One-Size-Fits-All Wireless Video

Szymon Jakubczak szym@mit.edu
Hariharan Rahul rahul@csail.mit.edu
Dina Katabi dk@mit.edu

Abstract— **We present SoftCast, a new approach for wireless video that flips the conventional design; instead of requiring the source to pick an 802.11 bitrate and video resolution** *before* **transmission, it allows the receiver to decode a video whose bitrate and resolution are commensurate with the observed channel quality** *after* **reception. This approach has applications for multicast and mobile wireless receivers, whose channels differ across time and space.**

I. Introduction

Wireless video is becoming increasingly important, driven by user demand for mobile TV, media sharing, and the broadcast of sporting events, lectures, and promotional clips, in universities, malls, and hotspots [9,11]. These applications however present a significant challenge to conventional wireless design. Consider, for example, streaming mobile TV to users in a university campus. Which of 802.11's bitrates should the source use? If the source transmits at a high bitrate, it reaches only nearby receivers. But, if it transmits at a low bitrate supported by all receivers, it reduces everyone to the performance of the worst receiver interested in the stream. Even with a single receiver, mobility can cause large unpredictable variations in channel SNR [7]. As a result, the source can either pick a conservative choice of bitrate or risk glitches in the received video when the instantaneous channel quality drops below the quality anticipated by the source [4]. The presence of interference further adds to these difficulties as different receivers suffer different loss rates depending on their proximity to the interferer. Adding FEC codes can help the receivers that suffer interference, however these codes consume bitrate available to video data, and hence reduce the video quality for receivers that do not experience interference.

The main reason for the above difficulties is that conventional wireless design assumes the source knows (or can easily measure) the quality of the channel to its receiver. Hence the source can select the best bitrate (i.e., modulation and FEC code) and video code rate for the channel. Multicast, mobility and random interference challenge this assumption and present the source with a channel quality that differs across receivers and varies quickly over time. As a result, the source becomes unable to pick a single bitrate and a video code rate that work well at any time and across all receivers. Ideally, one would like a scheme that does not require the source to know the channel quality, yet achieves the best performance for any channel quality.

This paper introduces SoftCast, a wireless video design that aims to approach this ideal. SoftCast has a new encoding technique that enables the source to broadcast its packets without fixing a bitrate or a video code rate and let each receiver decode a video quality commensurate with its channel quality. Receivers with good channels extract a higher information rate from the transmitted signal and hence obtain a better video quality. Receivers with worse channels extract less information and can watch the transmitted video at a lower quality. This happens naturally despite receiver mobility and interference, and does not require receiver feedback, bitrate adaptation, or varying video code rate.

The key idea underlying SoftCast is to ensure that distances between transmitted codewords are linearly related to differences between pixel values. As a result, a receiver with high SNR (i.e., low noise) receives codewords that are close to the transmitted codewords, and hence decodes pixel values that are close to the original values. It thus recovers an image with high fidelity to the original. A receiver with low SNR (i.e., high noise), on the other hand, receives codewords that are further away from the transmitted codewords, decodes them to pixel values that are further away from the original values, and hence gets a lower fidelity image. Thus, SoftCast provides graceful degradation of the transmitted video for different receivers, depending on the quality of their channel. This is unlike the conventional design, where the transmitted codewords do not preserve the numerical properties of the original pixels, and hence a small perturbation in the received signal, e.g., a bit flip, can cause an arbitrarily large error in pixel luminance.

SoftCast realizes the above idea via two components.

(a) Joint codec for compression and error-protection: The existing wireless design uses a video code for compression and a PHY layer code for error protection. Having a PHY codec that is unaware of the video pixels would prevent SoftCast from achieving its goal of making the distance between the transmitted codewords reflect the difference between pixel values. Thus, SoftCast introduces a new video codec that provides both compression and error protection. SoftCast's codec is linear over the field of real numbers, and hence ensures that differences between codewords are linearly related to the differences between pixel values. The codec compresses the video in a manner that does not force a fixed resolution. It protects from errors by expending more power on transmitting the high-level information in a frame and less power on transmitting the fine details. We prove that, given a hardware transmission power, the resulting codec minimizes video reconstruction errors.

