ParCast: Soft Video Delivery in MIMO-OFDM WLANs

Xiao Lin Liu*, Wenjun Hu†, Qifan Pu†*, Feng Wu†, and Yongguang Zhang†

Microsoft Research Asia† University of Science and Technology of China*

{wenjun, fengwu, ygz}@microsoft.com {lin717, puqf}@mail.ustc.edu.cn

ABSTRACT

We observe two trends, growing wireless capability at the physical layer powered by MIMO-OFDM, and growing video traffic as the dominant application traffic. Both the source and MIMO-OFDM channel components exhibit non-uniform energy distribution. This motivates us to leverage the source data redundancy at the channel to achieve high video recovery performance. We propose ParCast that first separates the source and channel into independent components, matches the more important source components with higher-gain channel components, allocates power weights with joint consideration to the source and the channel, and uses analog modulation for transmission. Such a scheme achieves fine-grained unequal error protection across source components. We implemented ParCast in Matlab and on Sora. Extensive evaluation shows that our scheme outperforms competitive schemes by notable margins, sometimes up to 5 dB in PSNR for challenging scenarios.

Categories and Subject Descriptors

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

General Terms

Algorithms, Design, Performance

Keywords

MIMO-OFDM, Video, Joint source-channel design

1. INTRODUCTION

MIMO-OFDM technologies have become the default building blocks for next generation wireless networks. As wireless capability continues to grow, so does the application demand. According to the Cisco Visual Networking Index [7], traffic from wireless devices will exceed traffic from wired devices by 2015. Internet video is a major source of such traffic growth, which accounted for 40% of consumer Internet traffic in 2010, and will reach 62% by the end of 2015. Video-on-demand traffic is expected to triple by 2015, and video unicast constitutes the bulk of the volume. As a result, considerable efforts have been devoted to improving video delivery quality over wireless links, especially unicast. Supporting in-home high-definition video streaming is precisely one of the motivations for 802.11ac.

*The work was done while Xiao Lin Liu and Qifan Pu were research interns with Microsoft Research Asia.

Traditionally, video sources are first compressed, represented digitally as a bit stream, and then transmitted in the same way as other binary data, as shown by MPEG [19]. Conventional video coding schemes normally include a Discrete Cosine Transform (DCT) to compact energy across all pixels and remove correlation between them. Non-uniformly distributed DCT coefficient energy levels imply statistical features, or *redundancy*, in the source data, and suitable entropy coding could remove such redundancy. Less redundancy means more power per unit compressed data, and hence better error resilience and overall decoding performance. Given the expected channel condition, we can then separately determine the compression rate at the source and the transmission rate at the channel. However, if the actual channel condition turns out worse than expected, the predetermined source and channel rates would have been too aggressive and cause bit errors in decoding. Since digital rates do not have a graceful fallback behavior in the face of bit errors, glitches will occur. This can happen frequently for wireless links, due to unpredictable channel conditions.

While wireless links are loss prone, the original video need not be received in its entirety to ensure a good visual quality. A synergy between the two sides could therefore be more effective than separately optimizing the source and the channel coding. This has motivated an industry standard effort, Wireless High Definition Interface (WHDI) [4], which sends uncompressed high definition videos over wireless links. Nevertheless, this is still inefficient due to the sheer volume of uncompressed video data for transmission, since the redundancy within the data is not utilized at either the source encoder or the decoder.

A few recent cross-layer approaches aim to better exploit the redundancy in the source to improve video delivery quality over wireless. They are motivated by the observation that such source redundancy could be used at the channel to protect against errors. *SoftCast* [13] starts with a similar DCT transform to that in MPEG, but then allocates power weights based on the redundancy level of the source components before transmission. The entire codec uses a series of linear transforms to ensure the mean squared error of the reconstructed pixel luminance is proportional to the channel noise throughout. While conventional systems perform *lossy compression over lossless digital communication*, SoftCast performs *lossless compression over lossy analog communication. FlexCast* [8] retains much of the MPEG encoding process, but replaces the entropy coding stage with a rateless code. The source blocks that contribute more to the overall distortion, i.e., the high-entropy components, are represented with more bits. Note that allocating different amounts of power or bits to different source components is a form of unequal error protection (UEP). Both SoftCast and FlexCast use a single code to simultaneously compress and protect the source. Their performance shows that it is unnecessary to optimally compress the source, provided the amount of residual source redundancy matches the required error protection over the channel.

However, these approaches were designed with single-antenna links in mind or channel oblivious for broadcast. OFDM (Orthogonal Frequency Division Multiplexing) decomposes a wideband channel into a set of mutually orthogonal *subcarriers*, and the channel gains across these subcarriers are usually different [11], sometimes by as much as 20 dB. With MIMO (Multiple-Input Multiple-Output), each subcarrier is further divided into a set of *spatial subchannels*, again with different channel gains. Furthermore, a channel dependent *precoding* operation is often necessary to make the spatial subchannels on the subcarrier mutually orthogonal, or *independent*, so that signals do not interfere with one another along different subchannels. As a result, even for unicast, there are several issues with running SoftCast or FlexCast directly over a MIMO-OFDM link. In particular, error behavior differs across subchannels. If a one-size-fits-all code rate is used for a few or all subchannels, it generally needs to be conservative, and hence suboptimal, to accommodate the worst subchannels. In this sense, unicast over a MIMO-OFDM link resembles a broadcast channel. The need for precoding makes it difficult to ensure that a single channel oblivious error protection scheme can perform well across MIMO-OFDM links.

