A Cross-Layer Design for Scalable Mobile Video

Szymon Jakubczak
CSAIL MIT
32 Vassar St.
Cambridge, Mass. 02139
szym@alum.mit.edu

Dina Katabi
CSAIL MIT
32 Vassar St.
Cambridge, Mass. 02139
dk@mit.edu

ABSTRACT

Today's mobile video suffers from two limitations: 1) it cannot reduce bandwidth consumption by leveraging wireless broadcast to multicast popular content to interested receivers, and 2) it lacks robustness to wireless interference and errors. This paper presents SoftCast, a cross-layer design for mobile video that addresses both limitations. To do so, SoftCast changes the network stack to act like a linear transform. As a result, the transmitted video signal becomes *linearly* related to the pixels' luminance. Thus, when noise perturbs the transmitted signal samples, the perturbation naturally translates into approximation in the original video pixels. This enables a video source to multicast a single stream that each receiver decodes to a video quality commensurate with its channel quality. It also increases robustness to interference and errors which now reduce the sharpness of the received pixels but do not cause the video to glitch or stall. We have implemented SoftCast and evaluated it in a testbed of software radios. Our results show that it improves the average video quality for multicast users by 5.5 dB, eliminates video glitches caused by mobility, and increases robustness to packet loss by an order of magnitude.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Miscellaneous

General Terms

Algorithms, Design, Performance, Theory

Keywords

wireless networks, scalable video communications, joint source-channel coding

1. INTRODUCTION

Mobile video is predicted to be the next killer application for wireless networks [1]. In particular, according to the Cisco visual index, mobile video traffic will grow 66 fold over a period of five years [1]. Such predictions lead to a natural question: can existing wireless technologies, e.g., WiFi, WiMax, or LTE, support this impending demand and provide scalable and robust mobile video?

(a) Scalability. As demands for mobile video increase congestion will also increase. The problem becomes particularly severe when many users try to watch a popular realtime event, e.g., the Super Bowl game. In such case, one would like to save bandwidth by multicasting the event as a single video stream. Different receivers, however, have different channel qualities (i.e., SNRs). Multicasting a single video stream to multiple receivers requires the source to transmit at the lowest bit rate supported by their channels. This reduces all receivers to the video quality of the receiver with the worst channel. Since such a design is undesirable from a user perspective, the typical approach today transmits an individual video stream to each receiver, even when all of these streams share the same content, which is unscalable.

(b) Robustness. The wireless medium suffers high bit error and packet loss rates due to both interference and channel noise. Video codecs however are very sensitive to errors and losses [2,29]. Fig. 1 plots the impact of interference-caused packet loss on MPEG4 (i.e., H.264/AVC) and SVC layered-video.[1] The figure is generated using the reference implementations of the two codecs [13,34], and by having an interferer transmit at regular intervals. (Other details are in §8.4.) The figure confirms past results [29], showing that both MPEG4 video and SVC layered video are highly sensitive to interference and become unviewable (i.e., PSNR < 20 dB) when the packet loss rate is higher than 1%.

The lack of scalability and robustness in today's mobile video stems from the existing design of the network stack. Specifically, mobile video is impacted by two layers in the stack: the application video codec, which compresses the video, and the physical layer, which protects the video from channel errors and losses. Today, video codecs do an excellent job in compressing the video and removing redundancy. However, they also make the video highly vulnerable to bit errors and packet losses. In particular, all common video codecs use entropy coding (e.g., Huffman), in which a single bit flip can cause the receiver to confuse symbol boundaries, producing arbitrary errors in the video. This compressed video has to be transmitted over an erroneous wireless channel. Thus, the PHY layer has to add back redundancy in the

[1]SVC produces a base layer necessary for decoding and a refinement layer that adds details for receivers with better channels.

MobiCom'11, September 19–23, 2011, Las Vegas, Nevada, USA.

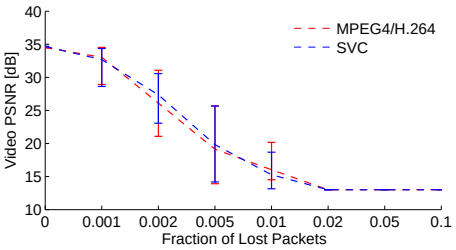


Figure 1: Impact of interference-related packet loss on video quality. PSNR below 20 dB corresponds to unacceptable video quality [23]. The figure shows that both H.264/MPEG4 and SVC (i.e., layered) videos suffer dramatically at a packet loss rate as low as 1%.

form of error protection codes. Since the compressed video is highly fragile, video streaming requires the PHY to add excessive redundancy to eliminate the possibility of bit flips or packet loss. This approach is particularly inefficient in mobile video because the PHY needs to add excessive coding to deal with channel variations across time due to mobility or interference, and channel variations across space due to receiver diversity.

