Energy-Aware Wireless Video Streaming

Ali Iranli, Kihwan Choi, and Massoud Pedram
Dept. of EE-Systems, Univ. of Southern California, Los Angeles, CA 90089
Email: {iranli, kihwanch, pedram}@usc.edu

Abstract

This paper presents a dynamic energy management policy for a wireless video streaming system, consisting of battery-powered client and server. The paper starts from the observation that the video quality in wireless streaming is a function of three factors: encoding aptitude of the server, decoding aptitude of the client, and the wireless channel. Based on this observation, the energy consumption of a wireless video streaming system is modeled and analyzed. Using the proposed model, the optimal energy assignment to each video frame is done such that the maximum system lifetime is achieved while satisfying a given minimum video quality requirement. Experimental results show that the proposed policy increases the system lifetime by 20%.

1 Introduction

With the availability of mobile, communication and computing systems, we have seen an explosive growth in wireless multimedia applications, e.g., streaming audio and 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 wireless system while meeting a minimum quality of service (QoS) requirement. Furthermore, it is desirable to provide a mechanism for graceful degradation in QoS so that a dynamic power manager (DPM) can incrementally trade off QoS for higher energy efficiency. Fine Granularity Scalability (FGS) coding technique [1], which was adopted as the standard in MPEG-4, 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 (optional) 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 a battery-powered server-client system has received little attention.

For video streaming application, there are two sources of energy consumption in wireless mobile hosts: computation energy for processing a video stream and communication energy for transmitting and receiving data. The computation energy of a server and a client is usually a strong function of the CPU frequency, which can be changed by employing methods such as dynamic voltage and frequency scaling (DVFS). The communication energy, on the other hand strongly affects the bit-error-rate (BER), and hence the video quality. There are detailed studies of the trade-off between energy consumption and BER in the communications field [5]. These studies can be divided into two main categories. The first category of techniques, which focus on the pass-band transceiver, exploits the fact that different modulation schemes result in different BER vs. signal-to-noise ratio (SNR) characteristics. The basic idea is that by adaptively changing the modulation and/or equalization, while keeping constant the received SNR at the receiver, one can achieve different BER. The second category of techniques, which focus on the base-band transceiver, studies the interaction between code performance and encoder/decoder design complexity. The main idea is to add a number of error controlling bits to the original data bits to protect them from channel changes. The key trade off is between the complexity of the encoding/decoding algorithms and the BER.

The achievable video quality in the streaming video systems is determined by following three factors: encoding capability of the server, decoding capability of the client, and the wireless channel error rate. It is well known that channel bandwidth fluctuation due to various factors result in the severe degradation in the video quality. This is due to the streaming nature of this real-time operation and the extra time, which is required for retransmissions if errors occur in the data packets.

The encoding (decoding) aptitude of the server (client) is defined as the amount of data that can be processed in a given deadline. This aptitude is proportional to the inverse of the video frame rate. When the server (or/and the client) changes its operating frequency and voltage to extend its lifetime the encoding (decoding) aptitude is also affected, so is the quality of the streaming video. This scenario is not unusual because many of the state-of-the-art processors that are designed for mobile application are equipped with DVFS for low-power operation [6]. In [7] a low energy MPEG-4 streaming policy using a client-feedback method was proposed where the client decoding capability at each time slot is sent to the server and the server adjusts its sending rate based on the feedback value from the client. By using this feedback approach, a significant amount of communication energy saving was achieved. However, the authors considered only energy consumption of the client side, not including that of the server. In [11], the authors proposed an energy-optimized image transmission system for indoor wireless applications, which exploits the variations in the image data and the wireless multi-path channel by employing dynamic algorithm transformations and joint source-channel coding. A detailed energy model for the client-server system was proposed and a global optimization problem solved by using feasible direction methods that resulted in an average of 60% energy saving for different channel conditions.

This research was supported in part by DARPA PAC/C program under contract DAAB07-02-C-P302 and by NSF under grant no. 9988441.

In this paper, we propose an adaptive policy for a wireless video streaming system in which the optimal energy assignment to each video frame considering both the server and client is employed such that the system consumes the minimum energy, while meeting the required video quality constraint. Hierarchical game theory is used to solve the corresponding mathematical optimization problem. Experimental results show an average of 20% increase in the overall system lifetime.

The remainder of this paper is organized as follows. Section 2 includes backgrounds on MPEG-4 FGS, model for energy consumption of the server and the client in the streaming system. Section 3 describes our energy assignment problem, and section 4 discusses the game theoretic formulation for this problem. Experimental results are described in Section 5 and it is followed by conclusion in Section 6.

2 Background

2.1 Fine Granularity Scalability (FGS)

To adapt to a time-varying channel capacity (which is in turn due to changes in the channel condition for example because of congestion or fading), a number of scalable video coding techniques have been proposed. Typical techniques include SNR scalability, temporal scalability, and spatial scalability in MPEG-2 and MPEG-4. In these layered scalable coding techniques, the total encoded bit-streams consist of a *base layer* and several *enhancement layers*. The bit-rate of the base layer is determined by the minimum channel bandwidth and is sufficient to ensure a minimum achievable video quality. The enhancement layers provide higher video quality when the channel has extra bandwidth for the transmission of extra layers.

