SoftCast: One Size Fits All Wireless Video

Szymon Jakubczak Dina Katabi
szym@mit.edu dk@mit.edu

***Abstract*— Video broadcast and mobile video challenge the conventional wireless design. In broadcast and mobile scenarios the bit rate supported by the channel differs across receivers and varies quickly over time. The conventional design however forces the source to pick a single bit rate and degrades sharply when the channel cannot not support the chosen bit rate.**

This paper presents SoftCast, a clean-slate design for wireless video where the source transmits one video stream that each receiver decodes to a video quality commensurate with its specific instantaneous channel quality. To do so, SoftCast ensures the samples of the digital video signal transmitted on the channel are *linearly* related to the pixels' luminance. Thus, when channel noise perturbs the transmitted signal samples, the perturbation naturally translates into approximation in the original video pixels. Hence, a receiver with a good channel (low noise) obtains a high fidelity video, and a receiver with a bad channel (high noise) obtains a low fidelity video.

We implement SoftCast using the GNURadio software and the USRP platform. Results from a 20-node testbed show that SoftCast improves the average video quality (i.e., PSNR) across broadcast receivers in our testbed by up to 5.5 dB. Even for a single receiver, it eliminates video glitches caused by mobility and increases robustness to packet loss by an order of magnitude.

I. Introduction

The conventional wireless design decomposes the problem of video transmission into two sub-problems: encoding the video for compression, then encoding the compressed data to protect it from errors during transmission over the wireless channel. Shannon's separation theorem tells us that separating source coding (i.e., video compression) from channel coding (i.e., error protection) can be done without loss of optimality if the channel is *point-to-point* (one sender receiver pair) and its statistics are *known* to the source [14,37,40]. For practical video transmission this means that if the source transmits to one receiver and the channel quality to that receiver is known or can be easily measured at the source, the source can select the optimal transmission bit rate for the channel and the corresponding forward error correction code (FEC) and modulation scheme. Once the transmission bit rate is determined, the video codec (typically MPEG) can compress the video so that it can be streamed at the chosen bit rate. This separate design is entirely appropriate for many scenarios, which involve a single sender-receiver pair that communicates over a relatively static channel whose characteristics vary slowly over time.

Consider, however, a scenario involving video multicast to mobile receivers. This scenario invalidates the two assumptions underlying the conventional design. The channel is no longer point-to-point: it is a broadcast channel where each receiver observes a different channel quality. The channel characteristics are no longer easy to predict at the source: the quality of the channel to each receiver can change quickly over time as the receiver moves [4,41]. With the conditions of the separation theorem unsatisfied, the conventional design is no longer efficient. In this scenario, the conventional design has to pessimistically choose the transmission bit rate supported by the worst receiver. It also has to code the video at a low quality to fit within the chosen low transmission bit rate. The drawback of this approach is that receivers with better quality channels can obtain only the video quality of the receiver with the worst quality channel.

This paper presents SoftCast, a clean-slate end-to-end architecture for transmitting video over wireless channels. In contrast to the separate conventional design, SoftCast adopts a unified design that both encodes the video for compression and for error protection. Our end-to-end approach enables us to deliver multicast video to multiple mobile receivers, with each receiver obtaining video quality commensurate with its specific instantaneous channel quality.

SoftCast starts with video that is represented as a sequence of numbers, with each number representing a pixel luminance. Taking an end-to-end perspective, it then performs a sequence of transformations to obtain the final signal samples that are transmitted on the channel. The crucial property of SoftCast is that each transformation is linear. This property ensures that the signal samples transmitted on the channel are linearly related to the original pixel values. Thus, increasing channel noise progressively perturbs the transmitted bits in proportion to their significance for the video application, i.e., high-quality channels perturb only the least significant bits while low-quality channels still preserve the most significant bits. Each receiver therefore decodes the received signal into video whose quality is proportional to the quality of its specific instantaneous channel.

SoftCast's end-to-end architecture has the following four linear components:

(1) Compression: Traditional video compression is designed in separation from the wireless channel. Hence, though the wireless channel has a high error rate, traditional compression uses Huffman and differential encoding which are highly sensitive to errors.[1] In contrast, SoftCast compresses a video by applying a three-dimensional decorrelation transform, such as 3D DCT. Using 3D DCT (as opposed to the 2D DCT used in MPEG), allows SoftCast to remove redundant information within a frame as well as across frames. Further, since DCT is linear, errors on the channel do not lead to disproportionate errors in the video.

(2) Error Protection: Traditional error protection codes may map values that are numerically far apart, *e.g.*, 2.5 and

[1] Huffman is a variable length code and hence a bit error can cause the receiver to confuse symbol boundaries. Differential encoding and motion compensation encode frames with respect to other frames and hence any error in a reference frame percolates to other correctly received frames.

0.3, to adjacent codewords, say, 01001000 and 01001001, causing a single bit flip to produce a dramatic change in the rendered video. In contrast, SoftCast's error protection is based on scaling the magnitude of the transmitted coded samples. Consider a channel that introduces an additive noise in the range ± 0.1. If a value of 2.5 is transmitted directly over this channel, it results in a received value in the range $[2.4 - 2.6]$. However, if the transmitter scales the value 10 times, the received signal varies between 24.9 and 25.1, and hence when scaled down to the original range, the received value is in the range $[2.51 - 2.49]$, and its best approximation given one decimal point is 2.5, which is the correct value. SoftCast has a built in optimization that identifies the proper scaling that minimizes video error subject to a given transmission power.

(3) Resilience to Packet Loss: Current video codecs employ differential encoding and motion compensation. These techniques create dependence between transmitted packets. As a result, the loss of one packet may cause subsequent correctly received packets to become undecodable. In contrast, SoftCast ensures that all packets contribute equally to the quality of the decoded video. Specifically, SoftCast employs a Hadamard transform [2] to distribute the video information across packets such that each packet has approximately the same amount of information.

(4) Transmission over OFDM: Modern wireless technologies (802.11, WiMax, Digital TV, etc.) use an OFDM-based physical layer (PHY). SoftCast is integrated within the existing PHY layer by making OFDM transmit SoftCast's encoded data as the I and Q components of the digital signal.

SoftCast builds on prior work on video multicast over channels with varying quality. The state of the art approaches to this problem still use a separate design. These schemes use a layered approach in which the video is encoded into a low-quality base layer (which all receivers must correctly decode to obtain any video at all) and a few higher-quality enhancement layers (which receivers with higher-quality channels can decode to obtain higher-quality video). In the limit, as the number of layers becomes very large, a layered approach would ideally deliver to each receiver a video quality proportional to its channel quality. In practice, however, encoding video into layers incurs an overhead that accumulates with more layers [43]. Thus, practical layered schemes (such as those proposed for Digital TV) use only two layers [8,12,21]. In contrast to a layered approach, a SoftCast sender produces a single video stream, with the video quality at each receiver determined by the significance of the bits that its channel delivers without distortion. The quality of the video degrades smoothly at the granularity of the individual luminance bits, rather than at the much coarser granularity of the number of layers in the transmitted video. SoftCast also builds on a growing literature in information theory tackles joint source and channel coding (JSCC) [28,33,38]. SoftCast's design is motivated by the same philosophy but differs in its emphasis on linear transforms. Furthermore, past work on JSCC is mainly theoretical and is not tested over an actual wireless channel.

We have implemented SoftCast and evaluated it in a testbed of 20 GNURadio USRP nodes [13,39]. We compare SoftCast with two baselines: 1) MPEG-4 (i.e., H.264/AVC) over 802.11, and 2) layered video where the layers are encoded using the scalable video extension to H.264 (SVC) and transmitted using hierarchical modulation as in [21]. We evaluate these schemes using the Peak Signal-to-Noise Ratio (PSNR), a standard metric of video quality [26,35]. We have the following findings:

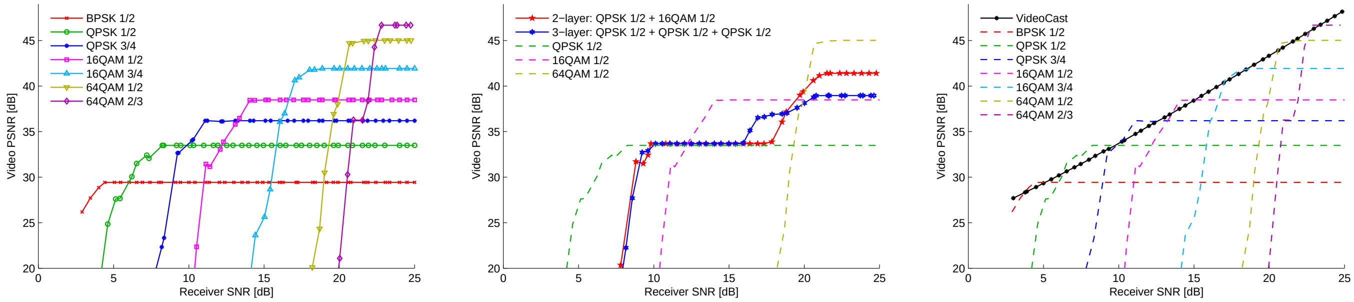
- SoftCast delivers to each multicast receiver a video quality that is proportional to its channel quality and is competitive (within 1 dB) with the optimal quality the receiver could obtain if it were the only receiver in the multicast group.
- For multicast receivers of SNRs in the range $[5, 25]$ dB, SoftCast improves the average video quality by 5.5 dB over the best performer of the two baselines.
- Even with a single mobile receiver, SoftCast eliminates video glitches, whereas 14% of the frames in our mobility experiments suffer glitches with the best performer of the two baselines.
- Finally, SoftCast tolerates an order of magnitude higher packet loss rates than both baselines.