Given that both the source DCT component energy and subchannel gains are non-uniformly distributed, if we encode the video source such that the more important, high-energy DCT components will be transmitted in high-gain subchannels, and less important parts in lower-gain subchannels, we may better utilize the overall channel. Note, however, that the DCT and the precoding steps are essential at the source and channel respectively to avoid interference between source components and between subchannels. This suggests that we should match DCT components to subchannels based on the respective sorted order of energy level, and then perform joint source-channel power allocation to optimize per DCT component and subchannel error performance. This is unequal error protection operating at a finer granularity than conventional layered video coding.

With these in mind, we present ParCast (**Par**allel video uni**Cast**), which tailors the unicast video delivery quality to the MIMO-OFDM channel. The key features are: (1) obtaining independent source components and subchannels, (2) matching important source components to high-gain subchannels, (3) scaling the source data with power weights computed with joint source-channel considerations, and (4) transmission using analog modulation.

We implement the video codec in Matlab, and the channel dependent modules of ParCast both as a Matlab simulation and on Sora [28]. We run experiments on Sora to validate the Matlab simulation, and then run channel trace driven simulations to compare ParCast against alternatives. Results show that ParCast outperforms the best conventional digital scheme, sometimes by 5 dB, for videos with fast motion and a large energy spread over links with a large subchannel gain spread. This is a challenging case for conventional schemes, but favors ParCast.

To summarize, our main contributions are: First, we highlight the analogy between the energy distributions for video sources and MIMO-OFDM channels, and show how imperfect distributions for both can work in synergy. Second, we show the importance of decorrelating both the source and the channel and matching them accordingly at a fine granularity. Third, we show that a simple scheme could achieve significant benefit by implementing the above steps. Extensive evaluation shows that this is indeed an effective video delivery mechanism over MIMO-OFDM links.

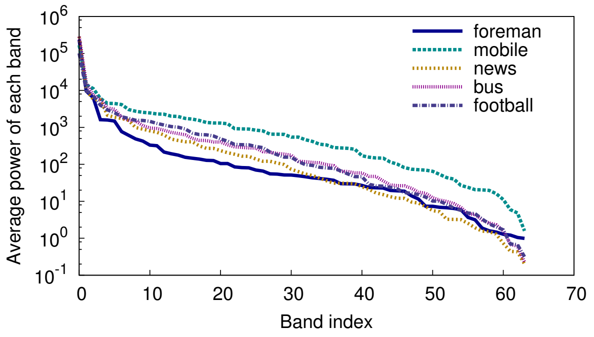

Figure 1: 2D 8×8 block DCT coefficient energy.

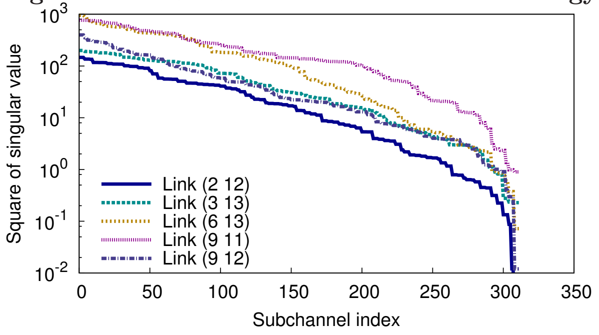

Figure 2: 3×3 MIMO-OFDM subchannel gains.

2. BACKGROUND AND MOTIVATION

2.1 Source and channel characteristics

Source. Video sources exhibit spatial correlation within a frame and temporal correlation across frames. A DCT transform is often a key step of video coding to remove redundancy. If we divide a frame into 8×8 blocks, and plot the energy distribution of the block DCT coefficients, we typically see very non-uniform distributions (Figure 1). The energy often spreads across 5 or 6 orders of magnitude, and the high energy end drops very quickly.

The non-uniform distribution implies statistical features, and hence remaining redundancy, in the DCT coefficients. If left as is, such residual source redundancy is considered harmful, because there would be more information to transmit, with less power per unit of information. Therefore, traditional video compression schemes follow DCT with entropy coding to further compress the DCT coefficients. The high-entropy coefficients naturally require more bits to be fully represented. The end result is that the energy distribution over each bit is the same and the encoded source appears Bernoulli distributed across bits.

Unequal error protection for the source is often a key element of video coding. Each DCT coefficient has a different contribution to the total distortion and should be protected accordingly. Entropy coding could be viewed as implicit unequal error protection, precisely because more bits are allocated to more complex, high-entropy signals. Since all bits are equal, more bits imply more power needed for transmitting the complex signal, which means more protection.

Channel. A MIMO-OFDM link can be viewed as a set of narrowband MIMO channels for each subcarrier of OFDM. Each subcarrier channel can be represented with a channel matrix H. We can obtain its singular value decomposition (SVD), $H = U\ S\ V^H$, where U and V are two unitary matrices, S is a diagonal matrix with the singular values of H on the diagonal, and $(\cdot)^H$ means conjugate transpose, or Hermitian. Each non-zero singular value s_i corresponds to a subchannel i with channel gain s_i^2. The diagonal form of S means the subchannels are mutually orthogonal. In general V is not diagonal, and neither is H, which means we cannot directly leverage the mutually orthogonal subchannels when sending a signal vector $\mathbf{x}$ unless all s_i are the same. If, instead, we always transmit $V\mathbf{x}$, or *precode* the signal vector $\mathbf{x}$ with a precoding matrix V, then the received signals would be mutually orthogonal and can be decoded after a rotation, i.e., multiplying by U^H. Therefore, with SVD based precoding for each subcarrier, a MIMO-OFDM link can simply be viewed as a set of independent subchannels.