The FGS video coding technique, which was adopted as the standard in MPEG-4, provides a very smooth variation in the video quality compared to other scalable coding technique because any number of bits in the enhancement layers may be truncated according to the channel condition. Therefore, the *Video quality* (*VQ*) can be represented as a linear equation in terms of number of bits transmitted:

$$VQ = k.R_{send} = k.(R_b + R_e) \tag{1}$$

where k is a regression coefficient, R_{send} is the total bit-rate (bits/sec), R_b is the base layer bit-rate, and R_e is the enhancement layer bit-rate. Note that R_b must be less than the minimum achievable bandwidth in the wireless channel, otherwise, no useful video transmission is possible and VQ goes to zero. R_e is varied in response to the channel conditions. R_{send} is thereby set to provide the minimum acceptable video quality by transferring the minimum amount of video data to the client subject to the existing channel conditions and the remaining battery lifetimes of the video server and/or the client. Recall that the higher the bandwidth of the channel is, the higher the VQ is for a fixed level of total energy consumption.

2.2 Energy Model of the Server

The energy consumption of the server for processing and transmitting a video frame may be written as:

$$E^S = E^S_{Comp} + E^S_{Comm} \tag{2}$$

where E^S_{Comp} and E^S_{Comm} denote the per-frame energy consumption costs of the computation and communication processes in the server. E^S_{Comp} and E^S_{Comm} are in turn calculated as follows:

$$\begin{aligned} E^S_{Comp} &= C^S_{eff} \cdot V_S^2 \cdot f^S \cdot T \\ E^S_{Comm} &= \left(P_{Enc} + P_{Mod} + P_{Amp}\right) \cdot T \end{aligned} \tag{3}$$

where C^S_{eff} denotes the effective switched capacitance per clock cycle time in the server, V_S is the supply voltage level (assuming full swing transitions) of the server CPU, f^S is the clock frequency of the server CPU, and T is the time duration of a frame (i.e., inverse of the frame rate). P_{Enc}, P_{Mod}, and P_{Amp} denote power consumptions of the corresponding blocks in the transmitter. The term representing the power consumption of the amplifier, P_{Amp}, is quite important. The other terms tend to be smaller in magnitude and depend linearly on the symbol rate with an additional constant. Hence, for our optimization purposes, the communication energy consumption of the server may be approximated as:

$$E^S_{Comm} = \left(P_{Tx} \cdot R_s + P_{const} + P_{Amp}\right) \cdot T \tag{4}$$

where P_{Tx} and P_{const} are the symbol-rate-dependent and constant power consumption components of the base-band transmitter. R_s denotes the symbol rate.

To characterize the bit error rate (BER) in terms of the power consumption of the transmitter, the relationship between the received signal-to-noise ratio (SNR) and the BER of the pass-band transceiver, i.e., the modulating/demodulating pair, can be used. For example, consider a Quadrature Amplitude Modulation (QAM) scheme where the BER is related to the received SNR by the following equation [5]:

$$\begin{aligned} BER &= 1 - (1 - P_{\sqrt{M}}) \\ P_{\sqrt{M}} &= 2.\left(1 - \frac{1}{\sqrt{M}}\right).Q\left(\sqrt{3 \cdot \frac{SNR_{rcvd}}{M-1}}\right) \end{aligned} \tag{5}$$

where M is the number of constellation points in the QAM modulation, typically $M = 2^b$ where b is the number of information bits represented by each constellation point. SNR_{rcvd} is the received signal-to-noise-ratio at the receiver. Let N_0, β, and R_s denote the noise spectral density, the spectral shaping factor, and the symbol rate, respectively. The received SNR is related to the transmit power level P_{Amp}, noise in the channel P_{Noise}, and the path loss parameter, σ, by [5]:

$$SNR_{rcvd} = \frac{P_{Amp}}{P_{Noise}} \cdot \sigma = \frac{P_{Amp}}{N_0 \cdot \beta \cdot R_s} \cdot \sigma \tag{6}$$

For a given BER and modulation scheme, i.e., for fixed b, one can calculate the required SNR, from equation (5), and then use equation (6) to find the minimum required transmit power level. The overall energy consumption of the transmitter for transmitting a single symbol is then calculated from equation (4).

