Fugue: Time Scales of Adaptation in Mobile Video

Mark D. Corner, Brian D. Noble, Kimberly M. Wasserman
Department of Electrical Engineering and Computer Science
University of Michigan, Ann Arbor, MI

ABSTRACT

Providing interactive video on hand-held, mobile devices is extremely difficult. These devices are subject to processor, memory, and power constraints, and communicate over wireless links of rapidly varying quality. Furthermore, the size of encoded video is difficult to predict, complicating the encoding task. We present *Fugue*, a system that copes with these challenges through a division along *time scales of adaptation*. Fugue is structured as three separate controllers: *transmission*, *video*, and *preference*. This decomposition provides adaptation along different time scales: per-packet, per-frame, and per-video. The controllers are provided at modest time and space costs compared to the cost of video encoding.

We present simulations confirming the efficacy of our transmission controller, and compare our video controller to several alternatives. We find that, in situations amenable to adaptive compression, our scheme provides video quality equal to or better than the alternatives at a comparable or substantially lower computational cost. We also find that *distortion*, the metric commonly used to compare mobile video, under-values the contribution smooth motion makes to perceived video quality.

Keywords: interactive video conferencing, mobile multimedia, wireless computing, distributed systems, H.263, time scales, adaptation

1. INTRODUCTION

Consider the task of building a small, hand-held, wireless video conferencing device. The central difficulty facing such a system is the constantly varying quality of the wireless channel. These changes are due to physical effects, and hence cannot be controlled. Instead, one must adapt to them.

There are several adaptive strategies available. For example, one can combat wireless fading with an increase in transmission power. However, there are physical and practical limits on the degree to which this is useful. This is particularly true for hand-held devices, which must rely on batteries until they can be charged at a stationary source. Beyond these limits, one must resort to rate-limiting strategies.

Video is particularly amenable to rate adaptation through reductions in *fidelity*, such as lowered frame rate or frame quality. However, it is often unclear how best to balance these two; such decisions must be guided by user preference. Furthermore, the savings obtained through reductions in frame rate and frame quality is uncertain. One could speculatively encode a video sequence several different ways in order to precisely match a target bit rate. Unfortunately, such speculation is computationally expensive. This is a chilling prospect in the embedded systems domain, where cost is a primary concern.

In order to manage these adaptive strategies, one could create an integrated, monolithic system. However, doing so would be unnecessarily complicated and difficult to maintain and evolve. We have chosen instead to attack the problem by dividing adaptive techniques according to the *time scales* over which they are effective. This leads to a simple, elegant design for providing interactive video services on mobile devices.

This paper presents *Fugue*, our realization of this design. It is composed of three separate *controllers*, each of which has a simple interface to the others. The *transmission controller* uses a truncated power, rate adaptive scheme to mask short-term fluctuations in wireless channel quality. The *video controller* chooses fine-grained video compression parameters to provide the best quality frames possible given an instantaneous measure of bit rate. The *preference controller* balances the conflicting demands for improved frame rate, frame quality, and transmission power given a long-term average channel state.

We present the detailed design of each of these controllers. We then present an initial evaluation of our design. We first detail the modest space and time costs required to implement the three controllers. Then, we present a simulation to demonstrate the effectiveness of combining rate and power control in the transmission layer over power control alone. We conclude our evaluation by comparing our encoding algorithm to three other schemes: a more expensive scheme that speculatively produces several different encodings for each frame, and two simpler schemes.

We find that when the available bit rate over-constrains the choice of video encodings, the best strategy is to conservatively encode the video at a single, static quality. When there is some freedom in choosing encoding parameters, our scheme provides video quality equal to or better than the others at a comparable or substantially lower computational cost. Finally, *distortion*, a metric commonly used to compare the qualities of differently-encoded video streams, over-values sharpness of frames compared to smoothness of motion. Focusing on distortion rather than perceptual quality can lead to incorrect compression decisions.

2. RELATED WORK

The Odyssey system[21] supports applications that vary their fidelity in response to changing network conditions. Like Fugue, Odyssey focuses on end-host adaptation. Odyssey infers changes in network performance from end-to-end packet observations; in contrast, Fugue considers wireless channel quality for finer-grained adaptations. Furthermore, Odyssey's video application deals only with stored video that has been pre-encoded at multiple fidelities.

In contrast to end-host approaches, several systems have placed responsibility for supporting adaptation in the network. Mobiware[1,4] is an architecture based on programmable, active services placed throughout the network. Each application submits a utility curve, and centralized points provide utility-fair allocations. The TIMELY architecture[3] provides a similar model based on revenue maximization. It also adds lower-bound resource reservations, with the possibility of additional resources allocated to increase revenue. These systems require substantial in-network deployments, while Fugue depends only on local observations. However, Mobiware and TIMELY both provide first-class support for handoff, while Fugue focuses mainly on single-channel performance. Fugue's preference controller could be used as a basis for generating utility and revenue curves, something that these systems have not fully addressed.

Fox et al.[9] promotes active services to degrade the fidelity of video at key points in a multi-hop network. The system handles interactive data, but only adapts the frame rate of the video and does not consider the particulars of wireless transmission or end-client preferences. MobiWeb,[20] another proxy-based solution, employs conventional resource reservation. Fugue, like Odyssey and Mobiware, takes the position that adaptation is preferable to reservation in rapidly varying wireless networks.

Bahl[2] presents a framework for supporting video over wireless networks. This framework explores the same space of unpredictable channel performance and compact, low power devices as Fugue. He puts forth the notion that adaptive techniques apply over diverse time scales, but does not structure his system around this observation. His system relies on replacing standard encoders with multi-resolution codecs, such as wavelets, and complex network reservation schemes such as RSVP.

Others have proposed video systems that control codecs and channels for variable wireless bit rates. Rate adaptable coding has been explored,[19] but does not take advantage of controlling video codec parameters. Perceptual quality metrics[26] were applied as well, but only for evaluation and not as a control method. Another method for optimization is to match multiple pre-encodings of the video[14] to a given a rate, optimizing each frame for low distortion. This method has a high computational burden; it also strives to minimize frame distortion at the expense of frame rate, diminishing perceived quality. Some work has been done to find optimal encodings using Lagrange multipliers,[27] but network supplied rate constraints were not considered. Adjusting the video frame rate has been considered in conjunction with a bit allocation scheme,[25] however prediction of bit rates and distortion from quantization parameters is not explored.

Power control is an area that has received considerable attention. An analysis of truncated power control with outage has been explored[10] as well as an analysis of truncated rate/power control[18] over Nakagami fading channels. Shadowing has not been considered, nor has it been shown how one can integrate the rate estimates with upper layer controllers.

3. HAND-HELD, MOBILE VIDEO

Hand-held wireless devices present several inherent constraints, and are subject to many sources of variability in performance. Processor and battery power are limited, wireless link performance changes rapidly, and the efficacy of video encoding algorithms is uncertain. There are a number of techniques one can use to cope with such dynamic change. For example, one can vary transmitter power, bit duration, frame rate, or frame quality. Unfortunately, none of these are able to cope with all sources of variation, and each is applicable to different components in the system.