(b) Raw OFDM: Since SoftCast has pushed error protection to the video codec, it does not need error correction codes at the PHY layer. This includes convolutional coding and QAM modulation (a form of coding), which together specify an 802.11 bitrate. Thus, SoftCast adds a switch to the 802.11 PHY layer to bypass these modules and directly transmit SoftCast videos over OFDM.

We evaluate SoftCast using trace-driven experiments performed with the WARP radio platform [20]. We compare SoftCast against two baselines: a single-layer MPEG video and two-layer multi-resolution video. Our preliminary results show that SoftCast can achieve the best of two worlds, that is in scenarios where it is easy to find the best bitrate, (e.g., a single static receiver), SoftCast's video quality is as good as the existing single layer video design and better than multi-layer video. However, when there is no single good bitrate or the choice is unclear, (e.g., fast mobility or multicast), SoftCast delivers a significantly higher video quality than both single and multi-layer video.

II. Related Work

Layered video, scalable video, and multiple resolution coding (MRC), all refer to an encoding technique that fragments a video stream into a base layer and enhancement layers [6,16,17,22,26]. The base layer is necessary for decoding the video stream, whereas the enhancement layers improve its quality. This approach is useful for wired multicast, where a receiver with a congested link can download only the base layer, and avoid packets from other layers. With wireless, all layers share the medium. Thus, the enhancement layers reduce the bandwidth available to the base layer and further reduce the performance of poor receivers. The scheme also assumes the source knows how to choose the best bitrate and video code rate for each layer.

Superposition coding [8] allows transmission to diverse receivers. The source uses the best modulation and code for the weaker receiver, and applies to the resulting signal the modulation and code for the stronger receiver. This allows the stronger receiver to decode more data. The signal of the strong receiver, however, acts like noise for the weaker receiver and reduces its capacity [8]. Also, the source still needs to know the best modulation and code rate for each receiver.

Related work also includes analog and digital TV. Like analog TV, SoftCast exhibits graceful degradation and can support diverse receivers. Unlike analog TV however, SoftCast codes the video to compress it and hence can fit the same quality video in less bandwidth. Also, similarly to digital TV, SoftCast employs OFDM digital transmission to address multipath effects. It also codes the video for compression and error protection. But SoftCast's codec operates over the real field and provides graceful degradation, while digital TV uses more traditional finite field codecs, and hence lacks graceful degradation [10]. We believe the ideas underlying SoftCast could benefit digital TV and plan to explore such extensions in our future work.

Our work also builds on past work in information theory on rate distortion and joint source and channel coding (JSCC) [8]. This work however mainly focuses on theoretical bounds [18,19]. Also the proposed codecs are typically non-linear [24] and significantly harder to implement than SoftCast.

III. SoftCast Overview

The design of SoftCast relies on a simple principle, that is, to ensure that the distance between codewords transmitted by the PHY layer reflects the difference between pixel values, so that a small perturbation on the channel produces a small perturbation in the video. This principle however cannot be achieved within the current wireless design. The conventional design maps real-value video pixels to finite field codewords, i.e., bit sequences, codes them for compression and error protection, and maps them back to real-value digital samples that are transmitted on the channel. The process of mapping to bits however destroys the numerical properties of the original pixels. As a result, small channel errors, e.g., a bit flip, can cause large deviations in the pixel values.

SoftCast realizes the above principle by leveraging the fact that both video and the transmitted digital signal are expressed as real numbers, and hence SoftCast can code the video for compression and error protection directly in the real field. Further, by using a linear codec, the coded values can be made to scale with the original pixel. The output of the codec can then be transmitted directly over OFDM as the I and Q components of the digital signal. Since the transmitted values are linearly related to the original video pixels, the noise in the channel, which perturbs the transmitted signal, translates to corresponding deviations in video pixels. When the transmitted signal is received with higher SNR (i.e., it is less noisy), the video is naturally received at a higher resolution.

IV. SoftCast's Video Encoder

In this paper, we focus on intra-frame video coding, i.e., coding information within a frame. We believe the approach can be extended to code across frames, however such extension is left for future work.