A. Graphical Comparison

Fig. 1 graphically displays the characteristics of the different video encoding and transmission schemes. This figure presents three graphs; each graph plots the video quality at the receiver as a function of the channel quality. All schemes use exactly the same transmission power and the same channel bandwidth over the same period of time, i.e., they are exposed to the same channel capacity and differences are due only to how effectively they use that capacity. The measurements are collected using GNURadio USRP nodes. For more details on the experimental setup see §VIII.

Fig. 1(a) illustrates the realizable space of video qualities for conventional MPEG-based approaches. Each line refers to a particular choice of transmission bit rate, i.e., a particular choice of forward error correction code and a modulation scheme. The video codec encodes the video at the same rate as the channel transmission bit rate. Fig. 1(a) shows that for any selection of transmission bit rate (i.e., modulation and FEC) the conventional design experiences a performance cliff, that is there is a critical SNR, below which the video is not watchable, and above that SNR the video quality does not improve with improvements in channel quality.

Fig. 1(b) illustrates the video qualities obtained by state of the art layered video coding. The video is encoded using the JSVM reference implementation for scalable video coding (SVC) [19]. The physical layer transmits the video using hierarchical modulation over OFDM, an inner convolutional



(a) Conventional Design **(b) Layered SVC with Hierarchical Modulation** **(c) SoftCast**

Fig. 1. **Approaches to Wireless Video:** Fig. (a) plots the space of video qualities obtained with the conventional design which uses MPEG4 over 802.11. Each line refers to a choice of transmission bit rate (i.e., modulation and FEC). Fig. (a) shows that for any choice of bit rate, the conventional design experiences a performance cliff. Fig. (b) plots 2-layer video in red and 3-layer video in blue. For reference, the dashed lines are the three equivalent single-layer MPEG4 videos. The figure shows that layered video makes the cliffs milder but each new layer introduces overhead and reduces the maximum video quality. Fig. (c) shows SoftCast (in black) and single-layer MPEG4. It shows that SoftCast video quality fully scales with channel quality.

code and an outer Reed-Solomon code following the recommendations in [8]. The figure shows two solid lines, the red line encodes the video into two layers while the blue line encodes the video into three layers. For reference, the figure also shows in dashed lines the single layer MPEG4 videos that span the range of channel SNRs spanned by the layers in the layered video. The figure shows that layered video transforms the performance cliff of the conventional design to a few milder cliffs. Layering however causes extra overhead [43] and thus increases the size of the video. Given a particular bit rate budget, the video codec has to reduce the quality of the layered video in comparison with the single layer video to ensure that the videos have the same size and can be streamed at the same bit rate. As a result, the enhancement layer of the 3-layer video has a lower quality than the corresponding layer in 2-layer video, which has a lower quality than the corresponding single layer video.

Fig. 1(c) illustrates the video qualities obtained with SoftCast. The figure shows that SoftCast's video quality is proportional to the channel quality and stays competitive with the envelope of all of MPEG curves.

B. *Contributions*

This paper makes the following contributions.

- It presents SoftCast, a novel design for wireless video, where the sender need not know the wireless channel quality or adapt to it. Still, the sender can broadcast a video stream that each receiver decodes to a video whose quality is commensurate with its channel quality. This happens without receiver feedback, bit rate adaption, or video code rate adaptation.
- Unlike existing video approaches where some packets are more important than others, in SoftCast all packets are equally important for the reconstruction of the video, which significantly increases resilience to packet loss.
- The paper presents an implementation and an empirical evaluation of SoftCast in a 20-node testbed of software radios. It shows that the protocol significantly improves robustness to mobility and packet loss and provides a better quality video multicast.

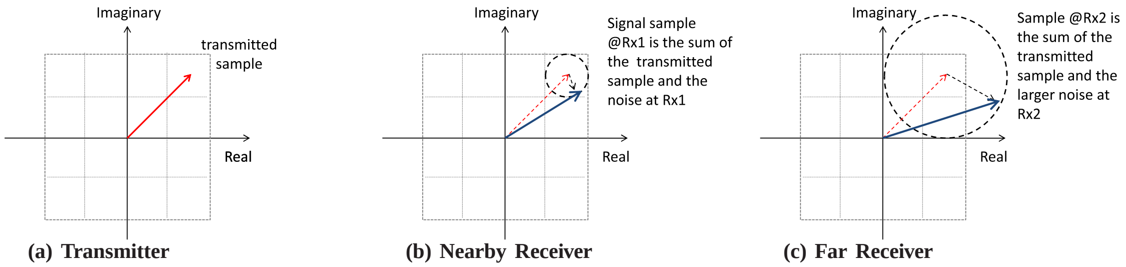
II. Why does the Conventional Design Not Allow One-Size-Fits-All Video?

Today's approach to compression and error protection coding prevents existing wireless design from providing one-size-fits-all video.

(a) Compression: Video pixels are highly correlated within a frame. Further, video frames are correlated in time. MPEG exploits this correlation by operating on sequences of successive video frames called GoPs (Group of Pictures). MPEG compresses a video in two steps [11]. First, it performs intra-frame compression to remove redundant information within each frame. This is done by applying a 2-dimensional DCT on small blocks of 8x8 pixels, and quantizing the resulting DCT components to a fixed precision. The conventional design then treats these quantized real values as sequences of bits and compresses them to a compact bit sequence using a Huffman code. Second, MPEG performs inter-frame compression to eliminate redundant information across frames in a GoP. In particular, it uses differential encoding, which compares a frame against a prior reference frame and only encodes the differences. It also uses motion compensation to predict the movement of a particular block across time. Using this combination MPEG achieves good compression ratios. However, it is this combination that prevents one-size-fits-all video:

- Quantization is performed by the source, and coarsens the resolution of the video to match a desired bitrate, and hence fixes the quality of the video, even if the receiver channel could support a higher bitrate.
- Huffman coding and differential encoding fail sharply in the presence of bit errors and packet losses. Specifically, a Huffman code is variable length, and a few bit flips can cause the receiver to confuse symbol boundaries, making the whole frame unrecoverable. Differential encoding and motion compensation create dependencies between different packets in a coded video, and hence the loss of some packets can prevent the decoding of correctly received video packets.

Note that layered and scalable video coding (SVC) also use


Imaginary
transmitted sample
Real
Imaginary
Signal sample @Rx1 is the sum of the transmitted sample and the noise at Rx1
Real
Imaginary
Sample @Rx2 is the sum of the transmitted sample and the larger noise at Rx2
Real

(a) Transmitter **(b) Nearby Receiver** **(c) Far Receiver**

Fig. 2. **Wireless broadcast delivers more signal bits to low noise receivers.** The figure shows the transmitted sample in red, the received samples in blue, and noise in black. The source transmits the signal sample in (a). A nearby receiver experiences less noise and can estimate the transmitted sample up to the small square, i.e., up to 4 bits. A far receiver sees more noise and hence knows only the quadrant of the transmitted sample, i.e., it knows only 2 bits of the transmitted sample.

quantization, variable-length coding, differential encoding and motion compensation, and hence are also highly sensitive to wireless errors.

(a) Error Protection: Error protection is typically done at the physical layer (PHY) by picking a bitrate, i.e., a combination of modulation and forward error correcting code, that ensures the receiver can decode the vast majority of the packets correctly. The packet decoding probability drops sharply when the bitrate chosen is higher than can be supported by the channel SNR [31], and hence the PHY layer is constrained to pick a low modulation and code rate that works well across time and receivers.

III. SoftCast Overview

SoftCast's design harnesses the intrinsic characteristics of both wireless broadcast and video. The wireless physical layer (PHY) transmits complex numbers that represent modulated signal samples, as shown in Fig. 2(a). Because of the broadcast nature of the wireless medium, multiple receivers hear the transmitted signal samples, but with different noise levels. For example, in Fig. 2, the receiver with low noise can distinguish which of the 16 small squares the original sample belongs to, and hence can correctly decode the 4 most significant bits of the transmitted sample. The receiver with higher noise can distinguish only the quadrant of the transmitted signal sample, and hence can decode only the two most significant bits of the transmitted sample. Thus, wireless broadcast naturally delivers to each receiver a number of signal bits that match its SNR.