3.1. Constraints and Variations

Hand-held devices are subject to unusually severe engineering constraints, particularly those of cost and power efficiency. For such devices to have mass-market appeal, cost must be a first-order concern. In contrast to general purpose devices, these embedded systems generally do not take full advantage of the aggressive improvements in capacity and performance of components, instead tracking reductions in cost. Therefore, when designing for these systems one must justify any techniques requiring additional processing power or memory.

Similarly, the power budgets on these devices are tight. Battery capacity, in terms of deliverable energy per pound, is growing at an extremely slow pace. Users often have some expectations of how long the device must operate, and this time

horizon can be used to adapt power consumption behavior. [8] Barring such user-supplied information, devices are typically designed with a specific battery life; this cannot be provided at the expense of extra weight. For example, current laptops must have a rated battery life of at least a few hours, and can not weigh more than several pounds. This limits the amount of energy one can devote to processing power and wireless transmission.

Wireless networks also give rise to substantial challenges. Wide-area coverage necessarily provides lower bit rates to individual devices[17] using channels that exhibit rapidly, dramatically changing performance. [22] The frequency of change depends on the speed of the mobile device. Slow moving users may have bad signal quality for long periods, and fast moving users may suffer from channels that are difficult to measure and react to.

Our target application presents its own difficulties. Interactive video is intolerant of latency beyond human perceptual limits. Therefore, even if one could devote resources on the device to buffering transmitted traffic, there are limited opportunities to do so. Furthermore, users place different values on the importance of high-quality frames versus smooth motion. A user watching a live lecture may prefer high resolution in order to read notes written on the board. Someone watching a sporting event may instead want the fluid motion provided by a high frame rate.

3.2. Coping with Variations

There are several ways to hide or adapt to variations in performance and user preference. At the transmission layer, these include power control, rate control, and adaptive coding and retransmission schemes. At the application layer, one can change the base frame rate at which video is delivered, and apply various degrees of lossy compression to components of each frame.

Power control — varying the power at which packets are transmitted — can be used to combat many sources of channel fading. However, for fast-moving devices, it is often hard to adapt power to the rapid variation in the channel's signal-to-noise ratio. Even for slow-moving devices, such power expenditures may be unwise due to battery limitations. Instead, one could combat losses in the channel by adapting the transmission rate; a low-rate sender will suffer fewer packet losses under fading channels than a high-rate one.

Combining these mechanisms is not straightforward. For example, it is better to increase power — and hence available bit rate — if doing so would not drop battery life below the desired horizon. Furthermore, transmission-layer adaptations are limited; they can smooth the variations seen by higher layers, but cannot hope to remove them entirely. Instead, higher layers are forced to adapt to changing circumstances. [16,24] One way is to change the *fidelity* of delivered data, [21] trading delivered data quality for resource consumption.

Video is particularly amenable to lossy compression, such as that provided by H.263. [23,13] We have based our system on H.263 for two reasons. First, it is explicitly designed for the low bit rates common in wireless deployments. Second, its algorithms form the core of the visual component of MPEG-4, [15] a system-level encoding standard.

These compression schemes use two techniques to reduce the size of the video: inter-frame motion compensation and intra-frame quantization. The quantization factor determines the quality of the resulting frame. In addition to per-frame compression, one can also vary the base frame rate at which the video is encoded. However, decreasing the frame rate increases the motion between adjacent frames, compounding demands on the encoder. Balancing these two dimensions of fidelity requires consideration of user preferences for smooth motion or sharp resolution.

For stored, off-line video, it is easy to measure the effectiveness of all possible compressions a priori, and then match a particular encoding to the available bit rate. Pre-computation is not possible for interactive video; look ahead is limited by latency intolerance. Because the effectiveness of encoding is dependent on inter-frame motion and intra-frame entropy, the sizes of each potential encoding are difficult to predict. One can encode each frame a number of different ways, and then choose the one that empirically fits the available rate. Such speculative encoding requires substantial processing power, which is in short supply on a hand-held, embedded device.

4. TIME SCALES OF ADAPTATION

Variation in link quality, size of the encoded video, and possible changes in users preferences lead to a complex, dynamic system that is difficult to control. One possible design is a monolithic but complicated system that integrates every part of the encoding and transmission process. However, a modular system based on adaptations appropriate to each disturbance simplifies the design, while yielding the desired performance.

Our system is structured around the notion of *time scales of adaptation*; adaptive techniques are arranged according to the time scales over which they are effective. There are four parameters we can control: frame rate of delivered video, quantization level of each separately encoded portion of a frame, transmitter rate, and transmitter power. Each of them is subject to a different set of constraints, and can be used to adapt to video and link variations on a different scale. Frame rate and average frame quality are subject to user preference, and can only be used to combat very long-term changes on the order of hundreds of milliseconds. Individual quantization choice is constrained by the desired long-term average bit rate, and applies only to

changes within an individual frame: tens to hundreds of milliseconds. The transmission layer parameters — transmitter rate and power — apply to individual packets, and are effective on the same and smaller scales. Figure 1(a) depicts the the logical organization of these layers.

In Fugue, each layer is realized as a separate controller. The remainder of this section describes each controller in turn, from shortest to longest granularity: the *transmission controller*, the *video controller*, and the *preference controller*. In this section, we present the responsibilities of each of these controllers and detail their interactions. Section 5 presents the design of each controller in detail.

4.1. Transmission Controller

At the finest grain, the transmission controller manages the transmission of packets across a wireless channel. This channel is subject to degradation from a number of sources, including multi-path fading and shadowing. As the channel quality drops, the bit error rate (BER) may exceed a tolerable level. The transmitter can combat increases in BER by increasing the transmission power, thereby improving the received signal.

Such power increases are limited by a physical transmitter maximum, but the practical limit might be below that maximum in order to maintain the desired battery life. When faced with such limits, the controller can lower the transmission rate instead. Transmitter power, P, and rate, T, are determined by an average power constraint, P_{av} and spot observations of channel quality.

The transmission controller must produce two estimates of bit rate for higher levels. The first is an instantaneous rate, R, used by the video controller. The second is a long-term average channel rate, R_{av}, used by the preference controller.

4.2. Video Controller

In many encoding schemes, an individual frame's pixels are grouped into regions called *macroblocks*; a typical size for these is sixteen by sixteen pixels. These macroblocks are grouped into *groups of blocks*, or *GOBs*. The GOB is the unit of compression; an encoder can vary the degree of quantization, or Q, for each GOB. This allows the encoder to adapt the size of encoded video every ten to fifteen milliseconds for QCIF frame sizes.[12]

The video controller is given a target frame rate, F, and initial quantization, Q_{init}, by the preference controller, as well as an instantaneous rate, R, by the transmission controller. The video controller's goal is to produce the highest-quality GOBs it can without exceeding the transmission time budget. For example, if the target rate is ten frames per second, the video encoder wants to produce an encoded frame that will take 100 ms to transmit in current channel conditions.

4.3. Preference Controller

In the long term, the system must trade off three competing concerns: frame rate, frame quality and battery life. The preference controller must relate these parameters through a *cost function*, which is expressed as three independent functions of F, Q_{init}, and P_{av}. By optimizing this cost function, the preference controller chooses optimal values for each of these parameters, and exposes them to the other controllers.