(a) Video Compression: Images are relatively smooth and show spatial correlation. Both MPEG and SoftCast exploit this property to compact the information in a frame by taking a 2-dimensional Discrete Cosine Transform (DCT) of pixel values [25]. MPEG however is designed with the assumption of a known channel, and hence the MPEG

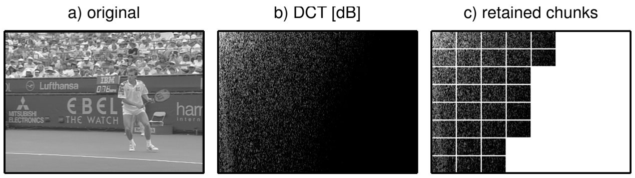


Fig. 1. A full DCT transform of the frame in (a), which is shown in (b), compacts the non-zero DCT values, the white dots, near the top left corner, making it easy to eliminate the zero-value DCTs, i.e., the black region. Further by grouping the DCTs in each square block into one chunk, as in (c), we can efficiently describe the discarded region.

encoder proceeds by quantizing the DCT values to compress the video as much as desired. Quantization discards video information at the transmitter itself, and forces all receivers to the same resolution. In contrast, SoftCast is designed to work with unknown and varying channels, it adopts an approach that is based on: (1) discarding the DCT components that do not contribute to the information in the image, (2) finding such DCT components without an excessive amount of meta data to describe their locations.

SoftCast treats the pixel values in a frame as a two dimensional matrix. It takes a 2-D DCT transform of this matrix. Such full DCT transform redistributes the energy (the information) in a frame to compact it in a few spatially concentrated DCT components, as shown in Fig. 1b. The DCT components in the top left corner refer to low spatial frequencies, i.e., slowly changing gradients, and tend to have large values. The DCT components in the right bottom corner refer to high spatial frequencies, i.e., the fine details in a frame, and have low values, close or equal to zero.

One can compress a frame by discarding the zero (and near-zero) value DCT components. This compression will have almost no impact on the information in a frame. However, it will require the encoder to send a large amount of metadata to the decoder to inform it of the exact location of the discarded DCT components.

To reduce the metadata, SoftCast operates on *chunks*. Specifically, it groups nearby DCT components in one chunk. SoftCast then makes one decision for all DCT components in a chunk, either retaining or discarding them. Fig. 1c shows the chunks that were kept from Fig. 1b. Since zero-value DCT components tend to be concentrated, making one decision for a whole chunk provides a good approximation of the compression quality resulting from discarding individual DCT components. Making one decision per chunk, however, allows SoftCast to reduce the metadata to one bit per chunk. This can be reduced further with run length encoding since discarded chunks tend to be close to each other.

(b) Error Protection: Traditional error protection codes transform the real-valued video data to bit sequences. This process destroys the numerical properties of the original video data and prevents us from achieving our design goal of having the distance between transmitted digital samples scale with the difference between the pixel values. Thus, SoftCast develops a novel approach to error protection that is aligned with its design goal. SoftCast's approach is based on scaling the magnitude of the DCT components in a frame. Scaling the magnitude of a transmitted signal provides resilience to channel noise. To see how, consider a channel that introduces an additive noise of ± 0.1. If a value of 2.5 is transmitted directly over this channel, as the I or Q of a digital sample, it results in a received value of $2.4 - 2.6$. However, if the transmitter scales the value by $10x$, the received signal varies between $24.9 - 25.1$, and hence when scaled down to the original range, the received value becomes $2.51 - 2.49$, and its best approximation given one decimal point is 2.5. However, since the hardware has a fixed power budget, scaling up and therefore expending more power on some signal samples translates to expending less power on other samples. SoftCast's optimization finds the optimal scaling factors that balance this tension.