Theoretical results show that the existing layer separation – i.e., separating source coding (i.e., video compression) from channel coding (i.e., error protection) – is acceptable *only* in the case of unicast channels and when the statistics of the channel are known *a priori* to the transmitter [27,30]. Such separation however becomes inefficient for multicast/broadcast channels, or when the channel's statistics are hard to predict due to mobility or interference [27,30].

This paper aims to improve the robustness and scalability of mobile video. The paper presents SoftCast, a cross-layer design of mobile video that both compresses the video and protect it from errors and losses. SoftCast starts with a video that is represented as a sequence of numbers, with each number representing a pixel luminance. 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 ensures that the signal samples transmitted on the channel are linearly related to the original pixel values. Therefore, increasing channel noise progressively perturbs the transmitted bits in proportion to their significance to the video application; high-quality channels perturb only the least significant bits while low-quality channels still preserve the most significant bits. Thus, each receiver decodes the received signal into a video whose quality is proportional to the quality of its specific instantaneous channel. Furthermore, this occurs with no receiver feedback, bitrate adaptation, or video code rate adaptation.

SoftCast realizes the above design using the following components:

(a) Error-Resilient Compression: SoftCast compresses the video using a weighted 3-dimensional DCT transform [32], where the weights are optimized to minimize the reconstruction errors in the received video. Using 3D DCT allows SoftCast to remove redundant information within a frame as well as across frames while maintaining its linear behavior. While DCT use is widespread in video compression, past work applies entropy coding (e.g., Huffman) after DCT thereby destroying linearity and making the video fragile to bit errors and packet losses [19]. This forces the PHY to compensate for the lack of robustness by adding back the redundancy in the form of error protection codes. In contrast, SoftCast does not use traditional entropy coding; instead, it weighs the DCT components according to their entropy, i.e., the amount of information they contribute to the decoded video. This allows SoftCast to leverage the basic idea underlying entropy coding but without destroying the linearity of its design. As a result, the physical layer does not need to add excessive redundancy to protect the video, which produces an efficient end-to-end design.

(b) 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 employs a linear Hadamard transform [3] to distribute the video information across packets such that each packet has approximately the same amount of information. As a result, all packets contribute equally to the decoded video, and the loss of a few packets does not cause sharp degradation in the video quality.

We note that despite its cross-layer design, SoftCast is relatively easy to incorporate within the existing network stack. Specifically, SoftCast is built atop an OFDM physical layer similar to that used in today's WiFi, WiMax and LTE, and hence can be realized in such systems by having the OFDM PHY layer send the values at SoftCast's output as the I and Q components of the transmitted digital signal.

We have implemented SoftCast and evaluated it in a testbed of 20 GNURadio USRP2 nodes. We compare it with two baselines: 1) MPEG4 (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 [15]. We evaluate these schemes using the Peak Signal-to-Noise Ratio (PSNR), a standard metric of video quality [23].[2] We have the following findings:

- SoftCast can multicast a single video stream that delivers to each receiver a video quality that matches – within 1 dB – the video quality the receiver would obtain if it were the only receiver in the multicast group and the source tailored its transmission to the receiver's channel quality.
- For multicast receivers of SNRs in the range $[5, 25]$ dB, SoftCast improves the average PSNR by 5.5 dB (a significant improvement to video quality [20,23]) over the best performer of the two baselines.
- SoftCast tolerates an order of magnitude higher packet loss rates than both 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.
- Our evaluation also explores the limitations of SoftCast. Our results show that SoftCast is suitable for scenarios in which the wireless bandwidth is the bottleneck. However, its performance becomes suboptimal when bandwidth is not the bottleneck, e.g.,in a wideband low SNR channel. We believe that many typical environments are bottle-

[2]In general, improvements in PSNR of magnitude larger than 1 dB are visually noticeable and a PSNR below 20 dB is not acceptable [20,23].

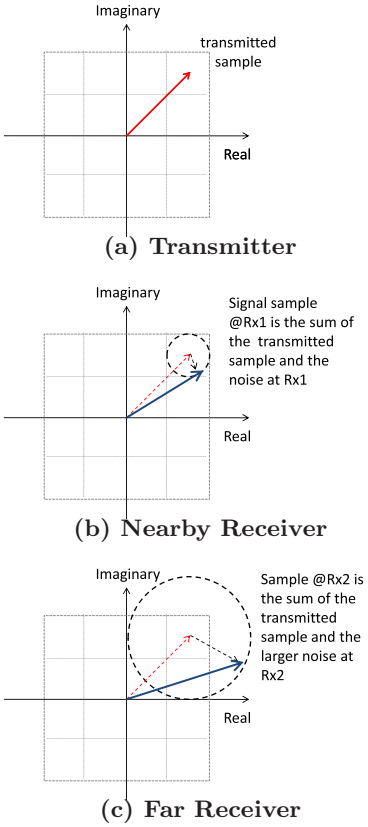


Figure 2: Wireless broadcast delivers more signal bits to low noise receivers. The figure shows the transmitted sample in red, the received samples in blue, and noise in black. The source transmits the signal sample in (a). A nearby receiver experiences less noise and can estimate the transmitted sample 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.

necked at the wireless bandwidth and hence can benefit from SoftCast.