Video is watchable at different qualities. Further, a video codec encodes video at different qualities by changing the quantization level [11], that is by discarding the least significant bits. Thus, to scale video quality with the wireless channel's quality, all we need to do is to map the least significant bits in the video to the least significant bits in the transmitted samples. Hence, SoftCast's design is based on a simple principle: ensure that the transmitted signal samples are linearly related to the original pixel values.

The above principle cannot be achieved within the conventional wireless design. In the conventional design, the video codec and the PHY are oblivious to each other. The codec maps real-value video pixels to bit sequences, which lack the numerical properties of the original pixels. The PHY maps these bits back to pairs of real values, i.e., complex samples, which have no numerical relation to the original pixel values. As a result, small channel errors, e.g., errors in the least significant bit of the signal sample, can cause large deviations in the pixel values.

In contrast, SoftCast introduces a clean-slate joint video-PHY architecture. SoftCast both compresses the video, like a video codec would do, and encodes the signal to protect it from channel errors and packet loss, like a PHY layer would do. The key characteristic of the SoftCast encoder is that it uses only linear codes for both compression and error and loss protection. This ensures that the final coded samples are linearly related to the original pixels. The output of the encoder is then delivered to the driver over a special socket to be transmitted directly over OFDM.

IV. SoftCast's Encoder

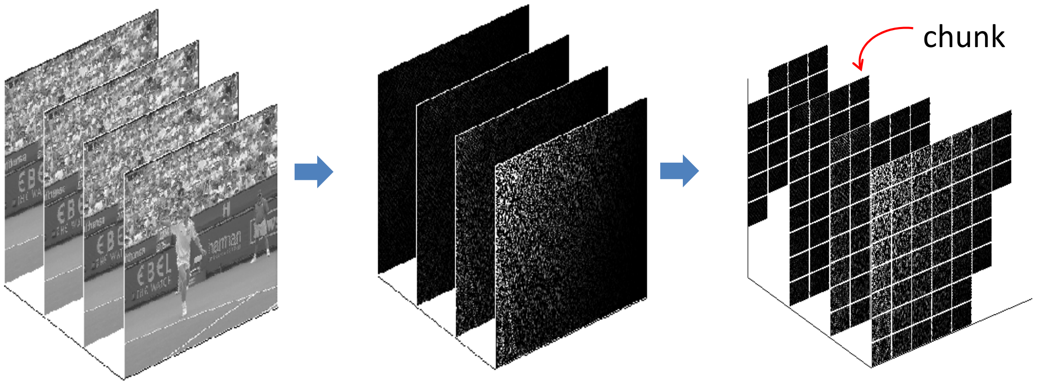
SoftCast's encoder both compresses the video and encodes it for error and loss protection.

A. Video Compression

Both MPEG and SoftCast exploit spatial and temporal correlation in a GoP to compact information. Unlike MPEG, however, SoftCast takes a unified approach to intra and inter-frame compression, i.e., it uses the same method to compress information across space and time. Specifically, SoftCast treats the pixel values in a GoP as a 3-dimensional matrix. It takes a 3-dimensional DCT transform of this matrix. The DCT transforms the data to its frequency representation. Since frames are correlated, their frequency representation is highly compact.

Fig. 3 shows a GoP of 4 frames, before and after taking a 3D DCT transform. The grey level after the 3D DCT reflects the magnitude of the DCT component in that location. The figure shows two important properties of 3D DCT:

(1) Most DCT components have a zero (black) value, i.e., have no information. This is because frames tend to be smooth [42], and hence the high spatial frequencies tend to be zero. Further, most of the structure in a video stays constant across multiple frames [11], and hence most of the higher temporal frequencies tend to be zero. This



(a) 4-frame GoP (b) 3D DCT of GoP (c) Discarding Zero-Valued Chunks

Fig. 3. **3D DCT of a 4-frame GoP.** The figure shows (a) a 4-frame GoP, (b) its 3D DCT, where each plane has a constant temporal frequency, and the values within a plane represent spatial frequencies at that temporal frequency, (c) the non-zero DCT components in each plane grouped into chunks. The figure shows that most DCT components are zero (black dots) and hence can be discarded. Further, the non-zero DCT components are clustered together.

means that one can discard all of these zero-valued DCT components without affecting the quality of the video.

(2) Non-zero DCT components are spatially clustered. This is because spatially nearby DCT components represent nearby spatial frequencies, and natural images exhibit smooth variation across spatial frequencies. This means that one can express the locations of the retained DCT components with little information by referring to clusters of DCT components rather than individual components.

SoftCast exploits these two properties to efficiently compress the data by transmitting only the non-zero DCT components. This compression is very efficient and has no impact on the energy in a frame. However, it requires the encoder to send a large amount of metadata to the decoder to inform it of the locations of the discarded DCT components.

To reduce the metadata, SoftCast groups nearby spatial DCT components into *chunks*, as shown in Fig. 3c. The default chunk in our implementation is 44x30x1 pixels, (where 44×30 is chosen based on the SIF video format where each frame is 352×240 pixels). Note that SoftCast does not group temporal DCT components because typically only a few structures in a frame move with time, and hence most temporal components are zero, as in Fig. 3c. SoftCast then makes one decision for all DCT components in a chunk, either retaining or discarding them. The clustering property of DCT components allows SoftCast to make one decision per chunk without compromising the compression it can achieve. As before, the SoftCast encoder still needs to inform the decoder of the locations of the non-zero chunks, but this overhead is significantly smaller since each chunk represents many DCT components (the default is 1320 components/chunk). SoftCast sends this location information as a bitmap. Again, due to clustering, the bitmap has long runs of consecutive retained chunks, and hence can be efficiently compressed using run-length encoding.

The previous discussion assumed that the source has enough bandwidth to transmit all the non-zero chunks over the wireless medium. What if the source is bandwidth constrained? It will then have to judiciously select non-zero chunks so that the transmitted stream can fit in the available bandwidth, and still be reconstructed with the highest quality. SoftCast selects the transmitted chunks so as to minimize the reconstruction error at the decoder:

$$err = \sum_i (\sum_j (x_i[j] - \hat{x}_i[j])^2), \qquad (1)$$

where $x_i[j]$ is the original value for the j^{th} DCT component in the i^{th} chunk, and $\hat{x}_i[j]$ is the corresponding estimate at the decoder. When a chunk is discarded, the decoder estimates all DCT components in that chunk as zero. Hence, the error from discarding a chunk is merely the sum of the squares of the DCT components of that chunk. Thus, to minimize the error, SoftCast sorts the chunks in decreasing order of their energy (the sum of the squares of the DCT components), and picks as many chunks as possible to fill the bandwidth.

Note that bandwidth is a property of the source, (e.g., a 802.11 channel has a bandwidth of 20 MHz) independent of receiver, whereas SNR is a property of the receiver and its channel. As a result, discarding non-zero chunks to fit the source bandwidth does not prevent each receiver from getting a video quality commensurate with its SNR.

Two points are worth noting about the used compression.

- SoftCast can capture correlations across frames while avoiding motion compensation and differential encoding. It does this because it performs a 3D DCT, as compared to the 2-D DCT performed by MPEG. The ability of the 3D DCT to compact energy across time is apparent from Fig. 3b where the values of the temporal DCT components die quickly (i.e., in Figs. 3b, the planes in the back are mostly black).
- The main computation performed by SoftCast's compression is the 3D DCT, which is $O(K\log(K))$, where K is the number of pixels in a GoP. A variety of efficient DCT implementations exist both in hardware and software [10,29].
- Finally, it is possible to replace 3D DCT with other 3D decorrelation transforms, such as 3D Wavelets [48].

> We have experimented with both 3D DCT and 3D Wavelets and found them to be comparable, with 3D DCT showing better clustering of non-zero components.

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 transmitted digital samples scale linearly with the 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 in the range ± 0.1. If a value of 2.5 is transmitted directly over this channel, (e.g., as the I or Q of a digital sample), it results in a received value in the range $[2.4 - 2.6]$. However, if the transmitter scales the value by $10x$, the received signal varies between 24.9 and 25.1, and hence when scaled down to the original range, the received value is in the range $[2.51 - 2.49]$, and its best approximation given one decimal point is 2.5, which is the correct value. 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 a single optimal scaling factor for all the DCT components in each chunk. To do so, we model the values $x_i[j]$ within each chunk i as random variables from some distribution $\mathcal{D}_i$. We remove the mean from each chunk to get zero-mean distributions and send the means as metadata. Given the mean, the amount of information in each chunk is captured by its variance. We compute the variance of each chunk, λ_i, and define an optimization problem that finds the per-chunk scaling factors such that GoP reconstruction error is minimized. In the appendix, we show:

Lemma 4.1: Let $x_i[j], j = 1 \ldots N$, be random variables drawn from a distribution $\mathcal{D}_i$ with zero mean, and variance λ_i. Given a number of such distributions, $i = 1 \ldots C$, a total transmission power P, and an additive white Gaussian noise channel, the linear encoder that minimizes the mean square reconstruction error is:

$$\begin{aligned} u_i[j] &= g_i x_i[j], \quad \text{where} \\ g_i &= {\lambda_i}^{-1/4}\left(\sqrt{\frac{P}{\sum_i \sqrt{\lambda_i}}}\right). \end{aligned}$$

Note that there is only one scaling factor g_i for every distribution $\mathcal{D}_i$, i.e., one scaling factor per chunk. The output of the encoder is a series of coded values, $u_i[j]$, as defined above. Further, the encoder is linear since DCT is linear and our error protection code performs linear scaling.