Figure 2 shows sorted subchannel gains for a few 3×3 MIMO-OFDM links, with 3 antennas on either end of the link. Each link is labeled (<source ID> <destination ID>). We can also see a large spread between the strongest and the weakest subchannels, though faster dropoff at the low gain end. Having more antennas increases the spread but flattens the high end slightly.

The diverse subchannels naturally merit different quantities of information each. This is the overarching goal of channel coding, so that the energy per bit sent on any subchannel would appear the same. Consequently, rate increase across a MIMO-OFDM channel is no longer monotonic with the overall channel SNR, unlike for single-antenna channels, which are scalar.

Source-channel separation and challenges. With fully random source bits and fully random channel bits, Shannon's source-channel separation theorem then shows we get the optimal result.

However, a capacity-achieving channel code must have an infinite block length. Given wireless channels are error prone, optimally loading source bits to each subchannel, along with sufficient protection bits, is impractical. Furthermore, the transmitter must know the exact subchannel gains and also notify the receiver how to decode on each subchannel. The former is susceptible to channel variation and the latter step would incur a high overhead.

2.2 Issues with existing approaches

Naturally, jointly performing source and channel coding would resolve some of the issues above, by retaining some source redundancy as channel protection. Such schemes typically still design separate, but matching codes for the source and the channel.

Instead, two recent systems, SoftCast and FlexCast, each designs a single code that compresses and protects simultaneously. SoftCast builds on an analog code that adjusts the compression-protection tradeoff with power allocation, whereas FlexCast achieves a similar goal with bit allocation.

For single-antenna channels, both schemes were shown to achieve almost optimal unicast performance, although SoftCast was designed for broadcast. However, directly running these over MIMO, without precoding, cannot avoid interference between received signals on different subchannels.

Even with precoding, neither distinguishes between different subchannel gains. Note also that FlexCast is currently built without knowledge of the channel, whereas precoding requires channel information. Once there is channel information, the source could narrow down the transmission rate to within a few choices, and the rateless code in FlexCast would be largely unnecessary.

Although there have been other digital unequal error protection schemes over MIMO or MIMO-OFDM (e.g., [25]), they are typically based on the scalable video coding (SVC) extension of H.264 [22]. These schemes divide the source into layers of different importance levels and allocate the most important layer to the best spatial subchannel. There is strong dependency between the layers. If the most important layer is not correctively decoded, any further layers are useless. Furthermore, there are usually more subchannels than layers. Limited digital modulation choices could constrain the performance when a layer needs to be transmitted over several heterogeneous subchannels.

2.3 Source-channel similarities

We observe a few similarities between the operations of source and channel coding, and therefore a joint source-channel scheme that exploits the synergy between the two is likely to work well.

First, SVD based precoding for a MIMO channel can be viewed as compacting energy to diagonalize the channel matrix, whereas DCT at the source is also expected to diagonalize the correlation matrix between source pixels. Without precoding at the channel, the received spatial signals would interfere with one another and hurt decoding performance, as some signal strength is used to cancel out the inter-signal interference before decoding. With correlation between the source components, power or bit allocation to them would be suboptimal. Therefore, it is important to avoid correlation at both the source and the channel.

Second, Figures 1 and 2 show a similar spread between the highest- and lowest-energy components, although the dropoff rate behavior differs. Therefore, it seems natural to match both sides, so that the high-energy DCT components are transmitted on the high-gain subchannels to avoid them acting against each other. Furthermore, the large number of subchannels allows fine-grained error protection levels.

Third, power weights can be allocated to transmitted source components to minimize the overall distortion. On the other hand, power allocation between different subchannels would also affect the achieved channel rates, which would eventually translate to the reconstructed source distortion based on a suitable rate-distortion function. Therefore, a joint source-channel power allocation strategy could optimize the overall recovery performance.

3. ParCast DESIGN

3.1 Overview

ParCast encodes and transmits a video sequence via a series of linear transforms.

The source encoder first computes 3D-DCT [10] coefficients for each 4-frame group of pictures (GOP). These DCT coefficients are grouped into chunks, the total number of which matches that of the subchannels in the MIMO-OFDM link, and a variance is computed for the energy of each chunk. The chunks are assigned to subchannels such that high-energy DCT coefficients would be sent on high-gain subchannels in successive packets, which is a permutation transform. Power whitening is then performed per chunk to even out power distribution.

The power allocation stage formulates an optimization problem to minimize the total distortion, and scales the magnitude of the DCT coefficients in each chunk by a weight. These power weights are computed by taking into account both the variance of the DCT coefficients and the corresponding subchannel gain.

Pairs of coded values are combined into complex symbols. Those for different spatial subchannels on each subcarrier are then precoded, so that they will be transmitted on orthogonal spatial channels and arrive at the receiver uninterfered by other spatial streams. The precoded values are then transmitted without further channel coding over the MIMO-OFDM channel.

The encoding and transmission process can be written as:

$$\mathbf{Y} = H\ Q\ G\ M\ R\ \mathbf{X} = C\ \mathbf{X} \qquad (1)$$

where H is the overall channel matrix of all the subcarriers stacked into one, Q is the precoding matrix, G is a diagonal matrix with the power weights on the diagonal, M represents per-stream whitening, R implements the mapping between DCT coefficient streams to subchannels, and $\mathbf{X}$ are the DCT coefficients.

The decoder does not perform regular MIMO processing to first decode the received signals on each antenna. Instead, it forms an aggregate matrix with the received signals on all the spatial and frequency domain subchannels, before invoking a Linear Least Square Estimator (LLSE) decoder to recover the DCT coefficients. Then, it reconstructs the original frames by taking the inverse of the 3D-DCT.