Again, we operate over chunks, i.e., instead of finding a different scaling factor for each DCT component, we find the optimal scaling factor for each chunk. To do so, we model the values within each chunk as random variables from some distribution. We can have the chunks approximate zero-mean distributions by subtracting the mean of all pixels in the frame before DCT and sending this value as metadata. We compute the variance of each chunk, λ_i. Given these variances, we define an optimization problem that finds the per-chunk scaling factors such that frame reconstruction error is minimized. In the appendix, we show:

Lemma 4.1: Let $x_1 \ldots x_N$ be random variables with zero mean. Let λ_i be the variance of x_i. Assume that we have a total power P, and that the channel is additive white Gaussian noise. The linear encoder that minimizes the mean square reconstruction error is:

$$\begin{aligned} y_i &= g_i x_i, \quad \text{where} \\ g_i &= {\lambda_i}^{-1/4} \left(\sqrt{\frac{P}{\sum_{j=1}^{N} \sqrt{\lambda_j}}} \right). \end{aligned}$$

(c) Notes on the encoder: We note a few points:

- First, the encoder is linear since DCT is linear and our error protection code performs linear scaling.
- Second, the output of the encoder is a series of coded values of the form $y_i[k] = g_i x_i[k]$, where $x_i[k]$ is the k^{th} DCT component in the i^{th} chunk, and g_i is the scaling factor of that chunk. These are packetized and delivered to the PHY (via a raw socket), which transmits them over OFDM, as explained in §VI.
- In addition to the video data, the encoder sends a small amount of metadata to assist the decoder in inverting the received signal. Specifically, the encoder sends the variance of each chunk, λ_i, a bit vector that indicates the discarded chunks, and the average of all pixels in the frame. The scaling factors, i.e., g_i's, can be computed from this information, while DCT is fixed and known. The metadata is compressed and coded for error protection using standard methods and sent at the

lowest 802.11 rate. Though it has to be delivered to all receivers, the overhead is low (0.005 bits/pixel in our implementation).

V. SoftCast's Video Decoder

At the receiver, and as will be described in §VI, the PHY estimates and corrects the attenuation and phase of the received signal. The end result is that for each value $y_i[k]$ that we sent, we receive a value $y_i[k]+n[k]$, where $n[k]$ is a random noise. It is common to assume the noise is additive, white and Gaussian. While this is not exact, it provides good insight and works reasonably well in practice.

The goal of the SoftCast receiver is to decode the received frame in a manner that minimizes the reconstruction errors. We can re-write the received values as

$$\hat{y_i}[k] \quad = \quad g_i x_i[k] + n[k],$$

We want to compute the best estimate of $x_i[k]$ given $\hat{y_i}[k]$. The linear solution to this problem is widely known as the Linear Least Square Estimator (LLSE) [15]. Specifically, if the decoder knows the variance of the source λ_i, and the noise power, σ^2, then the LLSE estimates the original value as:

$$\hat{x_i}[k] = \frac{\lambda_i g_i}{\lambda_i g_i^2 + \sigma^2} \hat{y_i}[k], \tag{1}$$

where $\hat{x_i}[k]$ refers to the LLSE estimate of the k^{th} DCT component in the i^{th} chunk.

Note that the λ_i's are transmitted as metadata by the encoder and are used by the decoder to compute the g_i's. As for noise power σ^2, it is available at the PHY and can be easily exposed as one number per packet.

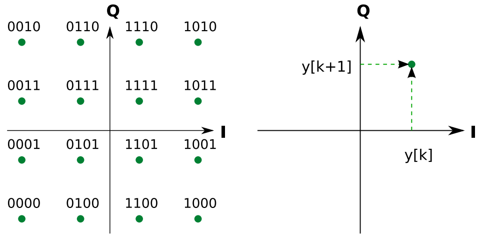
Consider how the LLSE estimator changes with SNR. At high SNR (i.e., small noise, $\sigma^2 \approx 0$), Eq. 1 becomes:

$$\hat{x_i}[k] = \frac{1}{g_i} \hat{y_i}[k]. \tag{2}$$

Thus, at high SNR, the LLSE estimator simply inverts the encoder computation. This is because at high SNR we can trust the measurements and do not need to leverage the statistics of the DCT components within a band, i.e., λ_i. In contrast, at low SNR, when the noise power is high, one cannot fully trust the measurements and hence it is better to re-adjust the estimate according to the statistics of the DCT components in a chunk.

Next, the decoder inverts the DCT to obtain the pixels in the frame. To do so, it tiles the decoded DCT components back and substitutes the DCT values that were discarded at the encoder by zero. The decoder applies the inverse DCT to generate the original video frame.