In general, the cost function must come from user and application input. However, one can implement reasonable defaults. The cost function for P_{av} can be derived from the desired battery life of the device; this can be provided by the user or the system designer. The cost functions for F and Q_{init} can be based on perceptual quality. By relating perception to measurable features of encoded video,[26] Fugue produces video that matches an average viewer's expectations.
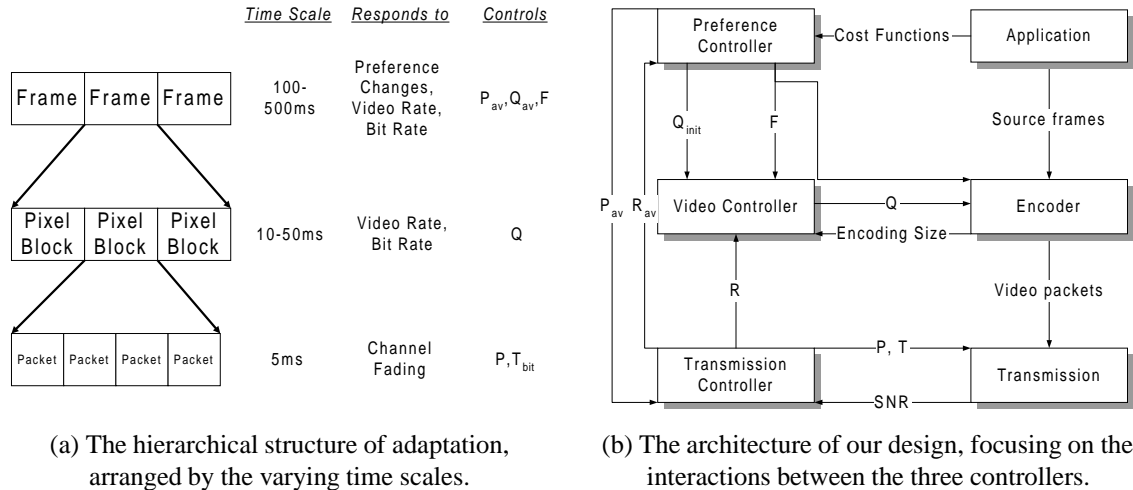
5. CONTROLLER DESIGN

Figure 1(b) summarizes how the three controllers work in concert. Each controller is shown with the inputs it considers in making decisions, along with the set of outputs it produces — either for other controllers, or for the encoder and transmitter. This section describes each controller in detail.

5.1. Fine Grain: Power and Bit Duration

The transmission controller smoothes variations in wireless channel quality by monitoring the channel and matching transmission power and rate to it. This truncated power, rate-adaptive scheme was analyzed for Nakagami channels [18] and similar analysis has been done for a system without rate adaptation,[10] referred to as channel inversion. Our analysis includes the addition of shadowing and an approximation technique to simplify implementation in a low-cost, embedded device.

The choice of transmission power is limited by the desired long-term average transmission power, P_{av}, which is supplied by the preference controller. By combining P_{av} with observations of channel behavior, the transmission controller supplies instantaneous and long-term transmission rates, R and R_{av}, to the other controllers. Due to space constraints, we provide only a sketch of the derivation; details can be found elsewhere.[6]

A transmitter sends a single bit with a certain power, P, for a certain duration, T. The total energy received, E_b, is determined by the channel gain. Because gain is a time-varying property, we write it as $g(t)$. The receiver measures $g(t)$, and

	Time Scale	Responds to	Controls
Frame Frame Frame	100-500ms	Preference Changes, Video Rate, Bit Rate	P_{av}, Q_{av}, F
Pixel Block Pixel Block Pixel Block	10-50ms	Video Rate, Bit Rate	Q
Packet Packet Packet Packet	5ms	Channel Fading	P, T_{bit}

(a) The hierarchical structure of adaptation, arranged by the varying time scales.

(b) The architecture of our design, focusing on the interactions between the three controllers.

Figure 1. Controller Design

reports it to the sender. The sender now has information about what the gain was one round-trip time, Δt, in the past. In other words, at time t, the sender knows $g(t - \Delta t)$. If one assumes that gain — and hence fade state — is slow to change, one can assume that this estimate is current; this turns out to be true for relatively slow-moving nodes. Faster nodes must augment this scheme with error correcting codes and interleaved transmission.

Given $g(t)$ and knowledge of the channel modulation scheme, it is easy to compute the instantaneous transmission power required to keep the probability of bit error, P_b, below a specified bound, $P_{b,max}$. Unfortunately, one must be careful in adjusting power; if one expends too much power early in the battery's lifetime, there may not be enough residual energy to meet the user's needs. So, we must cap the transmission power at P_{max}.

One can compute the value of P_{max} given the desired average power consumption, P_{av}, plus some knowledge about channel fading behavior. Our analysis combines Rayleigh-distributed multi-path fading with shadowing to produce an overall channel model. In particular we use the probability density function of the channel fade state. With these, one can express P_{av} as an integral of the fading state over the range of possible power levels; this range is capped by P_{max}. Solving numerically, one can express P_{max}, as a function of the average power, P_{av}. Figure 2(a) plots P_{max} for values of P_{av}, using a set of reasonable assumptions detailed in Figure 6.
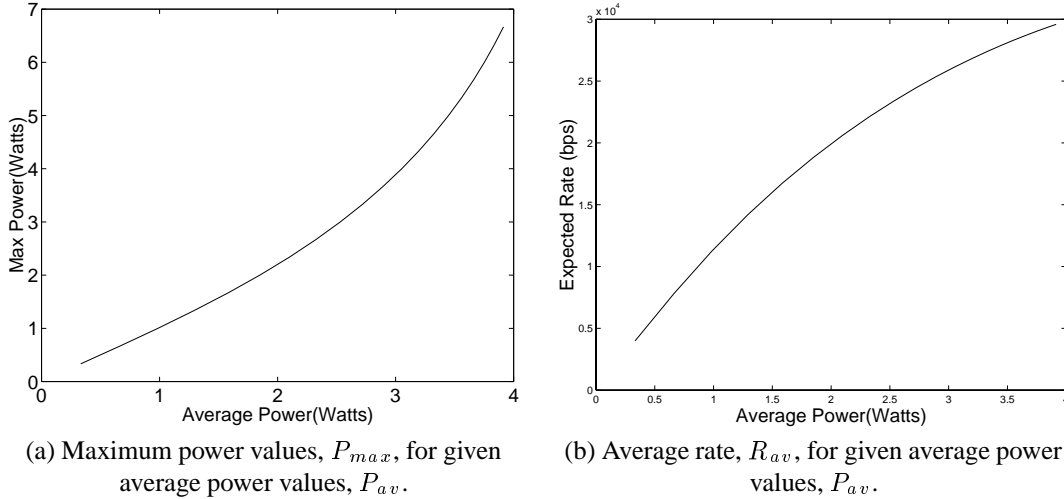
One question remains: what do we do when the channel requires more power to maintain $P_{b,max}$ than we are willing to expend? Recall that the figure of merit is the amount of delivered energy per bit, E_b, which is proportional to the power with which the bit is transmitted multiplied by time used to transmit it. Therefore, when we cannot increase the power, we can instead lengthen the bit transmission duration. As bit duration — and hence transmission rate, R — change over time, R is reported to the video controller for medium-grain rate matching.