C. *Resilience to Packet Loss*

Next, we assign the coded DCT values to packets. However, as we do so, we want to maximize SoftCast's resilience to packet loss. Current video design is fragile to packet loss because it employs differential encoding and motion compensation. These schemes create dependence between packets, and hence the loss of one packet can cause subsequent correctly received packets to become undecodable. In contrast, SoftCast's approach ensures that all packets equally important. Hence, there are no special packets whose loss causes disproportionate video distortion.

A naive approach to packetization would assign chunks to packets. The problem, however, is that chunks are not equal. Chunks differ widely in their energy (which is the sum of the squares of the DCT components in the chunk). Chunks with higher energy are more important for video reconstruction, as evident from equation 1. Hence, assigning chunks directly to packets causes some packets to be more important than others.

SoftCast addresses this issue by transforming the chunks into equal-energy *slices*. Each SoftCast slice is a linear combination of all chunks. SoftCast produces these slices by multiplying the chunks with the Hadamard matrix, which is typically used in communication systems to redistribute energy [2,25]. The Hadamard matrix is an orthogonal transform composed entirely of +1s and -1s. Multiplying by this matrix creates a new representation where the energy of each chunk is smeared across all slices.[2]

We can now assign slices to packets. Note that, a slice has the same size as a chunk, and depending on the chosen chunk size, a slice might fit within a packet, or require multiple packets. Regardless, the resulting packets will have equal energy, and hence offer better packet loss protection.

The packets are delivered directly to the PHY (via a raw socket), which interprets their data directly as the digital signal samples to be sent on the medium, as described in §VI.

D. *Metadata*

In addition to the video data above, the encoder sends a small amount of metadata to assist the decoder in inverting the received signal. Specifically, the encoder sends the mean and the variance of each chunk, and a bitmap that indicates the discarded chunks. The decoder can compute the scaling factors, i.e., g_i's, from this information. As for the Hadamard and DCT matrices, they are well known and do not need to be transmitted. The bitmap of chunks is compressed using run length encoding as described in §IV-A, and all metadata is further compressed using Huffman coding. The total metadata in our implementation after adding a Reed-Solomon code is 0.014 bits/pixel, i.e., its overhead is insignificant.

The metadata has to be delivered correctly to all receivers. To protect the metadata from channel errors, we send it using BPSK modulation and half rate convolutional code,

[2] Hadamard multiplication has an additional benefit which is to whiten the signal reducing the peak to average power ratio (PAPR).

which are the modulation and FEC code corresponding to the lowest 802.11 bit rate. To ensure that the probability of losing metadata because of packet loss is very low, we spread the metadata across all packets in a GoP. Thus, each of SoftCast's packets starts with a standard 802.11 header followed by the metadata then the coded video data. (Note that different OFDM symbols in a packet can use different modulation and FEC code. Hence, we can send the metadata and the SoftCast video data in the same packet.) To further protect the metadata we encode it with a Reed-Solomon code that can tolerate a loss rate up to 50%. The code uses a symbol size of one byte, a block size of 1024, and a redundancy factor of 50%. Thus, even with 50% packet erasure, we can still recover the metadata fully correctly. This is a high redundancy code but since the metadata is very small, we can afford a code that doubles its size.

E. The Encoder: A Matrix View

We can compactly represent the encoding process of a GoP as matrix operations. Specifically, we represent the DCT components in a GoP as a matrix X where each row is a chunk. We can also represent the final output of the encoder as a matrix Y where each row is a slice. The encoding process can then be represented as

$$\begin{aligned} Y &= HGX && (2) \\ &= CX && (3) \end{aligned}$$

where G is a diagonal matrix with the scaling factors, g_i, as the entries along the diagonal, H is the Hadamard matrix, and $C = HG$ is simply the encoding matrix.

V. SoftCast's Video Decoder

At the receiver, and as will be described in §VI, for each received packet, the PHY returns the list of coded DCT values in that packet (and the metadata). The end result is that for each value $y_i[j]$ that we sent, we receive a value $\hat{y}_i[j] = y_i[j] + n_i[j]$, where $n_i[j]$ is random noise from the channel. It is common to assume the noise is additive, white and Gaussian. While this is not exact, it works reasonably well in practice.

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

$$\hat{Y} = CX + N,$$

where $\hat{Y}$ is the matrix of received values, C is the encoding matrix from Eq. 2, X is the matrix of DCT components, and N is a matrix where each entry is white Gaussian channel noise.

Without loss of generality, we can assume that the slice size is small enough that a slice fits within a packet, and hence each row in $\hat{Y}$ is contained in a single packet. If the slice size is larger than the packet size, then each slice consists of more than one packet, say, K packets. The decoder simply needs to repeat its algorithm K times. In the i^{th} iteration ($i = 1 \ldots K$), the decoder constructs a new $\hat{Y}$ where the rows consist of the i^{th} packet from each slice.[3] For the rest of our exposition, therefore, we will assume that each packet contains a full slice.

The receiver knows the received values, $\hat{Y}$, and can construct the encoding matrix C from the metadata. It then needs to compute its best estimate of the original DCT components, X. The linear solution to this problem is widely known as the Linear Least Square Estimator (LLSE) [22]. The LLSE provides a high-quality estimate of the DCT components by leveraging knowledge of the statistics of the DCT components, as well as the statistics of the channel noise as follows:

$$X_{LLSE} = \Lambda_x C^T (C \Lambda_x C^T + \Sigma)^{-1} \hat{Y}, \qquad (4)$$

where:

- X_{LLSE} refers to the LLSE estimate of the DCT components.
- C^T is the transpose of the encoder matrix C.
- Σ is a diagonal matrix where the i^{th} diagonal element is set to the channel noise power experienced by the packet carrying the i^{th} row of $\hat{Y}$. The PHY has an estimate of the noise power in each packet, and can expose it to the higher layer.
- Λ_x is a diagonal matrix whose diagonal elements are the variances, λ_i, of the individual chunks. Note that the λ_i's are transmitted as metadata by the encoder.

Consider how the LLSE estimator changes with SNR. At high SNR (i.e., small noise, the entries in Σ approach 0), Eq. 4 becomes:

$$X_{LLSE} \approx C^{-1} Y \qquad (5)$$

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, Λ, of the DCT components. 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.

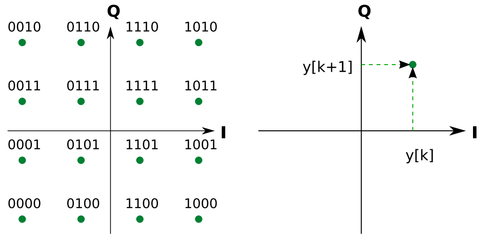
Once the decoder has obtained the DCT components in a GoP, it can reconstruct the original frames by taking the inverse of the 3D DCT.

A. Decoding in the Presence of Packet Loss

We note that, 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., in a hidden terminal scenario.

Still, what if a receiver experiences packet loss? When a packet is lost, SoftCast can match it to a slice using the sequence numbers of received packets. Hence the loss of a packet corresponds to the absence of a row in Y. Define Y_{*i} as Y after removing the i^{th} row, and similarly C_{*i} and N_{*i}

[3] Since matrix multiplication occurs column by column, we can decompose our matrix $\hat{Y}$ into strips which we operate on independently.



(a) 16-QAM (b) SoftCast

Fig. 4. Mapping coded video to I/Q components of transmitted signal. For example, to transmit the bit sequence 1010, the traditional PHY maps it to the complex number corresponding to the point labeled 1010. In contrast, SoftCast's PHY treats pairs of coded values as the real and imaginary parts of a complex number.

as the encoder matrix and the noise vector after removing the i^{th} row. Effectively:

$$\hat{Y}_{*i} = C_{*i}X + N_{*i}. \tag{6}$$

The LLSE decoder becomes:

$$X_{LLSE} = \Lambda_x C_{*i}^T (C_{*i} \Lambda_x C_{*i}^T + \Sigma_{(*i,*i)})^{-1} \hat{Y}_{*i}. \tag{7}$$

Note that we remove a row and a column from Σ. Eq. 7 gives the best approximation of Y when a single packet is lost. The same approach extends to any number of lost packets. Thus, SoftCast's approximation degrades gradually as receivers lose more packets, and, unlike MPEG, there are no special packets whose loss prevents decoding.

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. 4a. The real and imaginary parts of these complex numbers produce the real-valued I and Q components of the transmitted signal.[4]

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. 4b.[5]

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 [15]. As a result, it allows us to change how the data is coded and modulated without affecting the OFDM behavior. 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.