How does a receiver decode if it has packet loss? When a packet is lost, the decoder substitutes the lost DCT components by computing the corresponding DCT components from the previous frame. We note however that packet loss in SoftCast should be rare. Specifically, in contrast to conventional 802.11, where a packet is lost if it has any bit errors, SoftCast accepts all packets. Thus, packet loss occurs only when the hardware fails to detect the presence of a packet, e.g., a hidden terminal scenario.



(a) 16-QAM (b) SoftCast

Fig. 2. Mapping coded video to I/Q components of transmitted signal.

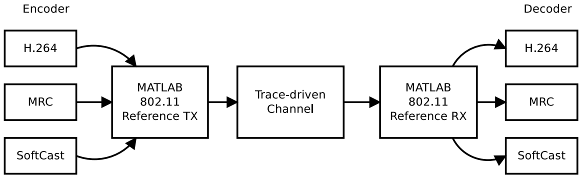


Fig. 3. Testing setup.

VI. SoftCast's PHY Layer

Traditionally, the PHY layer takes a stream of bits and codes them for error protection. It then modulates the bits to produce real-value digital samples that are transmitted on the channel. For example, 16-QAM modulation takes sequences of 4 bits and maps each such sequence to a complex number as shown in Fig. 2a. The real and imaginary parts of these complex numbers produce the real-value I and Q components of the transmitted signal.[1] In contrast to existing wireless design, SoftCast's codec outputs real values that are already coded for error protection. Thus, we can directly map pairs of SoftCast coded values to the I and Q digital signal components, as shown in Fig. 2b.

To integrate this design into the existing 802.11 PHY layer, we leverage the fact that OFDM separates channel estimation and tracking from data transmission [13]. Specifically, OFDM divides the 802.11 spectrum into many independent subcarriers, some of which are called pilots and used for channel tracking, and the others are left for data transmission. SoftCast does not modify the pilots or the 802.11 header symbols, and hence does not affect traditional OFDM functions of synchronization, carrier frequency offset (CFO) estimation, channel estimation, and phase tracking. SoftCast simply transmits in each of the OFDM data bins, as illustrated in Fig 2a. Such a design can be integrated into the existing 802.11 PHY simply by adding an option to allow the data to bypass FEC and QAM, and use raw OFDM.

VII. Evaluation Environment

We have implemented a prototype of SoftCast, which we evaluate in comparison with MPEG-4 and MRC.

Reference Baselines: For reference, we used the H.264/MPEG-4 AVC [21] codec implemented in x264 [2]. We select MPEG-TS as the stream format, which is the

[1] The PHY performs the usual FFT/IFFT and normalization operations on the I/Q values, but these preserve linearity.

common choice for streaming. Since at this stage our SoftCast design focuses on intra-frame coding, we enable all features of the codec, but for inter-frame coding. We also use x264 to implement a multiresolution coding (MRC) scheme that encodes the video into a base layer and an enhancement layer, based on the *SNR scalable profile* method described in [12].

Testing Setup: Our testing setup is shown in Fig. 3 and is based on trace-driven experiments. To obtain channel characteristics, we extract noise patterns from empirical measurements collected with the WARP radio platform [20]. The measurements span SNRs from 4 to 25 dB, which is the operational range of 802.11 OFDM. All schemes receive the same power (captured by SNR), and use the same wireless bandwidth of 2 MHz.[2] The 802.11 OFDM PHY (convolutional codes, modulation, and OFDM) is implemented using MATLAB's 802.11 reference provided in the communications toolbox [1], which we augment to send SoftCast's videos directly over raw OFDM.

Metric: We compare the schemes using the Peak Signal-to-Noise Ratio (PSNR). It is a standard measure of video/image quality [23] and is defined as a function of the mean squared error (MSE) between all pixels of the decoded video and the original version as follows:

$$PSNR = 10\ \log_{10}\frac{2^L-1}{MSE}\quad [dB],$$

where L is the number of bits used to encode pixel luminance, typically 8 bits. A PSNR below 20 dB refers to *bad* video quality [27], and differences of 1 dB or higher are visible [23].