The transmission controller must also report the long-term average transmission rate, R_{av}, to the preference controller. This is determined by the fluctuations of the instantaneous rate, R, which are in turn driven by same fading models that allowed us to derive the function for P_{max}. R_{av} can thus be described as a function of P_{av} by numerically solving a similar integral. Figure 2(b) plots R_{av} for values of P_{av}, computed with same set of parameters as for P_{max}.

The equations used to derive these functions are difficult to compute, but the functions themselves are smooth and monotonic. Therefore, we can avoid generating them on the fly by precomputing them for a number of P_{av} values, and interpolating between these precomputed values.

5.2. Medium Grain: GOB Quantization

The video controller operates under two sets of constraints. It is given instantaneous bit rate information from the transmission controller and a frame rate and initial quantization value by the preference controller. By combining the frame rate and bit rate information, the controller derives a *bit budget* for each frame; it strives to send each frame at the initial quantization value. However, since the bit rate can vary and the prediction of encoded video sizes is uncertain, the video controller must adjust the quantization parameter as it encodes the frame. The quantization parameter, Q, can be changed at the beginning of each GOB to increase or decrease the rate of the video.

(a) Maximum power values, P_{max}, for given average power values, P_{av}.

(b) Average rate, R_{av}, for given average power values, P_{av}.

Figure 2. Maximum Rate and Power

Before encoding each GOB, the controller checks to see how much time remains. If the estimated encoded size of the remaining GOBs is too large, then the quality is lowered. If the lowest quality produces GOBs that are too large, then the overrun is subtracted from the next frame's budget. If there is an under-run, that transmission capacity is lost; the next frame cannot be encoded until it becomes available.

The following equation expresses the decision that the controller makes at the beginning of the ith GOB:

$$\sum_{g=0}^{i-1} \frac{E(f,g)}{R(g)} + \sum_{g=i}^{GOB-1} \frac{E^*(f,g)}{R^*(g)} \leq \frac{1}{R_f} - O(f-1). \tag{1}$$

In this equation, $E(f,g)$ denotes the number of bits consumed by the previous GOBs and $R(g)$ is the actual bit rate during their transmittal. $E^*(f,g)$ is the estimate for the rest of the GOBs in the frame and $R^*(g)$ is the estimate for the transmission rate during the rest of the frame. If R_f is the frame rate, then $1/R_f$ is the time budgeted for this frame, and $O(f-1)$ is the overrun, if any, from the previous frame.
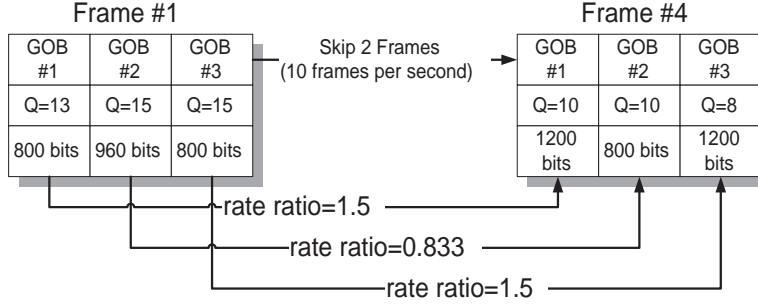
The system uses an empirical model to estimate the size of encoded GOBs. This model is derived from an experiment that measures bit rates of several video sequences encoded by an H.263 codec. Since quantization decisions are specific to a GOB, we measure at that granularity.

The benefits of lossy compression applied to video are highly dependent on scene content and motion. However, within a scene, the same GOB in two adjacent frames is likely to contain similar information. If the frame rate or GOB quantization did not change, one would expect the sizes to be similar across frames, modulo scene changes. However, such scene changes are likely to be rare for interactive, live sources; they typically arise through off-line editing.
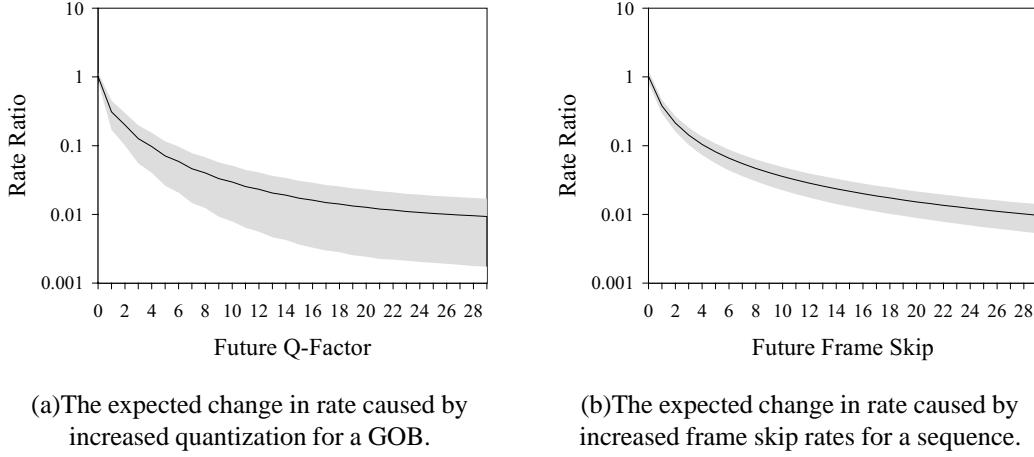
Due to these observations, we use the encoded size of the previous frame's GOB to predict the encoded size of that GOB in the current frame. We express changes in frame rate or quantization as the ratio of encoded sizes of the two GOBs. By measuring these ratios for each combination across several test sequences, we generate a distribution of ratios. These test sequences are single-scene, without abrupt changes. Measuring ratios removes some dependence on per-video differences. In order to make this experiment tractable, we make the simplifying assumption that changes in frame rate and quantization are orthogonal. While this is not strictly true, it turns out to give adequate results.

An example of this measurement is shown in Figure 3. The figure shows two sample frames of encoded video broken into three GOBs. The video source provides 30 frames per second. In this example, it is encoded at ten frames per second, skipping two source frames between each pair of encoded frames. This is referred to as the *frame skip rate*. The encoder uses different Q values for each GOB. Depending on the amount of motion between frames, it produces a variable number of bits. The ratio of the sizes is measured for each pair of current and future Q values. For example, the first GOB has a ratio of 1.5 for a present Q value of 13 and a future Q value of 10.

Because we treat changes in frame rate and quality separately, there are different experiments to measure the impact of each. In the first experiment, the encoder processes a suite of eight commonly available test videos at 10 frames per second

Figure 3. Rate Ratio Example

(a)The expected change in rate caused by increased quantization for a GOB.

(b)The expected change in rate caused by increased frame skip rates for a sequence.

Figure 4. Bit Rate Ratios

and at each integer quality level in the range [1..30]. Figure 4(a) shows the results for an initial Q of 1. The solid line gives the average ratio across all test sequences, and the shaded region gives the standard deviations for each point. Unfortunately, the uncertainty in prediction is quite high; the median standard deviation is 80% of the mean across all experiments. Furthermore, Figure 4(a) shows the best case; when the prior GOB encoded at full quality, it gives the most predictive power about the current GOB. Estimates based on lower-quality GOBs are even less certain.