2.3 Energy Model of the Client

The energy consumption of the client for receiving and processing a video frame may be written as:

$$E^C = E^C_{Comp} + E^C_{Comm} \tag{7}$$

where $E^C{}_{Comp}$ and $E^C{}_{Comm}$ denote the per-frame energy consumption costs of the computation and communication processes in the client. They are calculated as follows:

$$E^C_{Comp} = C^C_{eff} \cdot V_C^2 \cdot f^C \cdot T \tag{8}$$

where $C^C{}_{eff}$ denotes the effective switched capacitance per clock cycle time in the client, V_C is the supply voltage level (assuming full swing transitions) of the client CPU, and f^C is the clock frequency of the client CPU. $E^C{}_{Comm}$ is due to energy consumptions of the low noise amplifier, the demodulating block, and the channel decoding block and may be written as:

$$E^C_{Comm} = \left(P_{LNA} + P_{Demod} + P_{Dec}\right) \cdot T \tag{9}$$

where P_{LNA}, P_{Demod}, and P_{Dec} denote the power consumptions of the corresponding blocks in the receiver. Considering that all other blocks except the channel decoder are fixed and do not respond to changes in channel conditions, for optimization purposes, the client energy consumption may be approximated as:

$$E^C_{Comm} \cong \left(P_{Rx} \cdot R_s + P_{const} + P_{Dec}\right) \cdot T \tag{10}$$

where P_{Rx} and P_{const} are the symbol-rate-dependent and constant components of power consumption of the pass-band receiver.

Typically, a channel decoder is a multi-stage implementation of a recursive decoding function. Therefore, the accuracy of decoding is increased as the number of decoding stages (iterations) increases. On the other hand, increasing the number of stages would increase the power consumption of the decoder. In this work, a Viterbi decoder is studied as the channel decoder. In *adaptive Viterbi algorithms* (AVA), developed in [13]-[15], the decoding performance is increased by reducing the number of operations required to decode a single bit. This is achieved by reducing *Truncation Length* (*TL*) or by reducing the number of *Survivor Paths* (*SP*), i.e., those paths that are kept in order to find the optimum path. There are two main variations of the AVA. In the first variation, which is called the *T-Algorithm* [16], a fixed Threshold T, is chosen and then those paths that have path metrics equal to or less than T are included in the SP memory. In the second variation, called the *M-Algorithm* [16], a fixed number (M) of paths are kept and all other paths are discarded. These paths are selected by choosing the first M paths with the minimum path metric values.

Consider an adaptive Viterbi decoder with the functional block diagram depicted in Figure 1a. The decoder can be divided into three basic units. The input data (i.e., the noisy observation of the encoded information bits) is used in the Branch Metric Unit (BMU) to calculate the set of branch metrics $\lambda_{ji,k}$. These are then fed to the Add-Compare-Select Unit (ACSU) to update the path metric cost according to the following recursive equation:

$$\gamma_{i,k+1} = \min\left(\gamma_{j,k} + \lambda_{ji,k}, \gamma_{l,k} + \lambda_{li,k}\right) \tag{11}$$

where $\gamma_{i,k}$ is the path metric cost for state s_i in time step k, and $\lambda_{ji,k}$ is the branch metric cost between states s_i and s_j from time instances k and $k+1$, respectively (cf. Figure 1b). The Survivor Memory Unit (SMU) processes the decisions that are being made in the ACSU in order to carry out the ACS-recursion and outputs the estimated path, with a latency of at least *TL*.

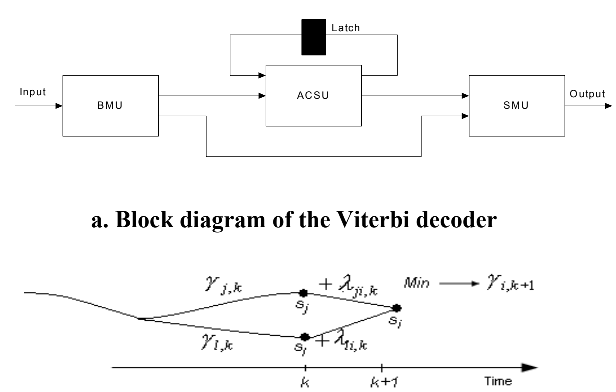
Power consumption for an adaptive Viterbi decoder may be macro-modeled by summing up the power consumption of each block times the number of paths that block is being used. This would result in following proposed power macro-model:

$$P_{Dec} = \left(P_{BMU} + 2^K.(P_{ACSU} + TL.P_{SMU})\right) \tag{12}$$

where P_{BMU}, P_{ACSU}, and P_{SMU} are the per-operation power consumptions of the corresponding modules and K represents the memory depth of the corresponding convolutional encoder. Notice that the ACSU module performs two additions and one comparison operation in each step (cf. equation 11).

3 Energy Optimization Problem

The encoding/decoding aptitude of an image processing core is a strong function of its operating frequency and voltage level. Thus, one can characterize the video quality *VQ* of frame i as:



a. Block diagram of the Viterbi decoder

b. Finding the optimum path

Figure 1: Adaptive Viterbi decoder

$$VQ_i = f(E_i^s, E_i^c, Ch_i) \tag{13}$$

where $E^S{}_i$ and $E^C{}_i$ denote the server and the client energy consumptions for frame i while ω_i denotes the wireless channel conditions for transmission of frame i.