Test Videos: Performance of codecs varies from one video to another. So, we create one monochrome 480-frame SIF (30 fps) test video by splicing 1 second from each of 16 popular reference videos from the Xiph [3] collection: *akiyo, bus, coastguard, crew, flower, football, foreman, harbour, husky, ice, news, soccer, stefan, tempete, tennis, waterfall*.

VIII. Results

(a) Benchmark Results. First, we study the implications of forcing the source to pick an 802.11 bitrate on the performance of MPEG, and compare it with SoftCast, where a source need not pick a bitrate. We experiment with MPEG for different choices of bitrates under various channel SNRs. For each choice of bitrate, we allow MPEG to pick the optimal video code rate supported by that bitrate. We also experiment with SoftCast for the same range of receiver SNRs. We repeat each run 5 times and report in Fig. 4 the median performance along with the minimum and maximum.

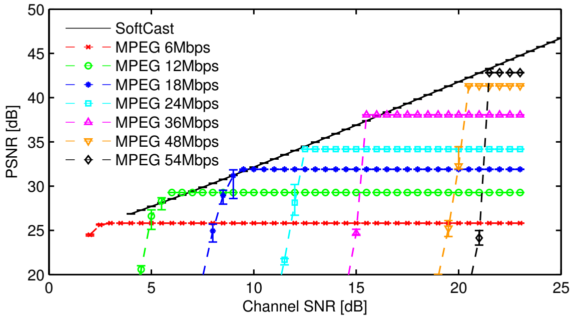


Fig. 4. **Receiver PSNR vs. Channel SNR.** Dashed curves correspond to MPEG-4 over different 802.11 bitrates. For any bitrate, there is a critical SNR, below which the MPEG video is unrecoverable, and above which the quality stays fixed despite improved channel quality. In contrast, SoftCast's video PSNR scales smoothly with channel SNR. Furthermore, it stays at the envelope of MPEG performance.

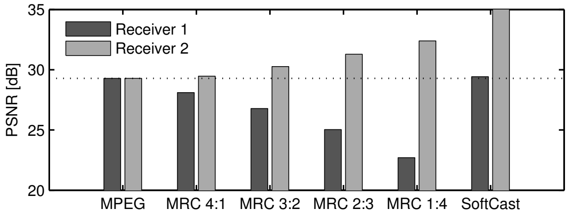


Fig. 5. **Multicast to Two Receivers.** The figure shows the video PSNR are both receiver for various video options at the source. It shows that MRC improves the performance of the stronger receiver at the cost of reducing the performance of the weaker receiver. In contrast, SoftCast can benefit the stronger receiver without hampering the weaker receiver.

The figure confirms the cliff effect characteristic of current wireless video [14]. Specifically, for each 802.11 bitrate, there exists a critical SNR below which the bit-error-rate increases sharply leading to irrecoverable video; conversely, above the critical SNR, the data is delivered virtually error-free but the PSNR is limited by the compression loss introduced at the MPEG encoder. In contrast, SoftCast's video PSNR scales smoothly with the channel SNR. Further, its video quality matches the envelope of MPEG quality, which shows that this smooth behavior is obtained without jeopardizing good PSNR at fixed and known channel SNR.

(b) Multicast. We run a simple multicast experiment with two receivers whose SNRs are 7 and 14 dB, and their optimal bitrates are 12 Mb/s and 24 Mb/s, respectively. We experiment with different alternatives for multicasting the video. If the source uses MPEG, it can at best transmit at 12 Mb/s. Alternatively, the source can use a two-layer MRC, and transmit the base layer at 12 Mb/s and the enhancement layer at 24 Mb/s. Since the two layers share the wireless medium, the source has to decide how to divide medium access between the them. We consider various such allocations. Finally, the source could use SoftCast in which case it does not need to pick a bitrate nor divide medium access between layers. Fig. 5 shows the PSNR of the two receivers given these options.

The figure shows that in comparison to one-layer MPEG, MRC has to make a trade-off: The higher the fraction of

[2] 802.11 bandwidth is 20 MHz, but we limit the video to 2 MHz to accommodate other flows on the channel and also because typical test videos [3] have a low code rate even at their maximum resolution.