3.2 Effects of a MIMO-OFDM channel

Per-subcarrier view. Let us consider a 2×2 MIMO-OFDM link as an example. By design, the signals on different OFDM subcarriers are independent from one another. On a particular subcarrier, each transmit antenna can send a complex symbol composed of two real values, and therefore 4 real values can be sent at a time. Let $H_{complex}$ be the 2×2 complex channel matrix for the subcarrier, with each matrix element h_{ij} to denote the path gain from transmit antenna j to receive antenna i. Let $x_i = m_{i1} + j\ m_{i2}$ be the complex symbol on the i^{th} transmit antenna, and y_i be the complex symbol received on the i^{th} receive antenna. Disregarding channel noise for now, we have

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{pmatrix} \begin{pmatrix} m_{11} + j\ m_{12} \\ m_{21} + j\ m_{22} \end{pmatrix}$$

Separating the real and imaginary parts on both sides:

$$\begin{pmatrix} \Re(y_1) \\ \Im(y_1) \\ \Re(y_2) \\ \Im(y_2) \end{pmatrix} = \begin{pmatrix} \Re(h_{11}) & -\Im(h_{11}) & \Re(h_{12}) & -\Im(h_{12}) \\ \Im(h_{11}) & \Re(h_{11}) & \Im(h_{12}) & \Re(h_{12}) \\ \Re(h_{21}) & -\Im(h_{21}) & \Re(h_{22}) & -\Im(h_{22}) \\ \Im(h_{21}) & \Re(h_{21}) & \Im(h_{22}) & \Re(h_{22}) \end{pmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{21} \\ m_{22} \end{pmatrix}$$

Essentially, the vector $(m_{11}, m_{12}, m_{21}, m_{22})^T$ has traversed an equivalent channel represented by the real matrix, H_{real}, that can be derived from the complex channel, $H_{complex}$.

Precoding at the channel. Recall that precoding based on the Singular Value Decomposition (SVD) can decorrelate the spatial subchannels to be mutually orthogonal. Since we can now view the channel as either a real or a complex matrix, there are two options for calculating the precoding matrix. The SVD of both, $H_{real} = U_{real}S_{real}V_{real}^H$ and $H_{complex} = U_{complex}S_{complex}V_{complex}^H$, are closely related. There are twice as many singular values in S_{real}, but just the same numbers from $S_{complex}$ and each with a duplicate copy. Since a singular value represents a subchannel, each *complex subchannel* of $H_{complex}$ corresponds to two equal gain *real subchannels* of H_{real}. V_{real} and $V_{complex}$ are related similar to H_{real} and $H_{complex}$.

Therefore, we simply compute $S_{complex}$ and $V_{complex}$, and convert them to S_{real} and V_{real} for later steps.

The overall link. For the overall link, we can stack together all the individual subcarrier matrices to form a large block diagonal matrix. Since each chunk is mapped to a subchannel on a subcarrier, we can still decode on each subcarrier separately before combining the chunks to reconstruct the video frame. However, the per-chunk whitening creates dependency between the symbols on the same subchannel in different packets (time slots), and therefore we should first stack the matrices across time on the same subcarrier.

3.3 Managing bandwidth requirement

We determine the number of source chunks per video frame based on the number of real subchannels and the available channel time, and then compute the required number of time slots to send all data in the source chunk. For each chunk, we calculate the variance of the energy of all data, λ. The λ is the metadata per chunk to be used at the decoder. More chunks would mean more metadata, which could help improve the reconstruction performance though at a higher overhead. That said, we need only enough chunks so that the distribution of λ has the desirable fast dropoff behavior as shown in Figure 1.

The product of the number of OFDM symbols, the number of packets (i.e., time slots), and the number of chunks per GOP reflects the bandwidth requirement of sending a GOP.

Insufficient bandwidth. When there are insufficient time slots, we should discard the least important chunks, since they contribute least to the overall reconstruction quality.

Bad subchannels. If some subchannels are very bad, it would be better to skip those. This way, the power budget can be expended on the remaining, more important chunks in higher gain subchannels. If this causes bandwidth shortage, we simply discard the least important source chunks.

Redundant bandwidth. If there is more bandwidth than we need to transmit the video sequence, we could also skip the bad subchannels and allocate the power to the remaining good channels. With each skipped bad subchannel, ParCast can benefit from a diversity gain.

3.4 Error protection and power allocation

The distribution of per source chunk energy variance, λ, is analogous to that of the 8×8 block DCT coefficients in Figure 1. If the chunk data are sent verbatim, their energy helps guard against the channel noise and serves as implicit channel coding. However, it is likely that the high-energy source components would consume too much transmission power budget and leave the remaining components more sensitive to the channel noise. Therefore, we need to re-distribute the power among source components before transmission.

Power weights. With the precoding operation at the encoder using V_{real} and equalization based on the matrix U^H

at the receiver, the MIMO-OFDM channel is separated into a set of orthogonal subchannels. The optimization problem in ParCast is to minimize the expected mean square error (MSE) under the transmission power constraint with budget P. Assuming inverse decoding, i.e., using C^{-1} from Equation (1), we can formulate the problem as:

$$\min \quad MSE = \sum_i \frac{\sigma^2}{s_i^2 g_i^2} \tag{2}$$

$$\begin{aligned}\text{subject to:} \quad & trace\ E[(QG\mathbf{x})(QG\mathbf{x})^T] \\ =& trace\ E[(G\mathbf{x})(G\mathbf{x})^T] \\ =& \sum_i g_i^2 \lambda_i \leq P\end{aligned}$$

where σ^2 is the average noise power, s_i is the singular value on subchannel i, Q is the channel precoding matrix, and G is the diagonal, power weight matrix, with g_i as its i^{th} diagonal entry. By using Lagrange multipliers, the solution is:

$$g_i = \sqrt{\frac{P}{\sqrt{\lambda_i s_i^2} \sum_j \sqrt{\lambda_j / s_j^2}}} \tag{3}$$

and the derived MSE is

$$\frac{\sigma^2}{P} \Big(\sum_i \sqrt{\lambda_i / s_i^2} \Big)^2 \tag{4}$$

The MSE (4) is determined by the mapping of λ to s, i.e., the matrix R in Equation (1). By the Rearrangement Inequality [12], the best R lets the λ_i follow the same sorted order as s_i. Hence, the chunk with the largest variance is assigned to the highest-gain subchannel.