We consider a wireless system operating over a fading channel. Time is assumed to be discrete. In each timeslot, which is equal to the inverse of the given frame rate, the channel state changes among different states chosen from finite set $C = \{C_1, C_2, \cdots, C_n\}$ according to a known probabilistic model [9]. The server and the client are assumed to be battery-powered, each with a fixed number of energy units available for use. Each channel state C_i determines the throughput that can be achieved per unit energy expended by the server/client.

The video encoding/decoding processing cores can operate with frequencies f_S and f_C in a range bounded by a lower bound f_{min} and an upper bound f_{max}. Let EL_{max} denote the maximum number of enhancement layers that can be processed by the server and the client while operating at f_{max}. In each timeslot i, one can calculate the Pareto-optimal curve for energy consumption of the server (E_i^S) and the client (E_i^C) in order to produce a given video quality (VQ) under a given channel condition, ω_i. More precisely, this curve denotes the trade-off between energy consumption of server and that of the client. Let's denote this Pareto-optimal curve by $g(E_i^S, E_i^C) = 0$.

Given the remaining energy levels of the server (E_0^S) and the client (E_0^C), the objective is to find a non-dominated energy allocation pair (E_i^S and E_i^C) for each timeslot i so as to maximize the overall system lifetime T_n, subject to:

$$\text{I)} \quad \sum_{i=1}^{T_n} E_i^S \le E_0^S \ \ and \ \ \sum_{i=1}^{T_n} E_i^C \le E_0^C \tag{14}$$

$$\text{II)} \quad avg(VQ_i) \ge VQ_{min}$$

$$\text{III)} \quad g\left(E_i^S, E_i^C\right) = 0 \quad \forall i$$

where $avg(VQ_i)$ denotes the average video quality over the system lifetime. Constraint I corresponds to the total energy bound, while constraint II guarantees the average video quality of the system. The third constraint ensures that Pareto-optimal points are chosen as the operating points in each timeslot.

In the following subsections, we first study the lifetime maximization under the condition that the channel state is known a priori and is constant. Next, we assume that channel states are random with a known probability distribution function $h(\omega_i)$ identically and independently distributed (i.i.d.) over time, and that ω_i is not revealed until just before the start of the timeslot. We will develop an approximate algorithm for this case, which produces a nearly optimal solution.

3.1 Known channel conditions

Let's start by examining the lifetime maximization problem in the simple case where the channel condition is a priori known and remains constant. In other words, ω_i is a known and fixed state for all timeslots. Although knowing the channel state for all times is an unrealistic assumption, the solution to this problem provides insight which is helpful for solving the more realistic problem scenario when the channel state is unknown and changing over time.

To solve this problem, one should first find the Pareto-optimal curve corresponding to the required video quality (VQ_{min}) and then find an operating point on that curve such that the lifetime of the system is maximized. Figure 2 shows typical behavior of function g in equation (14). Obviously, this function is non-increasing in terms of energy consumptions E_i^S and E_i^C.

The system lifetime is maximized when the server and the client run out of energy at the same time. To find an operating point at timeslot i, which guarantees the maximal system lifetime, one can set the ratio of energy consumption rates of the server and the client equal to the ratio of remaining energy levels of the server and the client. This solution is illustrated graphically in Figure 1. By drawing a line with slope equal to the ratio of remaining energies of the client and the server, and intersecting this line with the Pareto-optimal curve, we can find the optimal operating point and the corresponding energy consumptions of the server and client, E_{opt}^S and E_{opt}^C.

3.2 Unknown channel conditions

Let's examine the problem of lifetime maximization under the assumption that the actual channel condition ω_i is not known until just before transmission at time i. Moreover, let's consider a case where ω_i is i.i.d. with a known distribution function, $h(\omega_i)$. To solve this problem, we propose a dynamic policy, which attempts to minimize the difference between the remaining battery lifetimes of the server and the client during each timeslot.

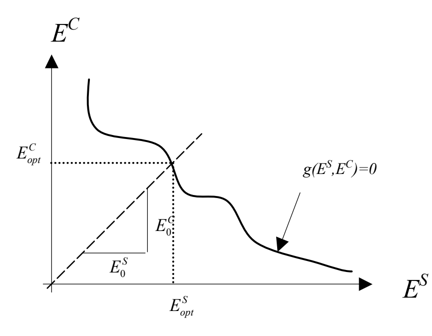


Figure 2: Pareto-optimal energy curve of the client-server pair

A simple way to calculate the estimated remaining battery lifetime of a mobile node (server or client) at the beginning of timeslot i is to divide the remaining energy level of the node by its energy depletion rate. A key challenge is to accurately estimate the expected depletion rate. Notice that simply using the energy depletion rate of the previous timeslot, *i-1*, is not suitable because it does not account for the long-term behavior of the node and may thus result in erroneous estimates. The approach we have taken is to calculate the (history-based) *aggregate energy depletion rate* of the node as a moving exponentially-weighted average so that the recent past has more influence, but the distant past is not completely ignored.