Contributions. The paper presents a novel cross-layer design for mobile video, where the whole network stack, from the PHY layer to the application, acts as a linear transform that both compresses the video and protects it from channel errors and packet loss. The paper also shows that such a linear stack can run on top of an OFDM physical layer, making it applicable to modern wireless technologies, e.g., WiFi and WiMax. Finally, the paper implements and empirically evaluates its design demonstrating its benefits in practice.

2. SoftCast OVERVIEW

SoftCast's integrated design harnesses the intrinsic characteristics of both wireless broadcast and video to increase robustness and scalability. 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 [9], 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. This principle naturally enables a transmitter to satisfy multiple receivers with diverse channel qualities, as well as a single receiver where different packets experience different channel qualities due to mobility or interference.

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 cross-layer integrated video-PHY design. It 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 real 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.

3. SoftCast'S ENCODER

SoftCast has a cross-layer encoder that both compresses the video and encodes it for error and loss protection.

3.1 Video Compression

Both MPEG and SoftCast exploit spatial and temporal correlation in a GoP[3] 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, transforming 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. The grey level after 3D DCT reflects the magnitude of the DCT component in that location. The figure shows two important properties of 3D DCT:

[3]GoP is Group of Pictures, a sequence of successive frames. The video stream is composed of successive GoPs.

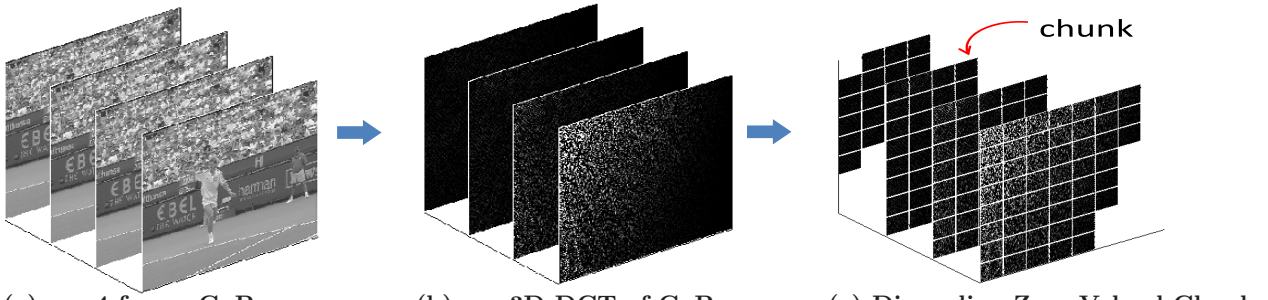


Figure 3: 3D DCT of a 4-frame GoP. The figure shows (a) a 4-frame GoP, (b) its 3D DCT, where each plane has a constant temporal frequency, and the values in the upper-left corner correspond to low spatial frequencies, (c) the non-zero DCT components in each plane grouped into chunks. Most DCT components are zero (black dots) and hence can be discarded. Further, the non-zero DCT components are clustered together.

(1) In natural images, most DCT components have a zero value, i.e., have no information because frames tend to be smooth [32], and hence the high spatial frequencies tend to be zero. Further, most of the higher temporal frequencies tend to be zero since most of the structure in a video stays constant across multiple frames [9]. We can discard all such zero-valued DCT components without affecting the quality of the video.
(2) Non-zero DCT components are spatially clustered. This means that we 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 can be compressed using run-length encoding.

The previous discussion assumed that the sender has enough bandwidth to transmit all the non-zero chunks over the wireless medium. What if the sender 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:

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

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 independent of the receiver (e.g., an 802.11 channel has a bandwidth of 20 MHz), whereas SNR is a property of the receiver's channel. Thus discarding non-zero chunks to fit the bandwidth does not prevent each receiver from getting a video quality commensurate with its SNR.

Two points are worth noting:

- 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 Fig. 3b, the planes in the back are mostly black).
- While some past work has tried 3D DCT compression it followed it with standard entropy coding [19]. Such an approach is inefficient because: 1) 3D DCT followed by standard entropy coding is a slightly less efficient compression scheme than H.264 [19]; 2) Once followed by entropy coding, 3D DCT loses linearity and becomes equally vulnerable to bit errors as H.264, requiring the PHY to add the redundancy back in the form or error protection codes. Instead, SoftCast preserves the linearity of 3D DCT, and replaces traditional entropy coding with an error protection code that weighs the DCT components according to their entropy. This allows SoftCast to leverage the basic idea underlying entropy coding but without making the video fragile to bit errors, as shown below.