In the second experiment the encoder processes the same suite of test videos — captured at 30 frames per second — at a constant quantization level of 13 and at each frame skip rate, between 0 (30 fps) and 29 (1 fps). The experiment measures the rate ratios of the different frame skip rates. The results for switching from full frame rate to a lower one are presented in Figure 4(b).

As with Figure 4(a), starting from full frame rate gives the best basis on which to predict. However, the standard deviation of the frame rate observations is lower than that for a quality change; the median standard deviation is 40% of the observed mean. This implies that changes in frame rate require less adaptation than quality changes. Since quality changes are done at a smaller time scale than frame rate changes, we are better able to cope with the unpredictability shown in these experiments.

5.3. Coarse Grain: Global Preferences

The preference controller must match the long-term changes in available bandwidth with changing user preferences, expectations for battery life, and underlying video properties. It specifies the target frame rate of the encoded video, the initial quality parameter provided to the encoder, and the average power constraint for the transmitter. There is a clear tradeoff between these parameters; given more power, higher bit rates can support either increased frame rates, increased quality, or both.

At the beginning of a video frame the preference controller sets its three parameters to meet the expected bandwidth constraints in the system. It does so by minimizing a cost function that weights the parameters according to user or application preferences. The cost function at frame number f is:

$$J(f) = \lambda_P \left(P_{av} \left(f \right) \right) + \lambda_F \left(F \left(f \right) \right) + \lambda_Q \left(Q_{init} \left(f \right) \right),$$ (2)

where $P_{av}(f)$, $F(f)$ and $Q_{init}(f)$ denote the transmitter power, the frame skip rate, and the initial Q-factor at frame f, respectively; the $\lambda\left(\ldots\right)$ terms are the associated cost functions.

Several constraints must be met in optimizing this cost function. The total estimated bit rate created by the video coder must match the available rate of the channel. This constraint is:
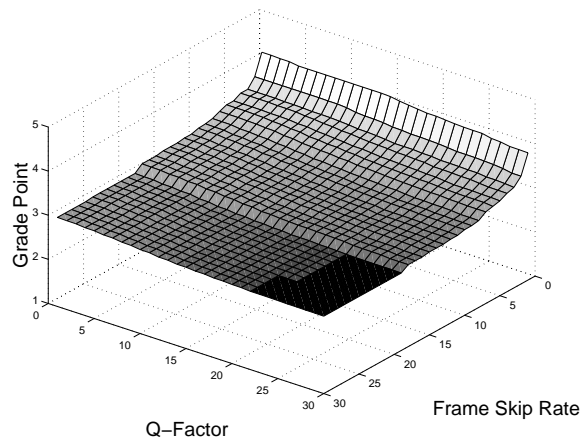
$$\sum_{g=0}^{GOB-1} \frac{E^*(f,g)}{R^*(g)} = \frac{1}{R_f} - O(f-1). \tag{3}$$

This equation is the same as (1), with $i = 0$.

Although applications can supply their own cost functions based on user preference, we provide a set of reasonable defaults. These default cost functions comprise two components. First, we incorporate a perceptual model of video that allows us to trade quantization and frame rate. Second, we provide a *cliff function* to value power, based on expectations of battery life supplied by the user of the device or its designer.

Fugue's perceptual quality model is based on a set of objective metrics — developed by Webster et al. [26] — for measuring the subjective quality of encoded video at different quality and frame rates. These metrics were selected by matching a linear combination of three quantitative measures of encoded video to qualitative observations. Human subjects were shown a set of test videos encoded at varying quantizations and frame rates. The test videos were drawn from a number of domains, and featured varying degrees of scene change, inter-frame motion, and scene detail. The subjects rated the resultant quality on a one through five scale, with five being the best. In parallel, Webster et al. examined a large set of quantitative metrics, and chose the three that together most successfully predicted user satisfaction. The first of these metrics, the sharpness of a rendered frame, captures spatial quality. The other two — the motion lost and the perceived burstiness added at lowered frame rates — express temporal properties.

We have implemented these metrics and applied them to eight "talking head" test videos. We filtered the results to ensure that grades were monotonic in quantization and frame rate; at high quantization factors, blocking artifacts are mistaken for increased detail and given positive weight by the first metric. This experiment yields a single, empirical valuation for the sum of the cost functions $\lambda_F\left(F\left(f\right)\right)$ and $\lambda_Q\left(Q\left(f\right)\right)$, and is depicted in Figure 5.

Figure 5. Subjective Grade Point

This graph shows several features of interest. First, all frame skip rates equal to or higher than 17 result in roughly equal grade points, given the same quantization. This means that, for extremely low frame rates, it is usually better to try to increase frame quality than to increase frame rate. However, at frame skip rates lower than 17, it is almost always better to increase frame rate rather than frame quality. Finally, recall that Figures 4(a) and 4(b) show that adjustments in quantization and frame skip each cover roughly two orders of magnitude variation in rate. However, they affect grade point differently; quantization accounts for less than one third of a grade point, while frame rate accounts for more than a full point. Therefore, given a limited increase in available transmission rate, it will almost always be better to decrease frame skip provided one can ultimately achieve better than 2 frames per second[*].

---

[*]Note that the dynamic range of quantization's impact on grade point may be understated if our smoothing was too conservative. However, such underestimation does not change the basic strategy of decreasing quantization below 2fps, and frame skip above that rate.

To place a value on power, we take advantage of the fact that a user often knows how long she expects to use a mobile device before she can recharge its batteries.[8] It is not useful to have any power remaining after the lifetime has expired, and it is infinitely costly to use power faster than that rate. This results in a cliff cost function:

$$\lambda_P\left(P_{av}(f)\right) = \begin{cases} 0 & : & P_{av}(f) \leq P_{design} \\ \infty & : & P_{av}(f) > P_{design} \end{cases} \tag{4}$$

Absent user advice about expected operating times, one can instead rely on a designed-for battery lifetime to set P_{av}.

The cost of computing the optimal point is the most computationally expensive part of Fugue's control system. Let N_p, N_f, and N_q be the number of discrete power levels, frame rates, and quantization factors, respectively. A naive solution to this optimization problem runs in $O(N_p N_f N_q)$. However, we believe that the monotonic properties of the cost functions allow for simplification. While Fugue does not require that the power cost function be a cliff, having such a cost function further simplifies the optimization space.

6. EVALUATION

There are three sets of questions that drive the evaluation of our design:

- What are the computational and space costs of our controllers? Are they amenable to implementation on a hand-held consumer device?
- How effective is our power control and rate adaptation scheme at smoothing channel behavior? Can bit error rate be controlled to suit our encoding scheme? What is the resulting bit rate?
- Our encoding scheme incrementally constructs frames based on per-GOB predictions. Alternatively, one could pre-encode each GOB a number of different ways to optimize transmission. How much extra computational overhead does pre-encoding require? How does the quality of our produced video compare to that of pre-encoding? How does our scheme compare to simpler schemes?