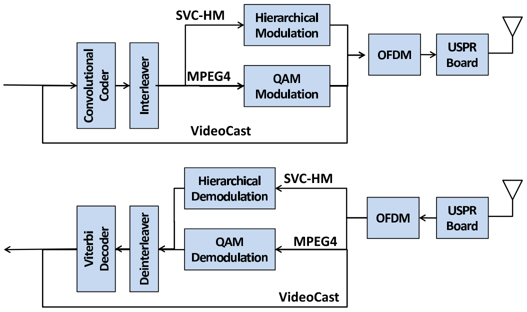


Fig. 5. Block diagram of our PHY implementation. The top graph shows the transmitter side the bottom graph shows the receiver.

SoftCast simply transmits in each of the OFDM data bins, as illustrated in Fig 4a. 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. Streaming media applications can choose the raw OFDM option, while file transfer applications continue to use standard OFDM.

VII. Implementation

We use the GNURadio codebase [13] to build a prototype of SoftCast and an evaluation infrastructure to compare it against two baselines:

- MPEG4 (i.e., H.264) over an 802.11 PHY.
- Layered video where the video is coded using the scalable video extension (SVC) of H.264/AVC [19] and is transmitted over hierarchical modulation [8]. This approach has been proposed in [18] to extend Digital TV to mobile handheld devices.

The Physical Layer. Since both baselines and SoftCast use OFDM, we built a shared physical layer that allows the execution to branch depending on the evaluated video scheme. Our PHY implementation leverages the OFDM implementation in the GNU Radio codebase, with minor modifications that better approximate OFDM as used in 802.11. Specifically, we have augmented the GNU Radio OFDM codebase to incorporate pilot subcarriers and phase tracking, which are standard components in OFDM receivers [15]. We also developed software modules that perform 802.11 interleaving, convolutional coding, and Viterbi decoding.

Fig. 5 shows a block diagram of the implemented PHY layer. On the transmit side, the PHY passes SoftCast's packets directly to OFDM, whereas MPEG4 and SVC-encoded packets are subject to convolutional coding and interleaving, where the code rate depends on the chosen bit rate. MPEG4 packets are then passed to the QAM modulator while SVC-HM packets are passed to the hierarchical modulation module. The last step involves OFDM transmission and is common to all schemes. On the receive side, the signal is passed to the OFDM module which performs carrier frequency offset (CFO) estimation and correction, channel estimation and correction, and phase tracking. The receiver then inverts the execution branches at the transmitter.

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

[5] An alternative way to think about SoftCast is that it is fairly similar to the modulation in 802.11 which uses 4QAM, 16QAM, or 64QAM, except that SoftCast uses a very dense 64K QAM.

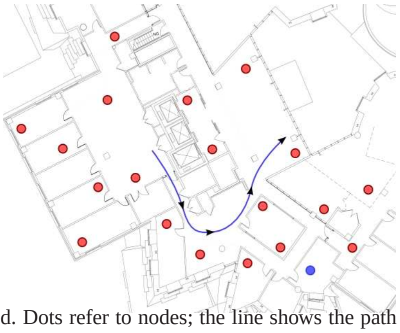


Fig. 6. Testbed. Dots refer to nodes; the line shows the path of the receiver in the mobility experiment when the blue dot was the transmitter.

Video Coding. We implemented SoftCast in Python (with SciPy). For the baselines, we used reference implementation available online. Specifically, we generate MPEG-4 streams using the H.264/AVC [17,34] codec provided in open source FFmpeg software and the x264 codec library [9,45]. We generate the SVC stream using the JSVM implementation [19], which allows us to control the number of layers. Also for MPEG4 and SVC-HM we add an outer Reed-Solomon code for error protection with the same parameters as used for digital TV [8]. All the schemes: MPEG4, SVC-HM, and SoftCast use a GoP of 16 frames.

VIII. Evaluation Environment

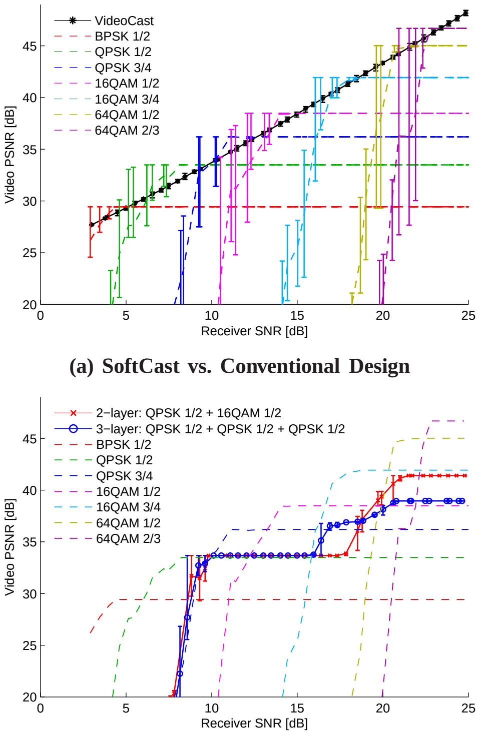
Testbed: We run our experiments in the 20-node GNURadio testbed shown in Fig. 6. Each node is a laptop connected to a USRP2 radio board [39]. We use the RFX2400 daughterboards which operate in the 2.4 GHz range.

Modulation. The conventional design represented by MPEG4 over 802.11 uses the standard modulation and FEC, i.e., BPSK, QPSK, 16QAM, 64QAM and 1/2, 2/3, and 3/4 FEC code rates. The hierarchical modulation scheme uses QPSK for the base layer and 16QAM for the enhancement layer as recommended in [21]. It is allowed to control how to divide transmission power between the layers to achieve the best performance [21]. The three layer video uses QPSK at each level of the QAM hierarchy and also controls power allocation between layers. SoftCast is transmitted directly over OFDM. The OFDM parameters are selected to match those of 802.11a/g.

The Wireless Environment. The carrier frequency is 2.4 GHz which is the same as that of 802.11b/g. The channel bandwidth after decimation is 1.25 MHz. Since the USRP radios operate in the same frequency band as 802.11 WLANs, there is unavoidable interference. To limit the impact of interference, we run our experiments at night. We repeat each experiment five times and interleave runs of the three compared schemes.

Metric: We compare the schemes using the Peak Signal-to-Noise Ratio (PSNR). It is a standard metric for video quality [35] and is defined as a function of the mean squared error (MSE) between all pixels of the decoded video and the original as follows:

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



(a) SoftCast vs. Conventional Design

(b) SVC-HM vs. Conventional Design

Fig. 7. **Basic benchmark.** The figure shows average video quality as a function of channel quality. The bars show differences between the maximum and minimum quality, which are large around cliff points. The top graph compares SoftCast (black line) against the conventional design of MPEG4 over 802.11 (dashed lines) for different choices of 802.11 modulation and FEC code rate. The bottom graph compares layered video (red and blue lines) against the conventional design.

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, and differences of 1 dB or higher are visible [35].

Test Videos: We use standard reference videos in the SIF format (352×240 pixels, 30 fps) from the Xiph [47] collection. Since codec performance varies from one video to another, we create one monochrome 480-frame test video by splicing 1 second from each of 16 popular reference videos: *akiyo, bus, coastguard, crew, flower, football, foreman, harbour, husky, ice, news, soccer, stefan, tempete, tennis, waterfall.*

Other Parameters: The packet length is 14 OFDM symbols or 250 bytes when using 16QAM with 1/2 FEC rate. The transmission power is 100mW. The channel bandwidth is 1.25 MHz. Note that all experiments in this paper use the same transmission power and the same channel bandwidth. Thus, the compared schemes are given the same channel capacity[6] and differences in their throughput and their streaming quality are due only to how effectively they use that capacity.

[6] Shannon capacity is $C = W\log(1+\frac{HP}{WN})$ where W is the bandwidth, P is the power, H is the channel function, and N is the noise power per Hz.

IX. Results

We empirically evaluate SoftCast and compare it against: 1) the conventional design, which uses MPEG4 over 802.11 and 2) SVC-HM, a state of the art layered video design that employs the scalable video extension of H.264 and a hierarchical modulation PHY layer [21,36].

A. Benchmark Results

We first revisit the result in §I-A, which we reproduce in Fig. 7 for convenience.

Method: In this experiment, we pick a node randomly in our testbed, and make it broadcast the video using the conventional design, SoftCast, and SVC-HM. We run MPEG4 over 802.11 for all 802.11 choices of modulation and FEC code rates. We also run SVC-HM for the case of 2-layer and 3-layer video. During the video broadcast, all nodes other than the sender act as receivers. [7] For each receiver, we compute the average SNR of its channel and the PSNR of its received video. To plot the video PSNR as a function of channel SNR, we divide the SNR range into bins of 0.5 dB each, and take the average PSNR across all receivers whose channel SNR falls in the same bin. This produces one point in Fig. 7. We use this procedure to produce points for all lines in the figure. We repeat the experiment by randomly picking the video source from the nodes in the testbed.