3.2 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 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 sets a fixed power budget for the transmitted signal, scaling up and hence expending more power on some signal samples translates to expending less power on other samples. SoftCast finds the optimal scaling factors that balance this tension in a manner that reflects the amount of information in the DCT components, i.e., their entropy or variance.

Specifically, 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 entropy, i.e., 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. We can show that:[4]

LEMMA 3.1. *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:*

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

Note that there is only one scaling factor g_i for every distribution $\mathcal{D}_i$, i.e., one scaling factor per chunk. The encoder outputs 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.

3.3 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 [3,24]. 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.

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 as the digital signal samples to be transmitted, as described in §5.

3.4 Metadata

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 mean and the variance of each chunk, and a bitmap that indicates the discarded chunks. The decoder can compute the scaling factors (g_i) from this information. As for the Hadamard and DCT matrices, they are well known and need not be sent. The bitmap of chunks is compressed using run length encoding as described in §3.1, 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, i.e, the modulation and FEC code of the lowest 802.11 bit rate. To ensure 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. 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. This is a high redundancy code but since the metadata is very small, we can afford a code

[4]Proof available in technical report, omitted for anonymity.

that doubles its size.

3.5 The Encoder: A Matrix View

We can compactly represent the encoding 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

$$Y = HGX = CX \tag{2}$$

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.

4. SoftCast'S VIDEO DECODER

At the receiver, and as will be described in §5, 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 transmitted value $y_i[j]$, we receive a value $\hat{y}_i[j] = y_i[j] + n_i[j]$, where $n_i[j]$ is random channel noise. It is common to assume the noise is additive, white and Gaussian, which though not exact, works well in practice.

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

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

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 noise.

Without loss of generality, we can assume the slice size is small enough that it fits within a packet, and hence each row in $\hat{Y}$ is sent in a single packet. If the slice 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.[5] Thus, for the rest of our exposition, we assume that each packet contains a full slice.

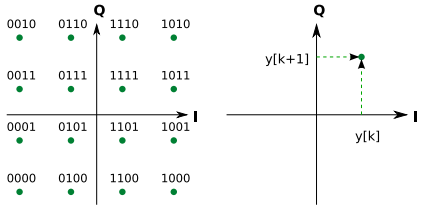
The receiver knows the received values, $\hat{Y}$, and can construct the encoding matrix C from the metadata. It then needs to compute its best estimate of the original DCT components, X. The linear solution to this problem is widely known as the Linear Least Square Estimator (LLSE) [16]. The LLSE provides 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:

$$X_{LLSE} = \Lambda_x C^T (C\Lambda_x C^T + \Sigma)^{-1} \hat{Y}, \tag{3}$$

where: 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 $\hat{Y}$ [6] and 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.

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

[6]The PHY has an estimate of the noise power in each packet, and can expose it to the higher layer.



(a) 16-QAM (b) SoftCast

Figure 4: Mapping coded video to I/Q components of transmitted signal. The traditional PHY maps a bit sequence to the complex number corresponding to the point labeled with that sequence. In contrast, SoftCast's PHY treats pairs of coded values as the real and imaginary parts of a complex number.

Consider how the LLSE estimator changes with SNR. At high SNR (i.e., small noise, the entries in Σ approach 0), Eq. 3 becomes:

$$X_{LLSE} \quad \approx \quad C^{-1} Y \tag{4}$$

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 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.

4.1 Decoding in the Presence of Packet Loss

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

$$\hat{Y}_{*i} = C_{*i} X + N_{*i}. \tag{5}$$

The LLSE decoder becomes:

$$X_{LLSE} = \Lambda_x C_{*i}^T (C_{*i} \Lambda_x C_{*i}^T + \Sigma_{(*i,*i)})^{-1} \hat{Y}_{*i}. \tag{6}$$

Note that we remove a row and a column from Σ. Eq. 6 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.

5. 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 se-

quences of 4 bits and maps each sequence to a complex I/Q number as shown in Fig. 4a.[7]

In contrast to the existing 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 of the digital signal samples, as in Fig. 4b.[8]

To integrate this design into the existing 802.11 PHY layer, we leverage that OFDM separates channel estimation and tracking from data transmission [11]. 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, CFO estimation, channel estimation, and phase tracking. SoftCast simply transmits in each of the OFDM data subcarrier, 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.

6. IMPLEMENTATION

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

- MPEG4 part 10 (i.e., H.264/AVC) over 802.11 PHY.
- Layered video where the video is coded using the scalable video extension (SVC) of H.264 [13] and transmitted over hierarchical modulation [15].