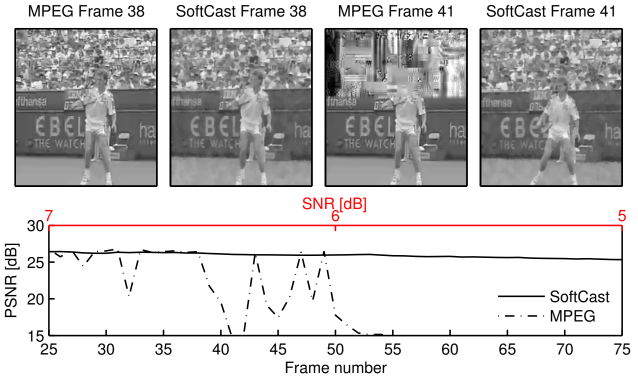


Fig. 6. **Mobility.** Small fluctuations of 1-2 dB in channel SNR can drastically affect MPEG. In contrast, SoftCast's performance is more robust to variations in SNRs and hence it can potentially help in mobile scenarios. The bottom x-axis shows the frame id, while the top x-axis shows the frame SNR. Note the sudden change in MPEG's PSNR between frames 38 and 41, though the channel SNR changed by less than 0.4 dB.

medium time devoted to the enhancement layer, the better the performance of the stronger receiver, but the worse the performance of the weaker receiver. This is because the two layers share the wireless medium, and hence whatever resources are allocated to a layer that the weak receiver cannot decode reduce its overall performance. In contrast, SoftCast does not divide the resources between layers or receivers, and hence can provide the stronger receiver with a higher PSNR without hampering the performance of the weaker receiver.

(b) Mobility. In this experiment, a receiver moves away from its source causing a relatively small change in channel SNR of about 1-2 dB. We plot in Fig. 6 the corresponding change in video PSNR for both SoftCast and the best MPEG. The figure shows that a small variation in channel SNR of less than 1 dB, can drastically change the PSNR of an MPEG video. This supports the earlier results in Fig. 4 which show that MPEG's degradation is quite sudden. On the other hand, rate adaptation protocols require some time lag to collect receiver feedback, and typically do not adapt within a fraction of a dB difference of SNR [5]. In comparison, SoftCast is much more stable, and hence could improve performance for mobile receivers.

References

[1] MATLAB IEEE 802.11a WLAN model. MATLAB Communications Toolbox.

[2] *x264 - a free H.264/AVC encoder*. `http://www.videolan.org/developers/x264.html`.

[3] Xiph.org media. `http://media.xiph.org/video/derf/`.

[4] G. Bai and C. Williamson. The Effects of Mobility on Wireless Media Streaming Performance. In *Proceedings of Wireless Networks and Emerging Technologies*, pages 596–601, july 2004.

[5] J. Bicket. Bit-rate selection in wireless. *M.S. Thesis*, 2005.

[6] J. Byers, M. Luby, and M. Mitzenmacher. A digital fountain approach to asynchronous reliable multicast. *IEEE JSAC*, 20(8):1528–1540, Oct 2002.

[7] J. Camp and E. W. Knightly. Modulation rate adaptation in urban and vehicular environments: cross-layer implementation and experimental evaluation. In *MOBICOM*, 2008.

[8] T. Cover and J. Thomas. *Elements of Information Theory*. 1991.

[9] M. Etoh and T. Yoshimura. Advances in wireless video delivery. *Proc. of the IEEE*, 93(1):111–122, Jan. 2005.

[10] G. Faria, J. Henriksson, E. Stare, and P. Talmola. DVB-H: Digital broadcast services to handheld devices. *Proc. of the IEEE*, 94(1):194–209, 2006.

[11] L. Hanzo, P. Cherrimana, and J. Streit. *Wireless Video Communications: Second to Third Gen and Beyond*. John Wiley, 2001.

[12] B. G. Haskell, A. Puri, and A. N. Netravali. *Digital Video: An Introduction to MPEG-2*. Springer, 1997.

[13] J. Heiskala and J. Terry. *OFDM Wireless LANs: A Theoretical and Practical Guide*. Sams Publishing, 2001.