Results: Fig. 7 shows that for any choice of 802.11 modulation and FEC code rate, there exists a critical SNR below which the conventional design degrades sharply, and above it the video quality does not improve with channel quality. In contrast, SoftCast's PSNR scales smoothly with the channel SNR. Further, SoftCast's PSNR matches the envelope of the conventional design curves at each SNR. The combination of these two observations means that SoftCast can significantly improve video performance for mobile and multicast receivers while maintaining the efficiency of the existing design for the case of a single static receiver.

It is worth noting that this does not imply that SoftCast outperforms MPEG4. MPEG4 is a compression scheme that compresses video effectively, whereas SoftCast is a wireless video transmission architecture. The inefficacy of the MPEG4-over-802.11 lines in Fig. 7a stems from the fact that the conventional design separates video coding from channel coding. The video codec (MPEG and its variants) assumes an error-free lossless channel with a specific transmission bit rate, and given these assumptions, it effectively compresses the video. However, the problem is that in scenarios with multiple or mobile receivers, the wireless PHY cannot present an error-free lossless channel to all receivers and at all times without reducing everyone to a conservative choice of modulation and FEC and hence a low bit rate and a corresponding low video quality.

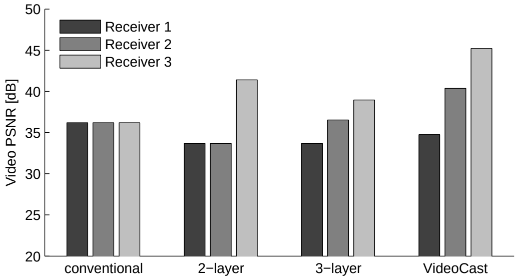


Fig. 8. **Multicast to three receivers.** The figure shows that layering provides service differentiation between receivers as opposed to single layer MPEG4. But layering incurs overhead at the PHY and the codec, and hence extra layers reduce the maximum achievable video quality. In contrast, SoftCast provides service differentiation while achieving a higher overall video quality.

Fig. 7b shows that a layered approach based on SVC-HM exhibits milder cliffs than the conventional design and can provide quality differentiation. However, layering reduces the overall performance in comparison with conventional single layer MPEG4. Layering incurs overhead both at the PHY and the video codec. At any fixed PSNR in Fig. 7b, layered video needs a higher SNR than the single layer approach to achieve the same PSNR. This is because in hierarchical modulation, every higher layer is noise for the lower layers. Similarly, at any fixed SNR, the quality of the layered video is lower than the quality of the single layer video at that SNR. This is because layering imposes additional constraints on the codec and reduces its compression efficiency [43].

B. Multicast

Method. We pick a single sender and three multicast receivers from the set of nodes in our testbed. The receivers' SNRs are 11 dB, 17 dB, and 22 dB. In the conventional design, the source uses the modulation scheme and FEC that correspond to 12 Mb/s 802.11 bit rate (i.e., QPSK with 1/2 FEC code rate) as this is the highest bit rate supported by all three multicast receivers. In 2-layer SVC-HM, the source transmits the base layer using QPSK and the enhancement layer using 16 QAM, and protects both with a half rate FEC code. In 3-layer SVC-HM, the source transmits each layer using QPSK, and uses a half rate FEC code.

Results: Fig. 8 shows the PSNR of the three multicast receivers. The figure shows that, in the conventional design, the video PSNR for all receivers is limited by the receiver with the worse channel. In contrast, 2-layer and 3-layer SVC-HM provide different performance to the receivers. However, layered video has to make a trade-off: The more the layers the more performance differentiation but the higher the overhead and the worse the overall video PSNR. SoftCast does not incur a layering overhead and hence can provide each receiver with a video quality that scales with its channel quality, while maintaining a higher overall PSNR.

Method: Next, we focus on how the diversity of channel SNR in a multicast group affects video quality. We create 40 different multicast groups by picking a random sender and

[7] We decode the received video packets offline because the GNUradio Viterbi decoder can not keep up with packet reception rate.

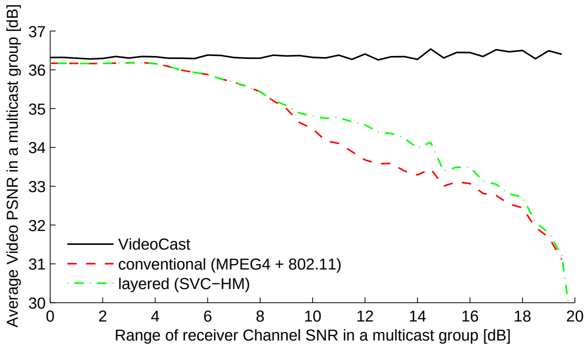


Fig. 9. **Serving a multicast group with diverse receivers.** The figure plots the average PSNR across receivers in a multicast group as a function of the SNR range in the group. The figure shows that the conventional design and SVC-HM provide a significantly lower average video quality than SoftCast for multicast group with a large SNR span.

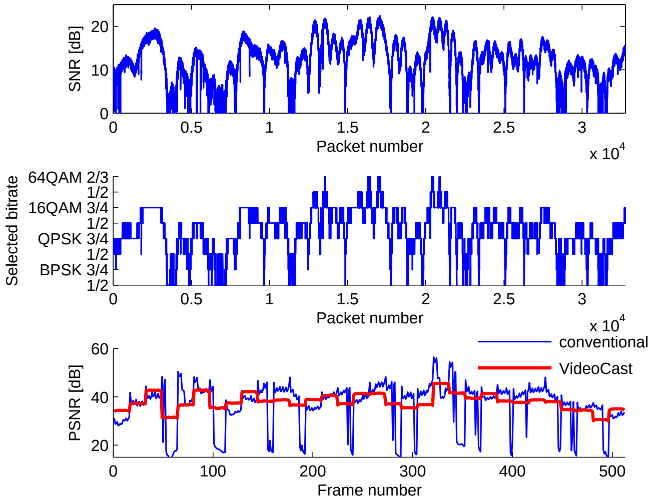


Fig. 10. **Mobility.** The figure compares the video quality of the conventional design and SoftCast under mobility. The conventional design is allowed to adapt its bitrate and video code rate. The top graph shows the SNR of the received packets, the middle graph shows the transmission bit rate chosen by SoftRate and used in the conventional design. The bottom graph plots the per frame PSNR. The figure shows that even with rate adaptation, a mobile receiver still suffers significant glitches with the conventional design. In contrast, SoftCast can eliminate these glitches.

different subsets of receivers in the testbed. Each multicast group is parametrized by its SNR span, i.e., the range of its receivers' SNRs. We keep the average SNR of all multicast groups at 15 (± 1) dB. We vary the range of the SNRs in the group from 0-20 dB by picking the nodes in the multicast group. Each multicast group has up to 15 receivers, with multicast groups with zero SNR range having only one receiver. For each group, we run each of the three compared schemes. The transmission parameters for each scheme (i.e., modulation and FEC rate) is such that provides the highest bit rate and average video quality without starving any receiver in the group. Finally, SVC-HM is allowed to pick for each group whether to use one layer, two layers, or three layers.

Results. Fig. 9 plots the average PSNR in a multicast group as a function of the range of its receiver SNRs. It shows that SoftCast delivers a PSNR gain of up to 5.5 dB over both the conventional design and SVC-HM. One may be surprised that the PSNR improvement from layering is small. Looking back, Fig. 8b shows that layered video does not necessarily improve the average PSNR in a multicast group. It rather changes the set of realizable PSNRs from the case of a single layer where all receivers obtain the same PSNR to a more diverse PSNR set, where receivers with better channels can obtain higher video PSNRs.

C. Mobility

Next, we study video glitches experienced by a single mobile receiver. Since a video PSNR below 20 dB is not watchable [26], we identify glitches as frames whose PSNR is below 20 dB.

Method: Performance under mobility is sensitive to the exact movement patterns. Since it is not possible to repeat the exact movements across experiments with different schemes, we follow a trace-driven approach like the one used in [41]. Specifically, we perform the mobility experiment with non-video packets. We then subtract the received soft values from the transmitted soft values to extract the noise pattern on the channel. This noise pattern contains all necessary information to describe the distortion that occurred on the channel including fading, interference, the effect of movement, etc. We then apply the same noise pattern to each of the three video transmission schemes to emulate its transmission on the channel. This allows us to compare the performance of the three schemes under the same conditions. Fig. 6 shows the path followed during the mobility experiments.

We allow the conventional design to adapt its transmission bit rate and video code rate. To adapt the bit rate we use SoftRate [41], which is particularly designed for mobile channels. To adapt the video code rate, we allow MPEG4 to switch the video coding rate at GoP boundaries to match the transmission bit rate used by SoftRate. Adapting the video faster than every GoP is difficult because frames in a GoP are coded with respect to each other. We also allow the conventional design to retransmit lost packets with the maximum retransmission count set to 11. We do not adapt the bit rate or video code rate of layered video. This is because a layered approach should naturally work without adaptation. Specifically, when the channel is bad, the hierarchical modulation at the PHY should still decode the lower layer, and the video codec should also continue to decode the base layer. Finally, SoftCast is not allowed to adapt its bit rate or its video code rate nor is it allowed to retransmit lost packets.