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 GNURadio, which we augmented to incorporate pilot subcarriers and phase tracking, two standard components in OFDM receivers [11]. We also developed software modules that perform 802.11 interleaving, convolutional coding, and Viterbi decoding.

The transmitter's PHY passes SoftCast's packets directly to OFDM, whereas MPEG4 and SVC-encoded packets are subjected to convolutional coding and interleaving, where the code rate depends on the chosen bit rate. MPEG4 packets are 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 applies CFO correction, channel estimation and correction, and phase tracking. The receiver then inverts the execution branches at the transmitter.

Video Coding. We implemented SoftCast in Python (with SciPy). For the baselines, we used reference implementation

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

[8]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.

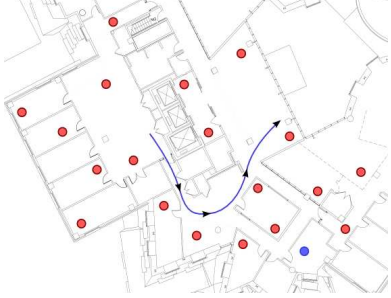


Figure 5: Testbed. Dots refer to nodes; the line shows the path of the receiver in the mobility experiment when the blue dot was the transmitter.

available online. Specifically, we generate MPEG4 streams using the H.264/AVC [12,22] codec provided by the FFmpeg software and the x264 codec library [34]. We configured x264 to use *high* profile and tuned it for very high quality as recommended in [35]. We generate the SVC streams using the JSVM implementation [13], which allows us to control the number of layers. We configured JSVM to use Coarse-Grain SNR Scalability (CGS). Also for MPEG4 and SVC-HM we add an outer Reed-Solomon code for error protection with the same parameters (188/204) used for digital TV [8]. Packets of each layer of MPEG4 and SVC-HM are individually interleaved between the outer Reed-Solomon code and the inner FEC in accordance with the same recommendation. All schemes: MPEG4, SVC-HM, and SoftCast use a GoP of 16 frames and are required to obey a fixed data rate over a buffer of 1 second.

7. EVALUATION ENVIRONMENT

Testbed: We run our experiments in the 20-node GNURadio testbed shown in Fig. 5. Each node is a laptop connected to a USRP2 radio board. We use the RFX2400 daughterboards which operate in the 2.4 GHz range.

Modulation and Coding: 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 [15]. It is allowed to control how to divide transmission power between the layers to achieve the best performance [15]. 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. After preambles, pilots and cyclic prefix the remaining data bandwidth equals 1.03 MHz. Since the USRP radios operate in the same frequency band as 802.11 WLANs but use a much narrower channel, 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 [23] and is defined as a function of the mean squared error (MSE) between all pixels of the decoded video and the

original as $PSNR = 20\ \log_{10} \frac{2^L - 1}{\sqrt{MSE}}$ $[dB]$, 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, and differences of 1 dB or higher are visible [23].

Test Videos: We use standard reference videos in the SIF format (352 × 240 pixels, 30 fps) from the Xiph [36] collection. Since codec performance varies from one video to another, we create one monochrome[9] 480-frame test video by splicing 32 frames (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*. Observe that 32 frames make two complete GoPs and hence such splicing does not affect compression potential of any of the compared schemes, since none of them is allowed to code across GoPs. For the mobility experiment we used the 512-frame video *football* on which the compared schemes performed similarly in the static scenario.

8. RESULTS

We empirically evaluate SoftCast and compare it against: 1) the conventional design, which uses H.264 (i.e., MPEG4 Part 10) 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 [15].

8.1 Benchmark Results

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. 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. 6. This procedure is used for all lines in the figure. We repeat the experiment by randomly picking the sender from the nodes in the testbed.

Results: Fig. 6a shows that for any selection of transmission bit rate 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. 6b 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 single layer MPEG4. Layering incurs overhead both at the PHY and the video codec. At any fixed PSNR in Fig. 6b, layered video needs a higher SNR than the single layer approach to achieve the same PSNR. This is because in hierarchical modulation, each higher layer is noise for the lower layers. Also, 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 [33].

[9]We omit the treatment of chroma (color) information as the coding of both SoftCast and MPEG can be extended to multiple video channels.

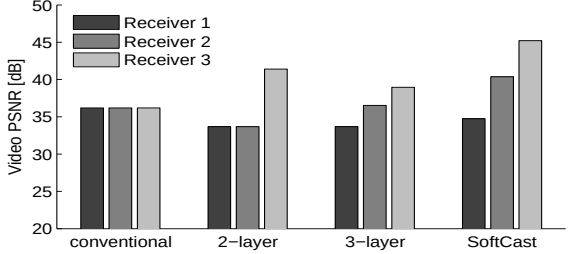


Figure 7: Multicast to three receivers. The figure shows that layering provides service differentiation between receivers as opposed to single layer MPEG4. But layering incurs overhead at the PHY and the codec, and hence extra layers reduce the maximum achievable video quality. In contrast, SoftCast provides service differentiation while achieving a higher overall video quality.