At the beginning of each timeslot i of duration T, the server chooses an energy consumption rate, E_{opt}^{S} / T , for itself based on the current estimates of the remaining battery lifetimes of itself and the client and the predicted channel state for timeslot i. In our approach, the channel state for timeslot i is taken to be the same as the channel state *i-1*. [1] From the chosen energy consumption rate, the server will then put into effect the respective encoding (computation) and transmit (communication) parameters by looking up these values from a pre-computed and locally-stored *policy parameter table*. Since the actual channel state and/or the remaining lifetime of the client may be different from the one predicted on the server side, the client will have to solve another optimization problem. This time the client knows the actual channel state and the adopted parameters of encoding and transmitting on the server side and has more accurate and up-to-date information about its own remaining lifetime; therefore, the client can optimize its energy consumption more effectively and thus obtain and enforce the reception and decoding parameters which are again looked up from the policy parameter table.

The aforementioned policy optimization problem, which involves a hierarchical variable determination process, is a form of multi-level optimization problems known as *Stackelberg game* [12] as detailed next.

4 A Game-theoretic Formulation for the Dynamically-varying Channel Condition

4.1 Background

In his monograph about market economy [17], H. V. Stackelberg used a hierarchical model to describe real market conditions. His model captured the scenario in which different decision makers attempt to make the best decisions in a market with respect to their own, generally different, utility functions. Generally speaking, these decision makers cannot determine their course of action independently of each other; rather, they are forced to act according to a certain hierarchy. Consider a simple case of such a problem where there are only two active decision makers. The hierarchy classifies these two decision makers into a leader, who acts independently of the market, and a follower, who has to act in a dependent manner. The leader is able to dictate the selling prices or to overstock the market with his products, but in making his decisions, he has to anticipate the possible reactions of the follower since his profit strongly depends not only on his own actions but also on the response of the follower. On the other hand, the choice of the leader influences the set of possible decisions as well as the objectives of the follower who in turn must react to the selections of the leader.

The aforementioned problem can mathematically be formulated as follows: Let X and Y denote the set of admissible strategies x and y of the follower and of the leader, respectively. Assume that the values of the choices are measured by the means of the functions $f_L(x,y)$ and $f_F(x,y)$, denoting the utility functions of the leader and follower, respectively. Then, with the knowledge of the selection y of the leader, the follower can select his best strategy $x(y)$ so that his utility function is minimized on X:

$$x(y) \in \Psi_L(y) = \underset{x}{Argmin}\left\{ f_F(x,y) \middle| x \in X \right\} \tag{15}$$

Being aware of this selection, the leader solves the Stackelberg game [17] for computing his best selection:

$$\underset{y}{Argmin}\left\{ f_L(x,y) \middle| y \in Y, x \in \Psi_L(y) \right\} \tag{16}$$

It is worth noting that the solutions to the Stackelberg game are different from the *Nash equilibrium points*, due to the special hierarchy that is imposed on the players. In Nash equilibrium solution all players have the same level of hierarchy and make decisions simultaneously, but in a Stackelberg game the decisions are made one after the other, following certain rules. In general, in an n-player Stackelberg game all players in same hierarchy level achieve the Nash's equilibrium point, but this is not true for players from different levels of hierarchy.

4.2 Application to Streaming Video

In our context, the follower and the leader become the client and the server, respectively. Strategy x for the client is the adoption of a specific vector of truncation lengths (*TL*'s) for the sub-carriers and an operating frequency for the decoder image processing core, f^C, and therefore, $X = \left\{ (TL_1, TL_2, \cdots, TL_n, f^C) \middle| \forall i : TL_i \in TLS, f^C \in FS \right\}$, where n is the number of sub-carriers in the Orthogonal Frequency Division Multiplexing (OFDM) signal, *TLS* denotes the set of all (feasible) *TL*'s for the adaptive Viterbi decoder, and *FS* is the set of feasible frequencies for the image processing core. Strategy y for the transmitter is a choice of specific overall

[1] Obviously, more elaborate channel estimation techniques may be employed to improve the selection process, but this simple channel prediction scheme serves our purpose of illustrating the general approach.

transmission power level and a set of modulation levels for the different sub-carriers and operating frequency for the encoder image processing core, and therefore, $Y = \left\{ (P_{Tx} / R_s, b_1, b_2, \cdots, b_n, f^S) \middle| \forall i : b_i \in MLS,\ P_{Tx} \in PLS,\ f^S \in FS \right\}$.

where *MLS* and *PLS* denotes the sets of (feasible) modulation levels for each sub-carrier and available power levels for signal transmission. These sets are known from chipset specification or the standard protocol supported by the chipset. Note that this formulation can easily be extended to take into account different transmit power levels for each sub carrier. This case is not explored here because it would require multiple output amplifiers (one per sub-carrier) in order to support independently controlled different power level per sub-carrier. This is quite expensive from implementation point of view.