Strictly speaking, the above is an approximation for when the optimal LLSE decoder is used. This is because inverse decoding could magnify the channel noise. When the channel condition is good, however, the two decoders have comparable performance, and the optimization of MSE of inverse decoding is much simpler to solve.

If we re-formulate the optimization using LLSE instead, the derived g_i depend on the channel noise σ^2 [15]. If λ_i/s_i^2 is too low, the corresponding g_i would be zero in the new optimization result. This means the subchannel and the corresponding chunk should be discarded, and their power returned to the overall power budget for others. This is analogous to the water-filling power allocation strategy in MIMO systems to achieve the channel capacity.

Per-chunk whitening. The data within a chunk normally carry different amounts of energy. If these are assigned to OFDM symbols directly, there could be a large power difference between symbols and cause issues at the receiver. Therefore, we first mix data within a chunk with a random orthogonal matrix to achieve the same effect as a Hadamard transform[1], such that the average power per OFDM symbol is the same. This also ensures the same average power across transmit antennas and eases the requirement on the dynamic range of the power amplifier at the transmitter.

3.5 Decoder

Given the linear encoder, we use the LLSE decoder at the receiver, which is the optimal decoder [15].

$$D_{LLSE} = \Lambda C^T (C \Lambda C^T + \Sigma)^{-1} \tag{5}$$

where C is the overall encoder from Equation (1), and Σ and Λ are both diagonal matrices. The i^{th} diagonal element of Σ is the channel noise experienced by the packet carrying the i^{th} row of the received $\mathbf{Y}$. The i^{th} diagonal element of Λ is λ_i/s_i^2, where λ_i is the variance of the i^{th} chunk and s_i is the gain on the corresponding subchannel.

After decoding the DCT coefficients, ParCast reconstructs the original frames by taking the inverse 3D-DCT.

3.6 Managing metadata

Source metadata. The λ per chunk is the metadata from the video source, which should be sent to the video decoder. The video sender transmits this metadata per GOP in a regular packet at the lowest rate before sending encoded video data. The overhead is less than 0.005 bits per pixel.

Channel state information (CSI). The video sender needs to know the channel detail to perform the chunk-subchannel mapping, power allocation, and precoding. Therefore, the channel state information can be viewed as metadata from the channel. Since the upcoming WiFi standard 802.11ac and LTE both expect to use channel dependent precoding, it is reasonable to assume the availability of CSI.

The CSI updates are *cumulative*. Since later packets would be precoded, the video receiver would then measure a precoded channel, $H_{measured} = H_{actual}\ V_{previous}$. $H_{measured}$ has the same singular values as the actual channel H_{actual}, so that we can assign chunks to subchannels and calculate power weights directly. However, typically $V_{actual} \neq V_{measured}$, so $V_{measured}$ alone cannot be used as the precoding matrix. We have two options: (1) the sender computes the actual channel as $H_{actual} = H_{measured} \cdot V_{previous}^H$, get V_{actual} as the new precoding matrix; or (2) the sender gets V_{new} from $H_{measured}$, and use $V_{previous} \times V_{new}$ as the precoding matrix. Although $V_{actual} \neq V_{previous} \times V_{new}$ in general, the two sides differ by a multiplication by a unitary block diagonal matrix, and achieve the same precoding effect. We have verified both options and currently use the first one.

Currently, the video receiver would measure the channel using the regular channel estimation technique, e.g., that for 802.11n MIMO-OFDM packets. It then feeds the CSI back to the sender by sending a regular packet. A CSI update every 100 ms is normally considered acceptable overhead. If there is also traffic from the receiver to the sender, we could leverage channel reciprocity by letting the sender estimate the channel. Therefore, the CSI update overhead can be easily managed.

3.7 ParCast vs SoftCast

ParCast is inspired by SoftCast and follows a similar linear transform process, but with several modifications and additions. Although the same techniques are adopted in several steps, they serve different goals.

The main stages in SoftCast include 3D-DCT, power weighting, and per-frame whitening via a Hadamard matrix, transmission over an uncoded OFDM channel, and decoding using LLSE. The matrix view of the overall encoding process is:

$$\mathbf{Y} = H\ H_a\ G\mathbf{X} = C_{SoftCast}\mathbf{X} \tag{6}$$

where H is the overall channel matrix, H_a is the Hadamard matrix, G is the diagonal power weight matrix, and $\mathbf{X}$ is the DCT coefficients.

ParCast also starts with 3D-DCT, not only because it achieves compression, but it allows us to generate equal-sized, independent chunks. Otherwise, different numbers of OFDM symbols would be required to transmit the chunks,

[1]Hadamard matrices can only have certain dimensions due to their orthogonal property, whereas our data can be of arbitrary dimensions.

necessitating a further power re-distribution. The independence between chunks allows us to gain performance from any successfully delivered and decoded chunk. We will return to this point in Section 5.