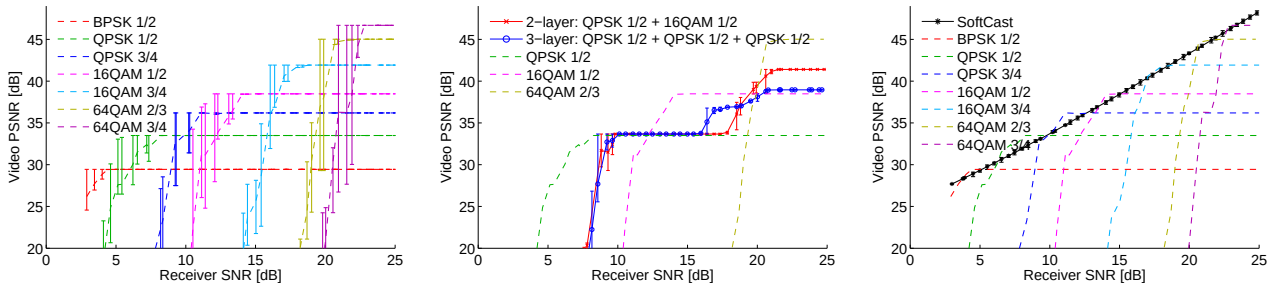
In contrast, SoftCast's performance shown in Fig. 6c scales smoothly with the channel SNR. Further, SoftCast's PSNR matches the envelope of the conventional design curves at each SNR. The combination of these two observations means that SoftCast can significantly improve video performance for mobile and multicast receivers while maintaining the efficiency of the existing design for the case of a single static receiver.

It is worth noting that these results do not mean that SoftCast outperforms MPEG4's compression. 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. 6a stems from 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.

8.2 Multicast

Method: We pick a single sender and three multicast receivers from 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 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. 7 shows the PSNR of the three multicast receivers. It shows that, in the conventional design, the 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



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

Figure 6: Approaches to Wireless Video: (a) 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). (b) 2-layer video in red and 3-layer video in blue. For reference, the dashed lines are the three equivalent single-layer MPEG4 videos. (c) Performance of SoftCast (in black) vs. single-layer MPEG4.

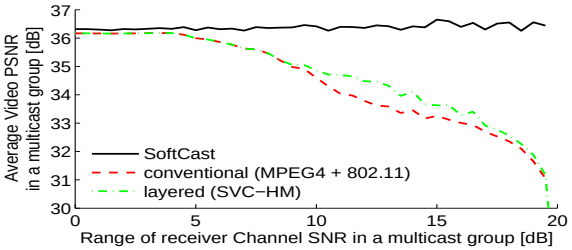


Figure 8: Serving a multicast group with diverse receivers. The figure plots the average PSNR across receivers in a multicast group as a function of the SNR range in the group. The conventional design and SVC-HM provide a significantly lower average video quality than SoftCast for multicast group with a large SNR span.

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 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 ($\pm$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. 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 1 layer, 2 layers, or 3 layers.

Results: Fig. 8 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. 7b 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.

8.3 Mobility of a Single Receiver

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 [31]. Specifically, we perform the mobility experiment with non-video packets from which we can extract the errors in the I/Q values to create a noise pattern. 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. 5 shows the path followed during the mobility experiments.

We allow the conventional design to adapt its bit rate and video code rate. To adapt the bit rate we use SoftRate [31], 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 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. 9a shows the SNR in the individual packets in the mobility trace. Fig. 9b shows the transmission bit rates picked by SoftRate and used in the conventional design. Fig. 9c 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 that in hierarchical modulation, the enhancement layers are effectively noise during the decoding of the base layer, mak-

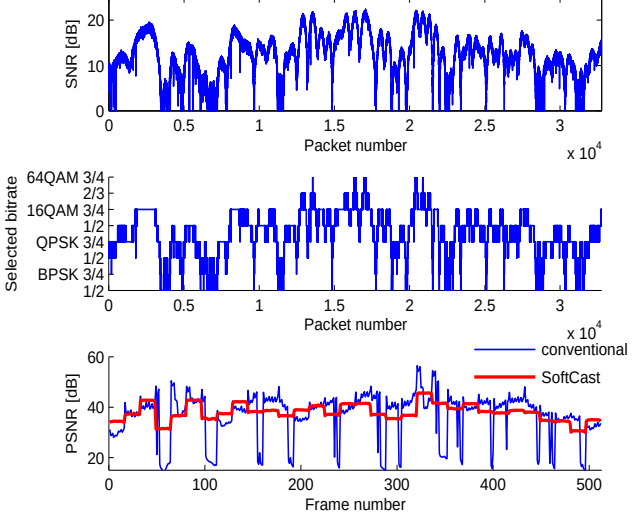


Figure 9: Mobility. The figure compares the video quality of the conventional design and SoftCast under mobility. The conventional design is allowed to adapt its bitrate and video code rate. The top graph shows the SNR of the received packets, the middle graph shows the transmission bit rate chosen by 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.

ing 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.