Similar to the case of fixed and known channel conditions, the overall objective of the client-server game is to ensure that they achieve an acceptable level of performance while maximizing the overall video service time. Notice that the video service is terminated as soon as any one of the server or the client exhausts its energy source. The way this optimization problem is solved is that the server and the client take turn at the beginning and end of each time slot. The server's goal is to minimize the overall energy consumption of the client-server system whereas the client's objective is to make sure that it will not exhaust its energy source any sooner than the server does. In this way, this two-player game results in extending the overall system lifetime by first minimizing the energy consumption and then by ensuring that no one dies earlier than the other. Details are explained below.

In time slot *t*, the client (follower) uses the absolute value of the difference between its expected lifetimes and that of the server as the cost function, $f_F(x,y)$. Obviously, the client must meet maximum energy consumption and BER constraints under the received SNR value, which is in turn dictated by the choice of the server parameters, $\hat{Y}$. The client knows the expected lifetime of the server, L^S, as a result of the last data transmission. It must adopt client parameters (i.e., *TL* and f^C) so as to minimize $f_F(x,y)$. Notice that L^S was calculated by the server at timeslot *t-1* as the ratio of the server's remaining energy level to the server's aggregate energy depletion rate.

Since the OFDM symbols are transmitted at a constant rate, we can factor the power coefficients and drop the constant values of equations (10) and (12). The optimization problem in the client may then be stated as follows:

$$\arg\min_{\hat{X}} \left\{ \left| \frac{E_t^C}{L_{t-1}^S} - \left\langle C, \hat{X} \right\rangle \right| : A\hat{X} + B\hat{Y} \le R\hat{E}Q^C,\ \hat{X} \in TLS^n \times FS \right\} \quad (17)$$

where E_t^C and L_{t-1}^S denote the remaining energy value of the client at time *t,* and the expected lifetime of the server, which was received at time *t-1*. The ratio of E_t^C to L_{t-1}^S signifies the power dissipation target for the client in order for it to survive until the end of the server's expected lifetime. $\hat{X}$ refers to the client's strategy. *C* is the coefficient vector for computing the power consumption (i.e., per-frame energy consumption) of the client, P_t^C, from the *TL's* and the f^C value, i.e., $P_t^C = \left\langle C, \hat{X} \right\rangle$. Where $\langle a,b \rangle$ is used to represent the inner product of vectors *a* and *b*. The client's objective in this optimization step is to find $\hat{X}$ such that its actual power consumption becomes as close as possible to its target power dissipation.

The client must however, do this optimization under appropriate maximum energy consumption and BER constraints. These constraints are written in matrix-vector form as: $A\hat{X} + B\hat{Y} \le R\hat{E}Q^C$ where $\hat{Y}$ refers to the server strategy and *A* and *B* denote the coefficient matrices that account for the channel conditions and per-frame energy consumptions of the basic building blocks of the client. $R\hat{E}Q^C$ is a vector consisting of an upper bound on overall energy consumption, E_{max}, and the required BER values as shown below:

$$A = \overbrace{\begin{pmatrix} 2^k \cdot P_{SMU}/R_S & 2^k \cdot P_{SMU}/R_S & \cdots & 2^k \cdot P_{SMU}/R_S & P_f \\ \alpha_{BER_1} & 0 & \cdots & 0 & 0 \\ 0 & \alpha_{BER_2} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \alpha_{BER_n} & 0 \end{pmatrix}}^{(n+1)\times(n+1)} \quad C = \begin{bmatrix} 2^k \cdot P_{SMU}/R_S \\ 2^k \cdot P_{SMU}/R_S \\ \vdots \\ 2^k \cdot P_{SMU}/R_S \\ P_f \end{bmatrix} \quad (18)$$

$$B = \overbrace{\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & P_f \\ 0 & \beta_{BER_1} & 0 & \cdots & 0 & 0 \\ 0 & 0 & \beta_{BER_2} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \beta_{BER_n} & 0 \end{pmatrix}}^{(n+1)\times(n+2)} \quad R\hat{E}Q^C = \begin{bmatrix} E_{max} \\ BER_1 \\ BER_2 \\ \vdots \\ BER_n \end{bmatrix}$$

α_{BER_i} and β_{BER_i} are empirical coefficients for linear estimation of BER for the i^{th} sub-carrier in terms of the modulation level and the truncation length of the decoder. P_f is the linear coefficient, which models frequency vs. power characteristics of the image processing core.