SoftCast treats each channel as a whole, whereas ParCast specifically optimizes for each subchannel individually. The power allocation formulation and the LLSE decoder in ParCast incorporate channel details, whereas those in SoftCast consider the source property only. At low SNR, ParCast naturally discards the worst subchannels and the corresponding source chunks. The diversity gain thus achieved mitigates the lack of error correction capability of analog modulation. This is unlike SoftCast, which can only send duplicate data to improve reliability if there are extra time slots.

SoftCast performs Hadamard *across DCT chunks* for each GOP to guard against packet loss, whereas the corresponding transform in ParCast operates *within a chunk* and mainly balances power across OFDM symbols for the same chunk and power across transmit antennas. Notice that dividing the source into chunks turns $\mathbf{X}$ in Equations (6) and (1) into a diagonal matrix. The overall encoding matrix, C, in Equation (1) is diagonal after a rotation, but $C_{SoftCast}$ is not even if H is. H_a specifically mixes $\mathbf{X}$. Therefore, any source correlation will manifest as channel correlation.

SoftCast uses analog modulation to achieve graceful degradation across different broadcast receivers, whereas ParCast uses this per subchannel of a unicast link to harness the channel capacity with a single modulation. ParCast effectively turns a unicast session into many parallel sessions, as if over a broadcast channel.

4. IMPLEMENTATION

4.1 ParCast implementation

We divide the whole system into the video codec at the application layer and physical layer operations.

At the video sender side, the video encoder performs the 3D-DCT transform, generates a whitened video data stream and the variance per chunk, and passes these to the PHY module. The number of chunks is determined based on the bandwidth availability. The PHY module performs the chunk-subchannel mapping, power allocation, and precoding, before grouping enough OFDM symbols into individual packets for transmission. In addition, the PHY module on the video sender sends the source metadata and processes the channel state feedback from the video receiver to calculate the SVD of each subcarrier channel matrix.

Note that although we currently implement the entire source encoder at the transmitter, this is not strictly necessary. The 3D-DCT transform is independent of the channel and could easily take place on a remote video server connected with the wireless transmitter over the wire. If the server is notified of the number of chunks, the division into chunks, chunk variance calculation, and the per-chunk power whitening could also be performed remotely, leaving only the PHY operations to be performed at the wireless transmitter.

At the video receiver, the PHY module performs channel estimation, compensation for carrier frequency offsets (CFO), and OFDM pilot phase tracking without compensating for the channel, before passing the received complex samples to the video decoder. It also decodes the control packet with the source metadata and feeds back the estimated channel matrices to the video sender. The video decoder then finds the LLSE solution to the received signal vector, decodes video data in each chunk, and finally performs the inverse DCT to reconstruct the video source.

We use Sora [28] as our experimental platform. Since the current Sora platform can support real-time 20 MHz wideband packet transmission and reception, but not quite single-device MIMO, we need to emulate MIMO with multiple single-antenna boxes. Due to our analog modulation, this means that we would need to send raw signal samples to the sending nodes separately, and copy the received signals from the receiving nodes to one box, both via Ethernet. The latency from moving the samples across the network makes it currently infeasible to run the whole system in real time. However, since CSI updates can be sent using a single antenna, we can still run the channel dependent components in real time to assess the latency of PHY processing.

Therefore, we currently implement the PHY modules using Sora SDK 1.5 to be run in real time. The actual video codec is implemented in Matlab and interfaces with the PHY module via the generated video data as real values and the received complex samples. The video data packets follow 802.11n like packet structure, so that we can leverage techniques for CFO compensation, channel estimation and precoding in 802.11n. Unlike in 802.11n, however, the pilot subcarriers for each antenna are used alternately across different OFDM symbols, similar to the approach in [18], so that we can update all the estimated channel coefficients for one transmit antenna per symbol. Noise is estimated from the error vector magnitude of the signal field in the packet preamble, and passed to the video decoder to be plugged into the LLSE decoder.

We also implement a channel simulation that could act in place of the Sora based PHY modules but replay measured channel traces. This allows us to study a larger variety of channel instances and compare different schemes more easily and fairly.

4.2 Schemes for comparison

The implementation of ParCast can be viewed as a superset of SoftCast. In addition, we consider two schemes for comparison: (1) a version of omniscient H.264/MPEG-4 [19] over 802.11n like channel rates (Omni-MPEG) as an optimal case, and (2) Layered Scalable Video Coding (Layered SVC) [22] with unequal error protection as a practical case. Both are based on MPEG-4.

MPEG-4 takes a sequence of raw frames and encodes it into three types of frames. The *I-frame* is an independently encoded still image, used as a reference frame. A *P-frame* encodes changes in the image from the previous frame, whereas a *B-frame* includes changes from both the previous and the future frames.

The raw I-frame is first divided into blocks. The blocks in consecutive raw frames are then compared to each other to identify the closest matches and the corresponding motion vector. The differences between the matching blocks are then stored in the P- and B-frames. The blocks for all frames then go through DCT transform, quantization, and entropy coding for further compression.

Omni-MPEG. Typically, the source encoder needs to know the expected channel throughput in order to select an appropriate quantization levels, and the physical layer transmitter should determine the most suitable bit rate accordingly. We alter the standard MPEG-4 encoding and transmission process to optimally choose the source and channel rates based on the given CSI.

We implement MPEG based on the reference implementation JM16.1 [3]. The source first generates encoded data under different Quantization Parameter (QP) values. We derive a table of PSNRs under various QPs, the corresponding data stream size and the channel throughput required.