We identify glitches as frames whose PSNR is below 20 dB [20]. Fig 9c shows that, with mobility, the conventional wireless design based on MPEG4 experiences significant glitches in video quality. These glitches happen when a drop in the transmission bit rate causes significant packet loss so that even with retransmissions, it might still prevent timely decoding of the video frames. In comparison, SoftCast's performance is stable even in the presence of mobility. SoftCast achieves high robustness to packet loss because it avoids Huffman and differential encoding and it spreads the video information across all packets. In this mobile experiment, 14% of the frames transmitted using the conventional design suffer from glitches. SoftCast however has eliminated all such glitches.

8.4 Robustness 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. 10 reports the video PSNR at the receiver

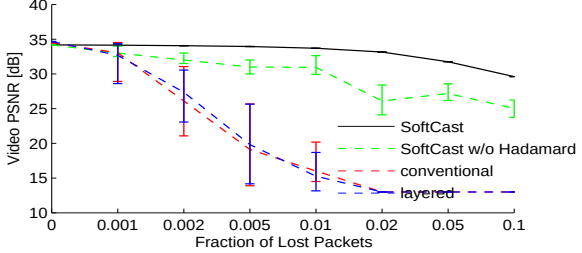


Figure 10: Robustness to packet loss. The figure shows that both SVC-HM and the conventional MPEG-based design suffer dramatically at a packet loss rate as low as 1%. In contrast, SoftCast's is only mildly affected even when the loss rate is as high as 10%. For reference, the figure shows the performance of SoftCast if it did not use the Hadamard matrix to ensure that all packets are equally important.

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 1%. 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 robustness to packet loss. Interestingly, SoftCast is more robust than MPEG4 even in the absence of Hadamard multiplication.

8.5 Impact of Available Wireless Bandwidth

Next, we are interested in exploring SoftCast's limitations. SoftCast is designed for environments where the wireless bandwidth is the bottleneck, i.e., the video source is too big to fit within the available channel bandwidth. (Note, if a 20MHz channel is shared by 10 users, then the available bandwidth per user is 2MHz.) The source bandwidth is typically defined as the number of dimensions/sec, which in the case of a video source refers to the number of pixel values per second [5]. If the available wireless bandwidth is less than the video source bandwidth, SoftCast compresses the video by dropping low energy 3D DCT frequencies. However, SoftCast's existing design has no particular approach to deal with environments where the source's bandwidth may be higher than the wireless bandwidth. The conventional design can leverage such scenarios to make a wideband low SNR channel perform as if it were a high SNR narrow bandwidth channel, using an approach called bandwidth expansion [5,28]. However we are unaware of good linear codes for bandwidth expansion. A straight-forward linear code would simply repeat the same signal; however repetition is not efficient. Below we show empirical results from scenarios with bandwidth expansion.

Method: We pick a single sender-receiver pair with 10 dB SNR. We vary the available wireless bandwidth by changing the packet rate on the USRP2, and transmit the same video with both with SoftCast and MPEG4. For scenarios that require bandwidth expansion we make SoftCast simply repeat the signal. As for MPEG4, the 802.11-like PHY naturally

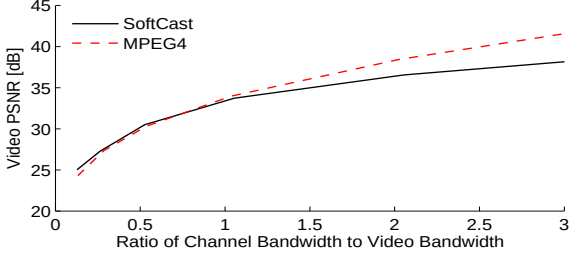


Figure 11: Impact of Available Wireless Bandwidth. The figure plots the performance of SoftCast and MPEG4 for a single receiver with 10 dB SNR as a function of the ratio of wireless bandwidth to video bandwidth (i.e., pixels/s). SoftCast is suitable for environments where it is desirable to send a large video over a relatively low bandwidth channel.

performs bandwidth expansion.