Results: Fig. 10 shows the results of our experiment. The top graph shows the SNR in the individual packets in the mobility trace. Fig 10b shows the transmission bit rates picked by SoftRate and used in the conventional design. Fig 10c shows the per-frame PSNR for the conventional design and SoftCast. The results for SVC-HM are not plotted because SVC-HM failed to decode almost all frames (80% of GoP were not decodable). This is because layering alone, and particularly hierarchical modulation at the PHY, could not handle the high variability of the mobile channel. Recall

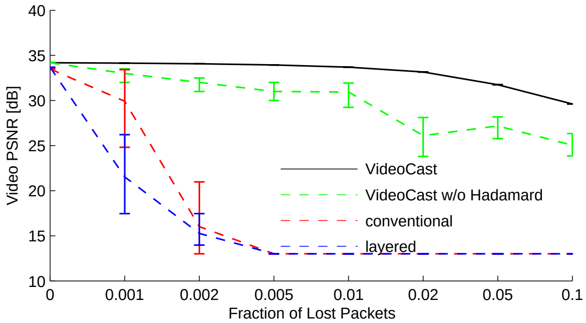


Fig. 11. **Resilience to packet loss.** The figure shows that both SVC-HM and the conventional MPEG-based design suffer dramatically at a packet loss rate as low as 0.5%. In contrast, SoftCast's is only mildly affected even when the loss rate is as high as 10%. For reference, the figure shows the performance of SoftCast if it did not use the Hadamard matrix to ensure that all packets carry equal amount of information.

that in hierarchical modulation, the enhancement layers are effectively noise during the decoding of the base layer, making the base layer highly fragile to SNR dips. As a result, the PHY is not able to protect the base layer from losses. In contrast single layer video reacted better to SNR variability because its PHY can adapt to use BPSK which is the most robust among the various modulation schemes.

Fig 10c shows that, with mobility, the conventional wireless design based on MPEG-4 experiences significant glitches in video quality. These glitches happen when a drop in the transmission bit rate causes significant packet losses such that even if the packets are recovered with retransmissions, they might still prevent timely decoding of the video frames. In comparison, SoftCast's performance is stable even in the presence of mobility. This is mainly due to SoftCast being highly robust to packet loss due to that it avoids Huffman and differential encoding and it spreads the video information across all packets. The results in Fig 10c show that, in this mobile experiment, 14% of the frames transmitted using the conventional design suffer from glitches. SoftCast however has eliminated all such glitches.

D. *Resilience to Packet Loss*

Method: We pick a random pair of nodes from the testbed and transmit video between them. We generate packet loss by making an interferer transmit at constant intervals. By controlling the interferer's transmission rate we can control the packet loss rate. We compare four schemes: the conventional design based on MPEG4, 2-layer SVC-HM, full-fledged SoftCast, and SoftCast after disabling the Hadamard multiplication. We repeat the experiment for different transmission rates of the interferer.

Results: Fig. 11 reports the video PSNR at the receiver across all compared schemes as a function of the packet loss rate. The figure has a log scale. It shows that in both baselines the quality of video drops sharply even when the packet loss rate is less than 0.5%. This is because both the MPEG4 and SVC codecs introduce dependencies between packets due to Huffman encoding, differential encoding and motion compensation, as a result of which the loss of a single packet within a GoP can render the entire GoP undecodable. In contrast, SoftCast's performance degrades only gradually as packet loss increases, and is only mildly affected even at a loss rate as high as 10%. The figure also shows that Hadamard multiplication significantly improves SoftCast's resilience to packet loss. Interestingly, SoftCast is more resilient than MPEG4 even in the absence of Hadamard multiplication.

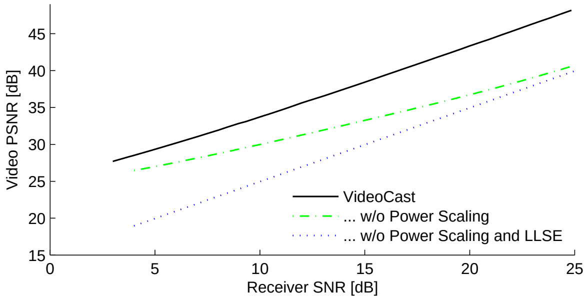


Fig. 12. **SoftCast Microbenchmark** The figure plots the contributions of SoftCast's components to its video quality. The figure shows that the use of LLSE is particularly important at low SNRs where as error protection via power scaling is important at high SNRs.

SoftCast's resilience to packet loss comes from:

- The use of a 3D DCT ensures that all SoftCast packets include information about all pixels in a GoP, hence the loss of a single packet does not create patches in a frame, but rather distributes errors smoothly across the entire GoP.
- SoftCast packets are not coded relative to each other as is the case for differential encoding or motion compensation. Hence the loss of one packet does not prevent the decoding of other received packets.
- All SoftCast packets have equal energy as a result of Hadamard multiplication, and hence the decoding quality degrades gracefully as packet losses increase. The LLSE decoder, in particular, leverages this property to decode the GoP even in the presence of packet loss.

E. *Microbenchmark*

We examine the contribution of SoftCast's components to its performance.

Method: We pick a sender receiver pair at random. We vary the SNR by varying the transmission power at the sender. For each SNR we make the sender transmit the video with SoftCast, SoftCast with linear scaling disabled, and SoftCast with both linear scaling and LLSE disabled. We repeat the experiments multiple times and report the average performance for each SNR value.

*Results:*The figure shows that SoftCast's approach to error protection based on linear scaling and LLSE decoding contributes significantly to its resilience. Specifically, linear scaling is important at high SNRs since it amplifies fine image details and protects them from being lost to noise. In contrast, the LLSE decoder is important at low SNRs when receiver measurements are noisy and cannot be trusted, because it allows the decoder to leverage its knowledge of the statistics of the DCT components.

X. Related Work

Recent years have witnessed much interest in making video quality scale with channel quality [3,24,27,44]. The general approach so far has been to divide the video stream into a base layer that is necessary for decoding the video, and an enhancement layer that improves its quality [6,12,16,36]. Proposals in this area differ mainly in how they generate the two layers and the code they use to protect them. For example, some proposals consider the I frames as the base layer and the P and B frames as the enhancement layer [49]. More recent approaches create a base layer by quantizing the video to a coarse representation, which is refined by the enhancement layers [12,36]. Given video layers of different importance, one has many choices for protecting them unequally. Some proposals put more FEC coding on the base layer than the enhancement layers [6,16]. Others employ embedded diversity coding, where a high-rate code allows the enhancement layer to harness good channel realizations, while the embedded high-diversity code provides guarantees that at least the base layer is received reliably [1,7,12]. Hierarchical modulation and super-position coding are examples of this approach [5,21]. Motivated by this prior work, SoftCast takes scalable video one step further; it disposes of the coarse granularity of layers in favor of a continuously scalable design.

Related work also includes analog and digital TV. Analog television also linearly transforms the luminance values for transmission. And, in fact, analog television also shares the property that the quality of the transmitted video degrades smoothly as the channel quality degrades. A key advantage of our approach is that our encoding scheme also leverages the powerful digital computation capabilities (which became available subsequent to the development of analog television) to encode the video both for compression and error protection. Hence, we can obtain transmission efficiency comparable to standard digital video coding schemes such as MPEG.

Digital TV also deals with video multicast [32]. The focus in digital TV however is on ensuring a minimum video quality to all receivers rather than on ensuring that each receiver obtains the best video quality supported by its channel. Further, the variability in channel quality is lower because there is neither mobility nor interference. In fact, proposals for extending Digital TV to mobile handheld devices argue for graceful degradation and propose to employs a 2-layer video with hierarchical modulation [21].

There is a large body of work that allows a source to adapt its transmission bitrate to a mobile receiver [4,20,41]. However, these schemes require fast feedback, and are limited to a single receiver. Further, they need to be augmented with additional mechanisms to adapt the video codec rate to fit within the available bitrate. In contrast, SoftCast provides a unified design that eliminates the need to adapt bitrate and video coding at the source, and instead allows the receiver to extract a video quality that matches its instantaneous channel.

Our work also builds on past work in information theory on rate distortion and joint source and channel coding (JSCC) [5]. This past work however mainly focuses on theoretical bounds [28,33]. Also the proposed codecs are typically non-linear [38] and significantly harder to implement than SoftCast.

Finally, SoftCast leverages a rich literature in signal and image processing, including decorrelation transforms such as 3D DCT [30], the least square estimator [22], the Hadamard transform [2], and optimal linear transforms [23]. SoftCast uses these tools in a novel PHY-video architecture to deliver a video quality that scales smoothly with channel quality.

XI. Concluding Remarks

This paper presents SoftCast, a clean-slate design for wireless video. SoftCast enables a video source to broadcast a single stream that each receiver decodes into a video quality commensurate with its instantaneous channel quality. Further, SoftCast requires no receiver feedback, bitrate adaptation, or video code rate adaptation.