Data is transmitted using 802.11n like modulation and coding, and we select the channel rate based on the effective SNR of the given channel [11]. Depending on the number of antennas on both sides, a sender could choose between sending 1 to 4 concurrent spatial streams (i.e., packet fragments) at a time. To use a good rate, we precode the transmitted streams and allow a different modulation and coding combination per transmitted spatial stream. This allows a higher overall rate than those provided by unequal modulation across streams in 802.11n [5]. If the worst spatial stream is too weak to support even BPSK, we allocate the power to a stronger spatial stream instead, using a high rate where possible. If any stream is decoded incorrectly, the entire packet is dropped at the receiver and retransmitted.

To compare Omni-MPEG with ParCast under the same bandwidth constraint, we select an encoded data size which requires a transmission time closest to ParCast, look up the appropriate QP, and select the corresponding PSNR as the performance of Omni-MPEG.

Layered SVC. In order to adapt to varying channel conditions, the H.264 scalable video coding extension provides spatial, temporal, and quality scalability. For SVC, the entropy coded bits are separated into layers with varying importance levels. The base layer data is always expected to be correctly received to achieve an acceptable video quality, while a higher quality is derived from the enhancement layers. There have been several proposals for unequal error protection between layers.

The SVD-based video transmission scheme [25] is an example design for a narrowband MIMO channel. The base layer would normally be assigned to the best spatial subchannel and sent using QPSK. The enhancement layers can use denser modulations. Power is carefully allocated to the subchannels to ensure a target error rate for the base layer and the highest possible rates for other layers. We extend it to run over 802.11n like OFDM for our comparison.

The source codec is implemented based on JSVM [6]. We use quality scalability, which adopts a large QP for the base layer, and smaller QPs for enhancement layers. Each enhancement layer is generated by using a smaller QP to encode the quantized residue from the previous layer. Therefore, there are strong dependencies between layers. When one transmitted packet can not be correctly decoded for the base layer data, the whole frame and any other frame that is dependent on the current frame has to be discarded.

The original proposal in [25] aimed to achieve an *uncoded* BER of 10^{-5} for the base layer and 10^{-3} for the enhancement layers. Our initial trials suggest these settings are too optimistic and would produce bad video quality. Therefore, we set the target *coded* BER for all layers to 10^{-5}. A Reed-Solomon code (239, 255) is used for all layers at the application layer, along with a $\frac{3}{4}$ channel code to provide better channel protection. We also skip the antenna selection suggested, since it often over-aggressively improves the enhancement layer quality at the expense of the base layer.

FlexCast. A comparison against FlexCast would be interesting but unfair. First, FlexCast is built to achieve a different goal. For our setting, the equivalent would be H.264 along with a capacity achieving channel code over MIMO-OFDM. Second, FlexCast was designed for single-antenna systems. Extending it to MIMO is beyond the scope of this paper. In particular, we are not aware of a rateless code for non-ideal MIMO channels, since MIMO rate increase is not monotonic with the overall channel SNR, as shown by the 802.11n rates [11].

5. EVALUATION

5.1 Experimental setup

Performance metric. We evaluate video delivery quality with the standard Peak Signal-to-Noise Ratio (PSNR) metric. $PSNR = 20\log_{10}\frac{2^L-1}{\sqrt{MSE}}$, where L is the number of bits used to encode pixel luminance, typically 8, and MSE is the mean squared error between all pixels of the decoded video and the original. We average the PSNR across frames to produce a single value for each video sequence.

Test videos. We select a few representative reference video sequences [1, 2] with different characteristics: *foreman* (little motion, large energy spread), *news* (even more stationary, large energy spread), *bus* (more motion, smaller energy spread), *mobile* (faster motion, larger motion area, even smaller energy spread), and *football* (significant motion, large energy spread). Figure 1 shows the energy distribution of the first frame of each sequence. The distribution of the variance of the chunk energy after the 3D-DCT is similar.

We use a video frame size of 352×288 pixels for all experiments. For a 2×2 link, we could send up to 2 values per complex symbol, 2 symbols per subcarrier, and 52 data subcarriers, or 208 values total in a single 802.11n OFDM symbol. We currently send 192 values (i.e., 48 chunks per video frame) in each OFDM symbol and skip the worst few subchannels, so that an entire video frame takes 528 OFDM symbols. For a fair comparison with other schemes, we define packet size in terms of number of OFDM symbols. Similarly, we use 72 chunks per video frame for a 3×3 link, and send 288 real values in one OFDM symbols. Calculations show that, if all subchannels within and between the corresponding 2×2 and 3×3 links have the same gain, the PSNR will increase by 0.5 dB from 192 chunks to 288 chunks, and 0.8 dB from 192 to 396 chunks.

We take 32 frames per sequence for microbenchmarks and 200 frames per sequence for the mobile experiments.

Sora setup. We use four single-antenna Sora boxes to emulate a 2×2 MIMO link, and move the antennas to different locations to get different channel instances. The carrier frequencies of all four radios are configured to be within a few hundred Hz of one another. This is the precision we can achieve given the configuration tool, and we have verified that pilot tracking can correct residual carrier frequency offsets between each transmit-receive pair sufficiently well.

We calibrate the CPU clocks of all boxes offline and use a SourceSync [18] like approach to achieve synchronized transmissions from the two transmit antennas.