Results: Fig. 11 shows that SoftCast remains competitive in a wide range of scenarios where the wireless bandwidth is smaller than the source bandwidth. In scenarios where wireless bandwidth is significantly larger, SoftCast is unable to efficiently utilize the bandwidth. This is a limitation of SoftCast's linear design which given surplus bandwidth can only apply repetition coding. However, the wireless bandwidth is a shared scarce resource. Hence, we believe, most practical scenarios are limited by the wireless bandwidth, and can benefit from SoftCast's design.

9. RELATED WORK

The general approach to video multicast divides the video stream into a base layer necessary for all receivers to decode the video, and an enhancement layer that improves the video quality for receivers with better channels [6,10,25,26]. 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 (reference) frames as the base layer and the P and B (differential) frames as the enhancement layer [26]. Others create the base layer by quantizing the video to a coarse representation, which is refined by the enhancement layers [10,25]. With 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]. Others employ embedded diversity coding [7,10], 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. Hierarchical modulation and super-position coding are examples of this approach [5,15,26]. In contrast to these designs, SoftCast adopts a cross-layer approach that disposes of the coarse granularity of layers in favor of a continuously scalable design. Also, SoftCast's approach is more resilient to interference and channel errors than layered video schemes since these schemes rely on entropy coding (e.g., Huffman), which is highly sensitive to errors. This is particularly the case in interference settings where all layers can be badly corrupted.

Related work also includes analog and digital TV. SoftCast shares with analog TV the property that the transmitted signal is linearly related to the pixel values. Analog TV however transmits uncompressed video, whereas SoftCast leverages the computation capabilities of digital hardware to compress the video, obtaining an efficiency comparable to digital video. Digital TV deals with wireless video multicast [21]. However, it focuses on delivering a minimum video quality to all receivers rather than providing each receiver the best video quality supported by its channel. Further, the variability in channel quality in digital TV is lower than mobile TV because the network is static and can be carefully designed by the provider. In fact, past proposals for extending Digital TV to mobile handheld devices argue the need for graceful degradation and propose to employ a 2-layer video with hierarchical modulation [15].

There is a large body of work that allows a source to adapt its transmission bitrate to a mobile receiver [4,14,31]. However, such schemes require fast feedback and are limited to a single receiver. They must also be augmented with additional mechanisms to adapt the video codec rate to fit 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 provides the receiver with a video quality that matches its instantaneous channel.

Our work builds on past work on rate distortion and joint source-channel coding (JSCC) [5]. This past work mainly focuses on theoretical bounds [18], and the proposed codecs are often non-linear [28] and hard to implement.

Finally, SoftCast leverages a rich literature in signal and image processing, including decorrelation transforms, e.g., DCT [19], the least square estimator [16], the Hadamard [3], and optimal linear transforms [17]. SoftCast uses these tools in a novel PHY-video architecture to deliver a video quality that scales smoothly with channel quality.

10. CONCLUSION

This paper presents SoftCast, a clean-slate design for mobile video. SoftCast adopts an integrated design for video and PHY layer coding, making the whole network stack act as a linear transform. We show that such a design improves the video quality for multicast users, eliminates video glitches caused by mobility, and increases robustness to interference and channel errors.

11. REFERENCES

[1] Cisco visual networking index: Forecast and methodology 2008-2013. `http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html`, June 2009.

[2] G. Bai and C. Williamson. The Effects of Mobility on Wireless Media Streaming Performance. In *In Proc. Wireless Networks and Emerging Technologies*, 2004.

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

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

[10] 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.

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

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

[13] SVC reference software. `http://ip.hhi.de/imagecom_G1/savce/downloads/`.

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

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

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

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

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

[19] C. Podilchuk, N. Jayant, and N. Farvardin. Three-dimensional subband coding of video. Feb 1995.

[20] P. W. Rabbani, M.; Jones. *Digital Image Compression Techniques*. Bellingham (WA):SPIE Optical Engineering Press, 1991.

[21] U. Reimers. DVB-The family of international standards for digital video broadcasting. *Proc. of the IEEE*, 2006.

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

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

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

[25] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the H.264/AVC standard. In *IEEE ISCAS*, 2007.

[26] S. Sen, S. Gilani, S. Srinath, S. Schmitt, and S. Banerjee. Design and implementation of an approximate communication system for wireless media applications. In *SIGCOMM*, 2010.

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

[28] M. Skoglund, N. Phamdo, and F. Alajaji. Hybrid digital-analog source-channel coding for bandwidth compression/expansion. *IEEE Trans. Information Theory*, 2006.

[29] M. Slanina, T. Kratochvíl, and V. Říčny. Robustness of compressed high definition video to transmission packet loss. In *ELMAR, 2010 PROCEEDINGS*, Sep 2010.

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

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

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

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

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

[35] Encoding with the x264 codec. `http://www.mplayerhq.hu/DOCS/HTML/en/menc-feat-x264.html`.

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