SoftCast also has limitations. Specifically, SoftCast requires a compression scheme that is linear. Consider for example a white ball moving on a black background. MPEG can encode each frame by simply sending the shift of the ball's center, which is more efficient than using a linear transform like 3D DCT to compress the video. Such videos however are atypical and if they arise they can be transmitted using standard video coding. For videos of natural scenes, linear transforms like DCT and Wavelets are highly effective in compressing the video information [30,46]. Furthermore, the gains of SoftCast arise mainly from its robustness to channel errors and packet loss. In contrast, existing nonlinear video codecs are highly sensitive to errors. Hence, the existing design has to pay the cost of heavy PHY layer error protection codes (e.g., 802.11 typically uses 1/2 rate FEC codes, which halves the throughput available for data). We believe that a better tradeoff can be reached if the PHY is allowed to leverage the intrinsic resilience of video signals to deal with errors on the channel. In general, the tradeoffs between the gains from efficient but error-sensitive compression and the cost of error correction codes and packet retransmission at the lower layers are important research topics for the future of mobile and broadcast video.

References

[1] A. B. Abdurrhman, M. E. Woodward, and V. Theodorakopoulos. *Robust Transmission of H.264/AVC Video Using 64-QAM and Unequal Error Protection*, pages 117–125. 2009.

[2] K. Beauchamp. *Applications of Walsh and Related Functions*. Academic Press, 1984.

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

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

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

[6] D. Dardari, M. G. Martini, M. Mazzotti, and M. Chiani. Layered video transmission on adaptive ofdm wireless systems. *EURASIP J. Appl. Signal Process.*, 2004:1557–1567, 2004.

[7] S. Diggavi, A. Calderbank, S. Dusad, and N. Al-Dhahir. Diversity embedded spacetime codes. 54, Jan 2008.

[8] ETSI. *Digital Video Broadcasting; Framing structure, channel coding and modulation for digital terrestial television*, Jan 2009. EN 300 744.

[9] FFmpeg. http://www.ffmpeg.com.

[10] Fftw home page. `http://www.fftw.org/`.

[11] D. L. Gall. Mpeg: a video compression standard for multimedia applications. *Commun. ACM*, 34(4):46–58, 1991.

[12] M. M. Ghandi, B. Barmada, E. V. Jones, and M. Ghanbari. H.264 layered coded video over wireless networks: Channel coding and modulation constraints. *EURASIP J. Appl. Signal Process.*, 2006.

[13] The GNU software radio. `http://gnuradio.org/trac`.

[14] D. Gündüz, E. Erkip, A. J. Goldsmith, and H. V. Poor. Transmission of correlated sources over multi-user channels. *CoRR*, abs/0807.2666, 2008.

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

[16] C.-L. Huang and S. Liang. Unequal error protection for mpeg-2 video transmission over wireless channels. In *Signal Process. Image Commun.*, 2004.

[17] ITU-T. *Advanced video coding for generic audiovisual services*, May 2003. ITU-T Recommendation H.264.

[18] H. Jiang and P. Wilford. A hierarchical modulation for upgrading digital broadcast systems. 51, June 2005.

[19] SVC reference software. `http://ip.hhi.de/imagecom_G1/savce/downloads/SVC-Reference-Software.htm`.

[20] G. Judd, X. Wang, and P. Steenkiste. Efficient channel-aware rate adaptation in dynamic environments. In *ACM MobiSys*, 2008.

[21] T. Kratochvíl. *Hierarchical Modulation in DVB-T/H Mobile TV Transmission*, pages 333–341. 2009.

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

[23] K.-H. Lee and D. P. Petersen. Optimal linear coding for vector channels. *IEEE Trans. Communications*, Dec 1976.

[24] 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.

[25] Martin Schnell. Hadamard codewords as orthogonal spreading sequences in synchronous DS CDMA systems for mobile radio channels. In *IEEE Intl. Symposium on Spread Spectrum Techniques and Applications*, 1994.

[26] M. O. Martínez-Rach et al. Quality assessment metrics vs. PSNR under packet loss scenarios in MANET wireless networks. In *MV: Intl. workshop on mobile video*, 2007.

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

[28] 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.

[29] W. Perera. Architectures for multiplierless fast fourier transform hardware implementation in vlsi. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 35(12):1750–1760, Dec 1987.

[30] C. Podilchuk, N. Jayant, and N. Farvardin. Three-dimensional subband coding of video. 4(2), Feb 1995.

[31] H. Rahul, F. Edalat, D. Katabi, and C. Sodini. Frequency-Aware Rate Adaptation and MAC Protocols. In *ACM MOBICOM*, Sep 2009.

[32] U. Reimers. DVB-The family of international standards for digital video broadcasting. *Proc. of the IEEE*, 94(1):173–182, 2006.

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

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

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

[36] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the H.264/AVC standard. In *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2007.

[37] C. E. Shannon. Two-way communication channels. In *the 4th Berkeley Symp. Math. Satist. Probability*, 1961.

[38] 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.

[39] Universal software radio peripheral 2. `http://www.ettus.com/downloads/ettus_ds_usrp2_v2.pdf`.

[40] S. Vembu, S. Verdu, and Y. Steinberg. The source-channel separation theorem revisited. In *IEEE Trans. Inf. Theory*, 1995.

[41] M. Vutukuru, H. Balakrishnan, and K. Jamieson. Cross-Layer Wireless Bit Rate Adaptation. In *ACM SIGCOMM*, Aug 2009.

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

[43] M. Wien, H. Schwarz, and T. Oelbaum. Performance analysis of SVC. *IEEE Trans. Circuits and Systems for Video Technology*, 17(9), Sept. 2007.

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

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

[46] Z. Xiong, K. Ramchandran, M. Orchard, and Y.-Q. Zhang. A comparative study of dct- and wavelet-based image coding. *IEEE Trans. on Circuits and Systems for Video Technology*, Aug 1999.

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

[48] J. Xu, Z. Xiong, S. Li, and Y.-Q. Zhang. Memory-constrained 3-d wavelet transform for video coding without boundary effects. *IEEE Trans. on Circuits and Systems for Video Technology*, 12, 2002.

[49] X. Xu, M. van der Schaar, S. Krishnamachari, S. Choi, and Y. Wang. Fine-granular-scalability video streaming over wireless lans using cross layer error control. In *IEEE ICASSP*, volume 5, May 2004.

APPENDIX

PROOF OF LEMMA 4.1

We want to determine the set of optimal linear scaling factors for each chunk that minimizes the expected reconstruction error (computed as mean square error), while within the total power constraint, as inspired by [23]. Note that, since the DCT transform is orthogonal, the reconstruction error of chunk i ($x_i[1 \ldots N]$, where N is the number of DCT components in a chunk) is directly proportional to the reconstruction error in the video frame.

Let us model the channel as one with additive white noise. Thus, for each value in chunk i, $x_i[j]$, we transmit $y_i[j] = g_i x_i[j]$, and the receiver receives $\hat{y}_i[j] = y_i[j] + n$, where g_i is the linear scaling factor for this chunk, and n is a random variable with zero-mean and specific variance, σ (the same for all chunks). Subsequently, the receiver decodes

$$\hat{x}_i[j] = \frac{\hat{y}_i[j]}{g_i} = x_i[j] + \frac{n}{g_i}.$$

The expected mean square error is:

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

Clearly, the best scaling factor would be infinite, if not for the power constraint. Let $\lambda_i = E[x_i^2]$ be the power of chunk i ($x_i[1 \ldots N]$), $\mu_i = E[y_i^2]$ be its power after applying the scaling factor, and P the total power budget. We can drop N in the minimand since it is merely a constant factor, and formally rewrite the problem as follows.

$$\begin{aligned} \min err &= \sigma^2 \sum_i \frac{\lambda_i}{\mu_i} \qquad (8) \\ \text{subject to:} \quad & \sum_i \mu_i \leq P \text{ and } \mu_i \geq 0 \end{aligned}$$

We can solve this optimization using the technique of Lagrange multipliers. The Lagrangian is

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

Differentiating separately by μ_i and γ and setting to 0, yields:

$$\begin{aligned} \sqrt{\gamma} &= \sum_i \sqrt{\lambda_i \sigma^2} / P \\ \mu_i &= \sqrt{\frac{\lambda_i \sigma^2}{\gamma}} = P \frac{\sqrt{\lambda_i}}{\sum_i \sqrt{\lambda_i}} \\ g_i &= \sqrt{\frac{\mu_i}{\lambda_i}} = \sqrt{\frac{P}{\sqrt{\lambda_i} \sum_i \sqrt{\lambda_i}}} \end{aligned}$$

The optimal scaling factor for each chunk is therefore such that the resulting power of the row is proportional to the *square root* of its original power. Some readers might find it more intuitive that the optimal solution should completely equalize the resulting power, i.e. $\mu_i = P/k$; but substituting this in (8) shows otherwise.