In this section we present experiments to answer these questions. These experiments are based on a simulated physical channel that incorporates models for multi-path fading and shadowing. It is used to evaluate our transmission controller's efficacy in controlling BER. We have added a video controller and a preference controller to the Telenor/UBC H.263 encoder.[7] The unmodified encoder is computationally expensive, requiring approximately 210 milliseconds to encode a single frame on a 300 MHz Pentium II. This performance is similar to that reported for an MPEG-4 software encoder on contemporary SPARC processors.[28] We also implement alternate video controllers for comparison over traces taken from our wireless channel simulator.

6.1. Computational Burden

Each controller — transmission, video, and preference — imposes computational and space burdens on the hand-held device. The transmission controller is computationally simple. Because P_{max} and R_{av} are smooth, monotonic functions, the transmission controller can store tens or hundreds of pre-computed points, and interpolate between them. The video controller requires more space, since it must store a matrix of ratios for variations in frame quality, but needs to perform at most Q different multiplications and comparisons per GOB. Both of these costs are trivial when compared to the space and time costs of the H.263 encoder.

The preference controller also has modest space costs; it only needs to store the ratio matrices for F and Q. We have measured a brute-force implementation of the preference controller where $N_p = 20$, $N_f = 30$, and $N_q = 30$. When the values of each cost function are precomputed for each of these discrete-valued inputs, the total time to solution is less than 1 millisecond on a 300 MHz Pentium II. Compared to the cost of encoding a frame on the same processor, this is small. Furthermore, a more sophisticated solver that takes advantage of known cost-function properties should perform substantially better.

6.2. Transmission Layer

The two key goals for the transmission layer are to adapt bandwidth to channel quality without unduly complicating higher layers, and to meet the battery lifetime specified by the user or system designer. Of course, one could transmit at the maximum possible bit rate at all times, but this can often be counterproductive during periods of poor channel quality. We have not examined joint source channel coding, but assume that a reasonable number of errors can be corrected through channel coding and a limited use of Automatic Repeat Request (ARQ).[19] Limiting the number of errors in the transmission system can save bandwidth that would otherwise be consumed by unnecessary coding and ARQ retransmissions, and incurs the overhead only when the channel is actually poor.

In this section, we present the performance of three control schemes: no adaptation, power adaptation, and power combined with rate adaptation. These schemes are evaluated using a simulation of the wireless link. We simulate a Rayleigh fading distribution[22] using Clarke's model[5]; it assumes that multiple reflected waves will arrive with arbitrary phase and angle of arrival. Shadowing is simulated directly from an autocorrelation of the process, which has been shown to give results closely matching physical channels.[11]

We created traces of the combined Rayleigh and shadowing processes. We then tested each of three control schemes over those traces, and report the resulting bit error rate on the channel. The first scheme uses constant transmitter power. The second scheme adapts the power of the transmitter but does not adapt the rate once the maximum power limit is reached. The third scheme adapts both the power and rate. The BER for each of the three schemes are shown in Figure 7(a-c). Simulation parameters are given in Figure 6; they were chosen to be representative of a typical cellular wireless network.

Simulation Parameter	Value
Velocity	10 km/hr
Δt	500μs
$P_{b,max}$ (goal)	1E-5
T_{min}	3.1250E-5 s
P_{av} (goal)	3.56 W
P_{max}	5.33 W
Warmup period	10 sec.

Figure 6. Simulation Parameters

Figures 7(a), 7(b), and 7(c) show the channel bit error rates for the flat power control, truncated power control and the power and rate control cases, respectively. The first two cases show a BER as high as -0.4 dB. A BER this large will exceed the error correcting capabilities of the code and ARQ retransmissions will consume a large amount of bandwidth on the channel. However, Figure 7(c) shows that the rate adaptation system has a much lower BER. Rate and power adaptation are helpful in maintaining a usable channel for the video encoder. Figure 7(d) depicts the achieved bit rate of the rate-adaptive scheme. In effect, this scheme converts uncertain bit errors into known short-term rates.

To be a fair comparison, all three schemes must meet the average power constraint given by the preference controller. The flat power scheme automatically meets the average power since it is fixed. The other two schemes meet the average power constraint over time since the P_{max} value is set according to the computational method described in Section 5.1.
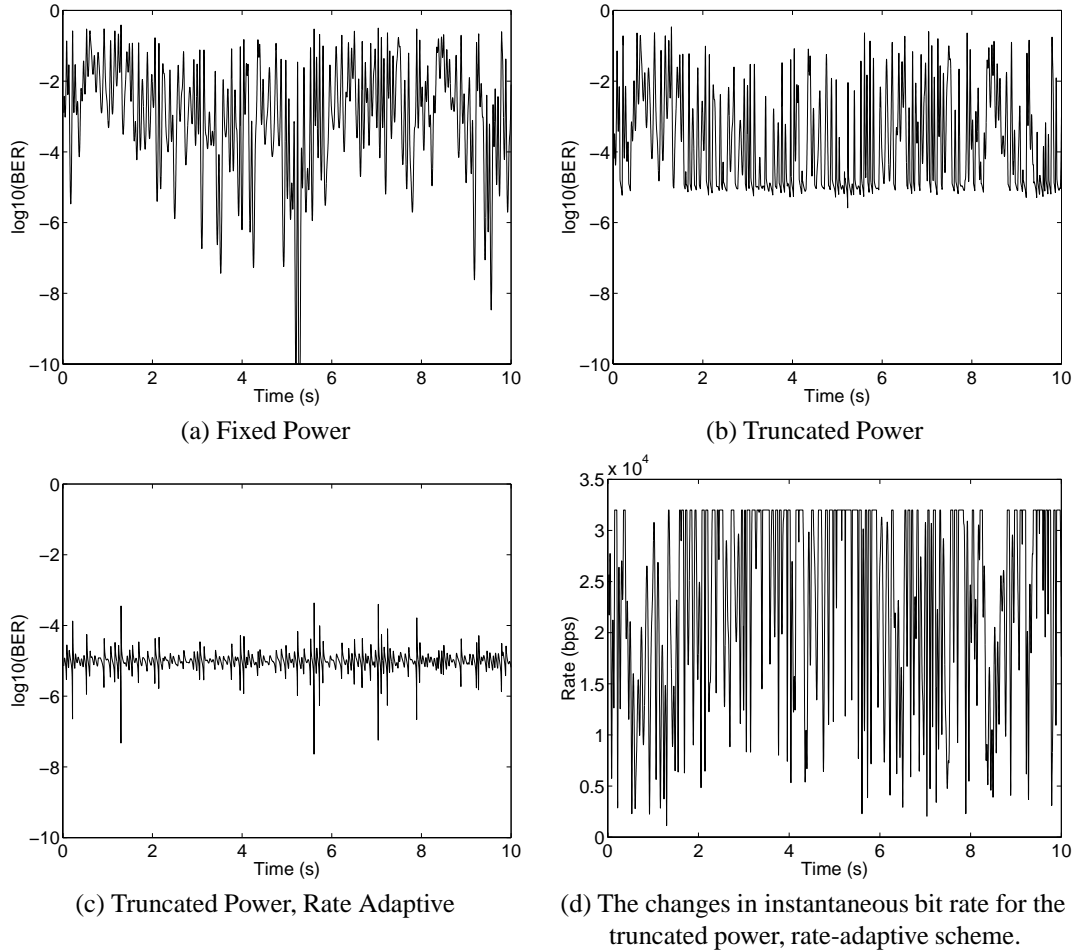
6.3. Video Layer

The final set of experiments explores the quality of the video produced by Fugue, compared to a number of alternate schemes. There are three different schemes against which we compare: static Q selection, per-frame Q prediction, and speculative pre-encoding. In static schemes, the video controller simply encodes every GOB at the same pre-selected Q. The per-frame scheme uses prediction of encoded sizes to select a single Q for all GOBs in a frame. While both these approaches are simpler than our own, the low overhead of our controller is not significant in comparison.