The server (leader), on the other hand, attempts to minimize the overall energy consumption of the client-server system given the channel conditions provided by the client. The optimization problem is mathematically formulated as:

$$\arg\min_{\hat{Y}} \left\{ \left\langle C, \hat{X} \right\rangle + \left\langle S, \hat{Y} \right\rangle : D\hat{Y} \ge R\hat{E}Q^S,\ \hat{Y} \in PLS \times MLS^n \times FS,\ \hat{X} \in \Psi_L(\hat{Y}) \right\} \quad (19)$$

where *S* is the vector shown below. $\left\langle S, \hat{Y} \right\rangle$ signifies the power consumption (i.e., per-frame energy consumption) of the server. *D* is the coefficient matrix for linear estimation of the VQ and BER in terms of the SNR and the modulation level. $R\hat{E}Q^S$ is a vector representing the minimum requirements for the VQ and BER. The $D\hat{Y} \geq R\hat{E}Q^s$ constraint ensures that a minimum video quality and BER for each frame is achieved.

$$S = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \\ P_f \end{bmatrix} \quad R\hat{E}Q^S = \begin{bmatrix} VQ_{\min} \\ BER_1 \\ BER_2 \\ \vdots \\ BER_n \end{bmatrix} \quad D = \overbrace{\begin{pmatrix} 0 & 1 & 1 & \cdots & 1 & \kappa_f \\ \omega_1 & \delta_{BER_1} & 0 & \cdots & 0 & 0 \\ \omega_2 & 0 & \delta_{BER_2} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \omega_n & 0 & 0 & \cdots & \delta_{BER_n} & 0 \end{pmatrix}}^{(n+1)\times(n+2)} \quad (20)$$

ω_i and δ_{BER_i} are the estimated channel transfer function and the BER estimation coefficient for sub-carrier i. Notice that when solving equation (19), the optimization variables are $\hat{X}$ and $\hat{Y}$, that is, the server estimates the best strategy that the client may put in practice in response to the server's strategy and then based on this estimate of the client's policy, the server goes on to determine both the optimum values of the server and the client parameters. It will then implement the server policy, but will not report or in any way attempt to enforce the client policy. The client in its own turn will determine its optimum policy parameters as explained previously.

5 Experimental Results

We implemented an MPEG-4 FGS streaming system on a high performance testbed. 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 @200MHz to 1.5V @733MHz. 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. RTP/RTCP on UDP was used as a network protocol between the server and the client. Both the server and the client were equipped with an IEEE 802.11b wireless LAN (WLAN) card. Energy consumption of the WLAN interface was measured by using a data acquisition (DAQ) system.

In order to simulate the system, the Simulink 5.0 environment from Matlab 6.5 Release 13, was used. To model a multi-path fading channel, a parallel combination of Rayleigh and Rician fading propagation channels was used [9]. The maximum Doppler shift and the spreading factor of the Rician fading channel were set to 40Hz and 1, respectively.

In order to take into account the effect of multi-path fading, three different paths with delays of 2us, 3us, and 5us and gains of –3, 1, and 2 were considered in Rayleigh propagation channel. The characteristics of this channel were simulated and recorded for duration of 4800 frames, i.e., five minutes @15 frames/second. To produce the channel probability distribution, $h(\omega_i)$, this data is used. Then these probability values are fed into the Stackelberg game for policy design. To show the effectiveness of our approach three different scenarios were simulated and compared with each other. In scenario number one, for each time slot i, after the detection of channel conditions we assigned enough energy to the server and the client to support the specified average video quality.

In scenario number two, average channel condition was used to determine the required energy in each timeslot, and finally in scenario three our adaptive algorithm was used to calculate the required energy for each timeslot.

Figure 3 shows the lifetime versus required average video quality graphs for scenarios 1 and 3. Total initial energy for this experiment is set to 600 J. According to this graph, our approach increases the system lifetime by as much as 20% for high values of the required video quality. On average, the proposed method increases the system lifetime by more than 15% over the whole range of video qualities.

Figure 4 shows the comparison between scenarios 2 and 3, since in scenario 2, the assigned energy to each frame is fixed, and is selected according to the average channel behavior, the system consumes extra energy to produce higher video quality, and hence, has a lower system lifetime.