[14] T. Kratochvíl and R. Štukavec. DVB-T Digital Terrestrial Television Transmission over Fading Channels. *Radioengineering*, 17(3):97, 2008.

[15] C. Lawson and R. Hanson. *Solving Least Squares Problems*. Society for Industrial Mathematics, 1987.

[16] A. Majumdar et al. Multicast and unicast real-time video streaming over wireless lans. *IEEE Trans. Circuits and Systems for Video Technology*, 12(6):524–534, 2002.

[17] S. McCanne, M. Vetterli, and V. Jacobson. Low-complexity video coding for receiver-driven layered multicast. *IEEE JSAC*, 15(6):983–1001, Aug 1997.

[18] U. Mittal and N. Phamdo. Hybrid digital-analog (hda) joint source-channel codes for broadcasting and robust communications. *IEEE Trans. Information Theory*, 48(5):1082–1102, May 2002.

[19] Z. Reznic, M. Feder, and R. Zamir. Distortion bounds for broadcasting with bandwidth expansion. *IEEE Trans. Information Theory*, 52(8):3778–3788, Aug. 2006.

[20] Rice University. Rice University Wireless Open-Access Research Platform. WARP, `http://warp.rice.edu/trac`.

[21] I. Richardson. *H. 264 and MPEG-4 video compression: video coding for next-gen multimedia*. John Wiley & Sons Inc, 2003.

[22] A. Said and W. Pearlman. An image multiresolution representation for lossless and lossy compression. *IEEE Trans. Image Processing*, 5(9):1303–1310, Sep 1996.

[23] D. Salomon. *Guide to Data Compression Methods*. Springer, 2002.

[24] M. Skoglund, N. Phamdo, and F. Alajaji. Hybrid digital-analog source-channel coding for bandwidth compression/expansion. *IEEE Trans. Information Theory*, 52(8):3757–3763, 2006.

[25] A. Watson. Image compression using the discrete cosine transform. *Mathematica Journal*, 4:81–88, Jan. 1994.

[26] D. Wu, Y. Hou, and Y.-Q. Zhang. Scalable video coding and transport over broadband wireless networks. *Proc. of the IEEE*, 89(1):6–20, 2001.

[27] O. Zigmund. Network-driven adaptive video streaming in wireless environments. In *IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications*, 2008.

Appendix

For each value in chunk i, x_i, we transmit $y_i = g_i x_i$, and the receiver receives $\hat{y}_i = y_i + n$, where g_i is the scaling factor for this chunk, and n is a random noise with zero-mean and variance σ^2 (the same for all chunks). Subsequently, the receiver decodes $\hat{x}_i = \frac{\hat{y}_i}{g_i} = x_i + \frac{n}{g_i}$. The expected mean square error is:

$$err = E\left[\sum_i (\hat{x}_i - x_i)^2\right] = \sum_i \frac{E[n^2]}{g_i^2} = \sum_i \frac{\sigma^2}{g_i^2}$$

Let $\lambda_i = E[x_i^2]$ be the power of chunk x_i, $\mu_i = E[y_i^2]$ be its power after applying the gain, and P the total power budget. The problem is:

$$\min err = \sigma^2 \sum_i \frac{\lambda_i}{\mu_i}, \quad \text{subject to} \sum_i \mu_i \leq P \quad \text{and} \quad \mu_i \geq 0. \tag{3}$$

We can solve this optimization by taking the Lagrangian:

$$L = \sigma^2 \sum_i \frac{\lambda_i}{\mu_i} + \gamma \left(\sum_i \mu_i - P\right).$$

Differentiating separately by μ_i and γ and setting to 0, yields: $\sqrt{\gamma} = \sum_j \sqrt{\lambda_j \sigma^2}/P$ and $\mu_i = \sqrt{\frac{\lambda_i \sigma^2}{\gamma}} = P\frac{\sqrt{\lambda_i}}{\sum_j \sqrt{\lambda_j}}$ and hence:

$$g_i = \sqrt{\frac{\mu_i}{\lambda_i}} = \sqrt{\frac{P}{\sqrt{\lambda_i}\sum_j \sqrt{\lambda_j}}} \tag{4}$$