu, "Channel-adaptive error protection for scalable video over channels with bit errors and packet erasures," GLOBECOM'01. IEEE, vol.3, pp.2045-2049, 2001

Kihwan Choi (M'01) received B.S. degree in Electronics from the Hanyang University, Korea in 1992 and M.S. degree in Electronics from the Pohang University of Science and Technology (POSTECH), Korea in 1994. He joined the Low Power Design research group led by Dr. Pedram at the University of Southern California in 2000. His area of research is low-energy system design.

Kwanho Kim received his M.S. degree in Computer Engineering from the Seoul National University, Korea in 1996. His current work focuses on embedded system design.



Massoud Pedram (S'88-M'90-SM'98-F'01) received a B.S. degree in Electrical Engineering from the California Institute of Technology in 1986 and M.S. and Ph.D. degrees in Electrical Engineering and Computer Sciences from the University of California, Berkeley in 1989 and 1991, respectively. He then joined the department of Electrical Engineering-Systems at the University of Southern California where he is currently a professor. Dr. Pedram has served on the technical program committee of a number of conferences, including the Design Automation Conference (DAC), Design and Test in Europe Conference (DATE), Asia-Pacific Design Automation Conference (ASP-DAC), and International Conference on Computer Aided Design (ICCAD). He served as the Technical Co-chair and General Co-chair of the International Symposium on Low Power Electronics and Design (SLPED) in 1996 and 1997, respectively. He is the Technical Program Chair of the 2002 International Symposium on Physical Design. Dr. Pedram has published three books, 50 journal papers, and more than 130 conference papers. His research has received a number of awards including two ICCD Best Paper Awards, a Distinguished Paper Citation from ICCAD, a DAC Best Paper Award. He is a recipient of the NSF's Young Investigator Award (a.k.a PECASE Award) (1996).

Dr. Pedram is a Fellow of the IEEE, a member of the Board of Governors for the IEEE Circuits and systems Society, and IEEE Solid State Circuits Society Distinguished Lecturer, a board member of the ACM Interest Group on Design Automation, and an associate editor of the IEEE Transactions on Computer Aided Design and the ACM Transactions on Design Automation of Electronic Systems.

His current work focuses on developing computer aided design methodologies and techniques for low power design, system-level dynamic power management, smart battery design and management, and integrated RT-level synthesis and physical design.