In contrast, the speculative scheme is more computationally demanding than the others. Its video controller pre-encodes each GOB in the frame using k different quantization values, and computes the resulting distortion of each encoded version.[14] The distortion metric is the signal-to-noise ratio in the luminance plane (YSNR). It can then compute the set of GOBs to send such that the bit rate constraint is satisfied while total YSNR is minimized. For interactive video streams, this optimization can take place only within a frame, not across them. This produces a per-frame encoding with the lowest possible YSNR, subject to the constraint that each GOB is encoded with one of the k values of Q.

The choice of which specific Q values to pre-encode has an impact on the effectiveness of this scheme. The scheme against which we are comparing considers four values: 12, 14, 20, and 30. Exploring more encodings allows better optimization within the given constraints. However, each additional Q considered increases computational overhead substantially. One encoding is required, the remaining $k-1$ are overhead. Profiling the H.263 encoder reveals that each additional Q value adds 22% to the base cost of encoding a frame; this is approximately 46 ms on our hardware.

We compare seven different video controllers using the physical channel simulation. The first four, which are the least expensive to compute, encode all GOBs at a single, static Q. Each of the four uses one of the levels suggested by the speculative scheme. The fifth controller predicts the best Q for a frame, and encodes each GOB in the frame with that Q. The sixth is our per-GOB prediction scheme with default cost functions. The seventh, and most expensive to compute, is the speculative encoder.

(a) Fixed Power

(b) Truncated Power

(c) Truncated Power, Rate Adaptive

(d) The changes in instantaneous bit rate for the truncated power, rate-adaptive scheme.

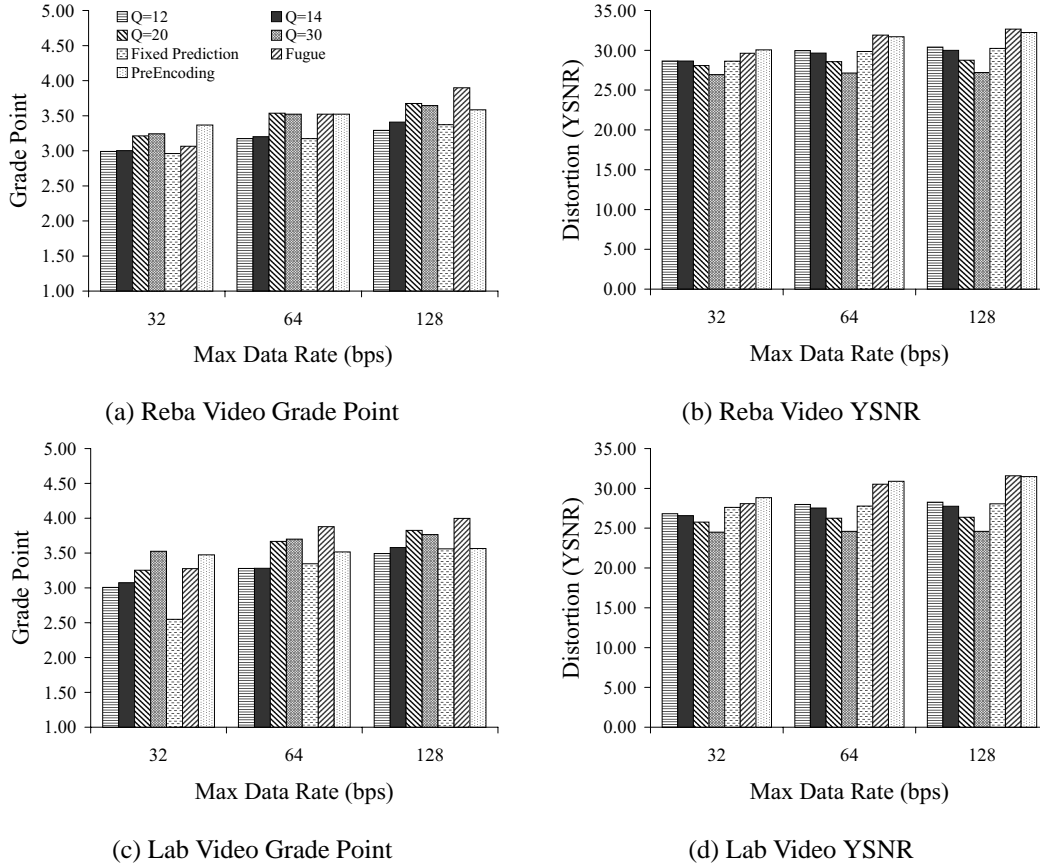**Figure 7.** Transmission Layer Control Schemes

We use these schemes to encode two different video clips, called *reba* and *lab*. These videos are not used in the ratio experiments of Section 5.2, and were generated independently. Encoded at 30 frames per second, with a quantization factor of 14, they require a long-term average bit rates of approximately 42 Kb/s. However, the instantaneous bit rate is very bursty.

We simulate transmission of these videos over three different wireless networks. These networks differ only in the maximum rate they support: 32, 64, and 128 Kb/s. Other simulation parameters for these networks appear in Figure 6. The resulting videos are compared across two metrics. The first is perceptual grade point, as presented in Section 5.2. The second is YSNR. We report YSNR results for two reasons. First, it is the metric that the pre-encoding scheme is attempting to optimize. Second, while it does not directly model perceptual quality, it is the metric most commonly used to compare wireless video systems.

The results are shown in Figure 8. Each bar represents the average of 12 trials. For clarity, we do not present the standard deviations for these results. For the grade point metric they range up to 0.33, and for YSNR, they can be as large as 0.55. However, standard deviations are smaller at higher bit rates, lending more faith to those results. For example, at 128 Kb/s, the largest standard deviations are 0.15 for grade point, and 0.25 for YSNR.

These experiments yield four interesting results. First, the per-frame predictive scheme never compares well to any other scheme along the perceptual quality metric, as shown in Figures 8(a) and 8(c). This is due to the substantial uncertainty in predicting the sizes of encoded GOBs. Frames with significant motion result in very high bit rates, stealing from later frames and potentially reducing the frame rate. This has a very high penalty in the perceptual cost functions. Frames with very little motion leave gaps in the transmission schedule that could have been used to produce better GOBs for that frame.