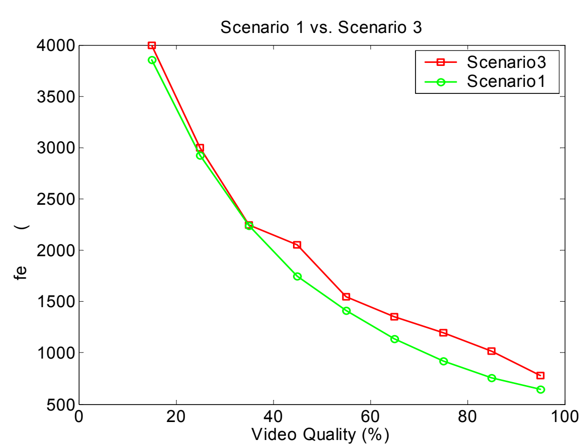
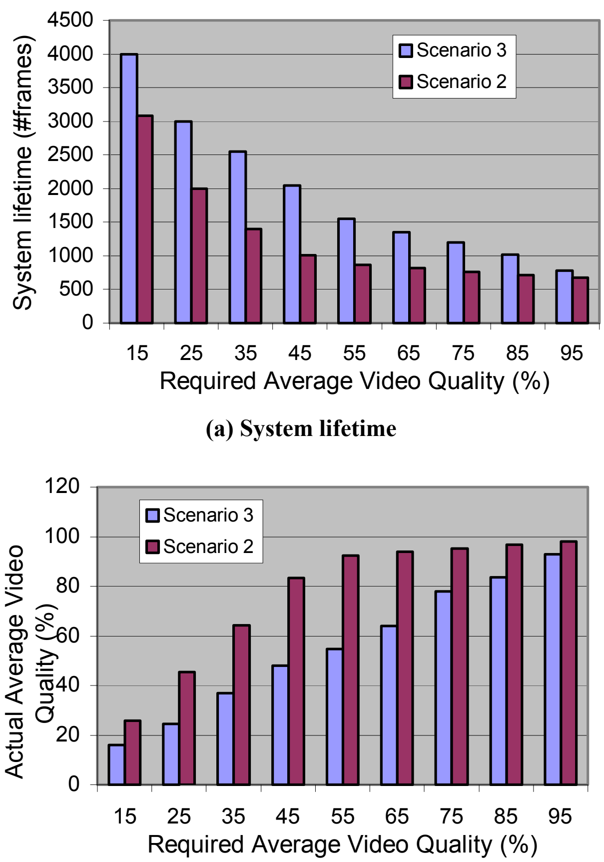


Figure 3. Lifetime comparison between Scenario 1 and 3

Figure 4(a) shows comparison between the system lifetimes for scenarios 2 and 3. It is clear that system lifetime is significantly increased for scenario 3 where we employed the proposed dynamic policy approach. Notice that the average video quality was maintained above the required value. However, for scenario 2, the average video quality is unnecessarily improved, which may be OK if there was no energy dissipation overhead (cf. Figure 4(b)).

6 Conclusion

Mobile video streaming system energy consumption is modeled and analyzed. Using this model an adaptive approach to energy assignment to each frame is proposed. The proposed approach guarantees the minimum video quality for all frames, and also meets a required average video quality over lifetime of system. Actual experimental data is used to extract the proposed models parameters and based on these parameters simulation are setup to show the effectiveness of the proposed approach. Based on this approach the overall system lifetime is increased by 20%.



(a) System lifetime

(b) Video quality

Figure 4. Comparisons between Scenarios 2 and 3

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, Mar. 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," Proc. of the ICME, vol.3, pp.1487-1490, 2000.

[3] R. Cohen and H. Radha, "Streaming Fine-Grained Scalable Video over Packet-Based Networks," Proc. of the GLOBECOM. IEEE, 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, Dec. 13-15, 2000 Sydney, Australia.

[5] J. Proakis, Digital Communications, McGraw-Hill, 3rd Edition, 1995.

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

[7] K. Choi, K. Kim, and M. Pedram, "Energy-aware MPEG-4 FGS Streaming," Proc. of the Design Automation Conference, pp.912-915, 2003.

[8] T. Pering, T. Burd, and R. Broderson, "The simulation and evaluation of dynamic voltage scaling algorithms," Proc. of Int'l Symp. on Low Power Electronics and Design, pp.76-81, 1998.

[9] Wireless propagation bibliography, http://w3.antd.nist.gov/wctg/manet/wirelesspropagation_bibliog.html

[10] T.M. Cover, J.A. Thomas, Information Theory, 2nd Edition , John Wiley & Sons Inc., New York, 1991.

[11] S. Appadwedula, M. Goel, N.R. Shanbhag, D.L. Jones, and K. Ramchandra, "Total System Energy Minimization for Image Transmission," Journal of VLSI Signal Processing Systems Feb. 2001.

[12] Stephan Dempe, Foundations of Bilevel Programming, Kluwer Academic Publishers, Boston, 2002.

[13] R. Henning and C. Chakrabarti, "Low-power approach to decoding convolutional codes with adaptive viterbi algorithm Approximations," in Proc. of Proc. of Int'l Symp. on Low Power Electronics and Design, pp. 68 –71, Aug. 2002.

[14] F. Chan and D. Haccoun, "Adaptive Viterbi Decoding of Convolutional Codes over Memory less Channels," IEEE Trans. on Comm., Vol. 45, No. 11, pp. 1389-1400, Nov. 1997.

[15] S. Swaminathan, R. Tessier, D Geockel, and W. Burleson, "A dynamically Reconfigurable Adaptive Viterbi Decoder," in Proc. of the FPGA Conf., Monterey, California, Feb. 2002.

[16] C. F. Lin and J. B. Anderson, "M-Algorithm Decoding of channel convolutional Codes," in Proc. of Princeton Conference of Information Sci. and System, pp 362-366, Princeton, NJ, Mar. 1986.

[17] H.v. Stackelberg, "Marktfrom und Gleichgewicht", Springer-Verlag, Berlin 1934. engl. Transl. The theory of the Market Economy, Oxford University Press, 1952.