Second, in very low bit rate environments where the choice of encoding is over-constrained, both the speculative encoder and the static encoders with Q values 20 and 30 outperform our per-GOB predictive scheme in the perceptual quality metric. This is also due to the uncertainty in the predictive model. When the budget is tight, large errors cannot be corrected before the end of the frame time. Therefore, in constrained bit rate environments, one should use a conservative static encoder if

Figure 8. Video Simulation Results

processing power is a concern, and the speculative encoder otherwise.

Third, at higher bit rates, our per-GOB predictive scheme equals or exceeds the speculative scheme along both metrics, but at substantially reduced processing costs. This is somewhat surprising, since the speculative encoder explicitly attempts to optimize for distortion. This discrepancy occurs because the predictive scheme has all potential values of Q at its disposal, while the speculative scheme only has a small number available. If these values are chosen poorly, the speculative scheme cannot adapt over the full useful range of Q.

The final and most surprising result is the disagreement by the two metrics over the ranking of each scheme. The YSNR results imply that it is never correct to use the conservative scheme and always encode at a quantization factor of 30. However, at very low data rates, a user is quite likely to disagree with this conclusion. Distortion is not the right metric to use in comparing video quality if the goal is to deliver the best quality as measured by the human user. This is because it under-values the importance of smooth, frequent motion in overall perceptual quality.

7. CONCLUSION

Providing interactive video on hand-held mobile devices is an extremely difficult problem. There are a number of challenges inherent to the device, its wireless network, and the application itself. While the system can control a number of parameters to address these challenges, it must be structured carefully to avoid unnecessary complexity. Our system, *Fugue*, is structured by separating adaptive capabilities based on the *time scales* over which they are effective. This leads to a three-controller design: transmission, video, and preference.

Fugue's three controllers have modest space and time requirements compared to the basic task of video encoding. Simulations show that Fugue's transmission layer — a truncated power, rate adaptive scheme — effectively controls bit error rates and provides the abstraction of a more stable channel to higher-layer controllers. Experiments with Fugue's video controller show that, in situations where adaptation is useful, it provides the best perceived quality of video at the lowest computational cost. Furthermore, the traditional metric used to evaluate compressed video, distortion, under-values the contribution of motion to perceived video quality.

REFERENCES

1. O. Angin, A.T. Campbell, M.E. Kounavis, and R.R.-F. Liao. The Mobiware toolkit: Programmable support for adaptive mobile net-woking. *IEEE Personal Communications Magazine*, 5(4):32–43, August 1998.
2. P. Bahl. Supporting digital video in a managed wireless network. *IEEE Communications Magazine*, pages 94–102, June 1998.
3. V. Bharghavan, K.-W. Lee, S. Lu, S. Ha, J. R. Li, and D. Dwyer. The TIMELY adaptive resource management architecture. *IEEE Personal Communications Magazine*, 5(4), August 1998.
4. G. Bianchi and A. T. Campbell. A programmable MAC framework for utility-based quality of service support. *IEEE Journal on Selected Areas in Communications*, 18(2):244–255, February 2000.
5. R. H. Clarke. A statistical theory of mobile-radio reception. *Bell Systems Technical Journal*, 47:957–1000, 1968.
6. M. D. Corner, B. D. Noble, and K. M. Wasserman. Fugue: Time scales of adaptation in mobile video. Technical Report CSE-TR-431-00, University of Michigan, Aug 2000.
7. G. Côté, B. Erol, M. Gallant, and F. Kossentini. H.263+: Video coding at low bit rates. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(7), 1998 1998.
8. J. Flinn and M. Satyanarayanan. Energy-aware adaptation for mobile applications. In *Proceedings of the 17th ACM Symposium on Operating Systems and Principles*, Kiawah Island, SC, December 1999.
9. A. Fox, S. D. Gribble, E. A. Brewer, and E. Amir. Adapting to network and client variability via on-demand dynamic distillation. In *Proceedings of the Seventh International ACM Conference on Architectural Support for Programming Languages and Operating Systems*, Cambridge, MA, October 1996.
10. A. J. Goldsmith. The capacity of downlink fading channels with variable rate and power. *IEEE Transactions on Vehicular Technology*, 46(3):569–580, August 1997.
11. M. Gudmundson. Correlation model for shadow fading in mobile radio systems. *Electronic Letters*, 27(23):2145–6, November 1991.
12. ITU-T Recommendation H.261. Video codec for audiovisual services at 64 kbit/s, March 1993.
13. ITU-T Recommendation H.263. Video coding for low bitrate communication, March 1996.
14. C.-Y Hsu, A. Ortega, and M. Khansari. Rate control for robust video transmission over burst-error wireless channels. *IEEE Journal on Selected Areas in Communications*, 17(5):756–773, May 1999.
15. ISO/IEC JTC1/SC29/WG11. Overview of the MPEG-4 standard. MPEG, International Standard, ISO N3342.
16. R. H. Katz. Adaptation and mobility in wireless information systems. *IEEE Personal Communications*, 1(1), 1994.
17. R. H. Katz and E. A. Brewer. The case for wireless overlay networks. In *SPIE Multimedia and Networking Conference*, January 1996.
18. S. W. Kim and Y. H. Lee. Combined rate and power adaptation in DS/CDMA communications over Nakagami fading channels. *IEEE Transactions on Communications*, 48(1):162–168, January 2000.
19. H. Liu and M. El Zarki. Performance of H.263 video transmission over wireless channels using hybrid ARQ. *IEEE Journal on Selected Areas in Communications*, 15(9):1775–1786, December 1997.
20. M. Margaritidis and G.C. Polyzos. MobiWeb: Enabling adaptive continuous media applications over wireless links. In *IEEE International Conference on Third Generation Wireless Communications*, Silicon Valley, San Francisco, California, June 2000.
21. B. D. Noble, M. Satyanarayanan, D. Narayanan, J. E. Tilton, J. Flinn, and K. R. Walker. Agile application-aware adaptation for mobility. In *Proceedings of the Sixteenth ACM Symposium on Operating Systems Principles*, St. Malo, France, October 1997.
22. T. S. Rappaport. *Wireless Communications:Principles and Practice*. Upper Saddle River, New Jersey:Prentice Hall, 1996.
23. K. Rijkse. ITU standardization of very low bitrate video coding algorithms. *Signal Processing: Image Communication*, 7(4-6):553–65, November 1995.
24. M. Satyanarayanan. Mobile information access. *IEEE Personal Communications*, 3(1), February 1996.
25. H. Song and C.-C Jay Kuo. H.263+ rate control via variable frame rates and global bit allocation. In *Visual Communications and Image Processing '98*, pages 372–382, San Jose, CA, January 1998.
26. A. A. Webster, C. T. Jones, M. H. Pinson, S. D. Voran, and S. Wolf. An objective video quality assessment system based on human perception. In *Human Vision, Visual Processing, and Digital Display IV.*, pages 15–26, San Jose, CA, February 1993.
27. T. Wiegand, M. Lightstone, D. Mukherjeee, T. G. Campbell, and S. K. Mitra. Rate-distortion optimized mode selection for very low bit rate video coding and the emerging H.263 standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(2):182–190, April 1996.
28. H. Yong, I. Ahmad, and M. L. Liou. Real-time interactive MPEG-4 system encoder using a cluster of workstations. *IEEE Transactions on Multimedia*, 1(2):217–33, June 1999.