Channel traces. We use two sets of channel traces of 3×3 802.11n MIMO-OFDM links from previous work [11] for trace-driven simulations, one including around 120 stationary links and the other a mobile trace. These were collected on commodity Intel `iwl5300` NICs. Figure 2 shows

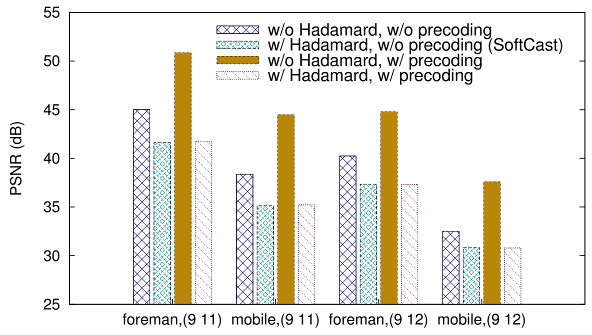

Figure 3: Effects of Hadamard and precoding.

the squared singular value distributions for example links. Each link is identified by (<source ID> <destination ID>).

For each stationary trace, there is one packet approximately every 6 ms. Experiments show that results differ mainly for whether there is CSI feedback delay, and is not sensitive to the delay period. Therefore, we simply present the results for one-packet delay, or around 6 ms. For the mobile trace, we select a CSI sample approximately every 10 ms to ensure channel variation.

5.2 Microbenchmarks from Sora experiments

The main purposes of the Sora experiments are to validate the simulation implementation and assess the complexity of several operations. Subsequent experiments are all done using trace-driven simulation, since that makes comparison fairer and the mobility experiment more manageable.

Minimum CSI delay. In our current implementation, the time cost for an explicit CSI update is less than 1 ms, which is much smaller than the typical update period.

SVD calculation latency. Given all subcarriers are independent, we only need to calculate SVD for 52 4×4 real matrices. This takes around 300μs for a box with a 3.40 GHz CPU frequency.

Residual CFO. Since the two numbers in a complex symbol are correlated, any residual CFO after pilot tracking could induce an error in the decoding process. However, our experiments show that the error is insignificant at low channel SNR, and within 0.5 dB even for a 200 Hz residual CFO at high SNR.

Limited sample precision. Although the power allocation step results in a large energy spread in the frequency domain, the IFFT of OFDM has a mixing effect so the time domain signal energy is better evened out. Therefore, limited precision of transmitted time-domain samples causes an error of around 0.2 dB to the final PSNR.

Computational complexity. The computational overhead of 3D-DCT in ParCast is the same as for SoftCast, and was evaluated previously [13]. The main overhead in the remaining steps of ParCast is in computing the precoding matrix (SVD), evaluated above. Current wireless transmitters need to do this anyway in order to apply channel-specific precoding. Precoding in a broad sense (i.e., channel independent spatial expansion or mapping) is in the 802.11n standard and already performed on commercial cards (e.g., by default on Intel `iwl5300`), which shows that the overhead is acceptable.

5.3 ParCast microbenchmarks

We run a few benchmarks to assess the individual contributions of the operations of ParCast, one of which also serves as a comparison against SoftCast. We use two representative video sequences, *foreman* and *mobile*, which feature a large and small spread of energy distribution respectively. We also choose two representative 3×3 channel traces, links (9 11) and (9 12), which feature a very small and large spread of squared singular values respectively. The CSI is delayed unless otherwise stated.

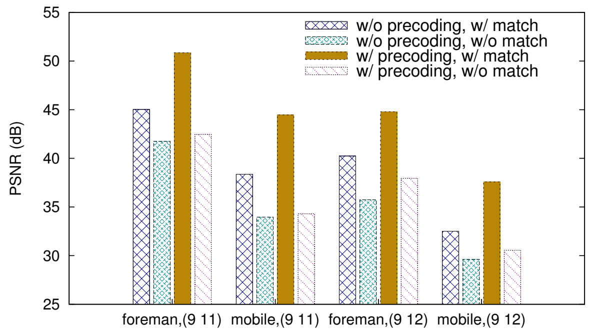

Figure 4: Effects of matching important chunks with high gain subchannels.

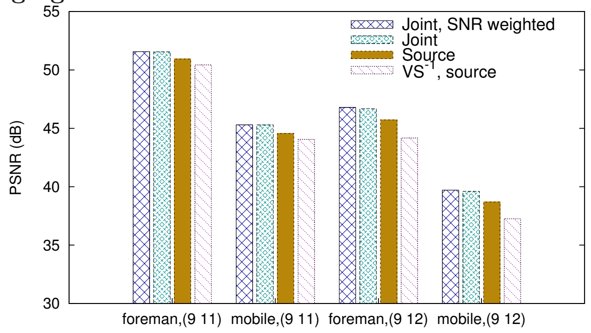

Figure 5: Effects of joint source-channel power allocation.

Spatial correlation over MIMO. SoftCast includes a Hadamard transform to even out energy between packets and provide resilience to packet loss. However, applying it naively over a MIMO channel would cause interference between the signals on different spatial subchannels of the same subcarrier and degrade the decoding performance. Instead, a subcarrier-specific precoding matrix should be used.

Figure 3 shows that Hadamard does not help when we *do not* precode the channel, but hurts when we *do* precode[2]. Without precoding, the PSNR without Hadamard is on average 2.8 dB higher than with. With precoding, the PSNR without Hadamard is a further 5.4 dB higher. The reason is that Hadamard acts as *random* precoding, which is statistically equivalent to not precoding at all [29]. Note that the source-channel matching is implicit for 'without Hadamard', but there is no matching for 'with Hadamard'. All are without the joint source-channel power allocation but use the source aware power allocation as in SoftCast.

The importance of matching. We next show the importance of sending the more important source components on high gain subchannels. Since this can be affected by whether there is precoding, we compare between ParCast with different combinations of with and without precoding and matching. Without matching means that chunks are simply allocated to subchannels on each subcarrier in a ran-

[2]We let SoftCast divide each video frame into 72 chunks.

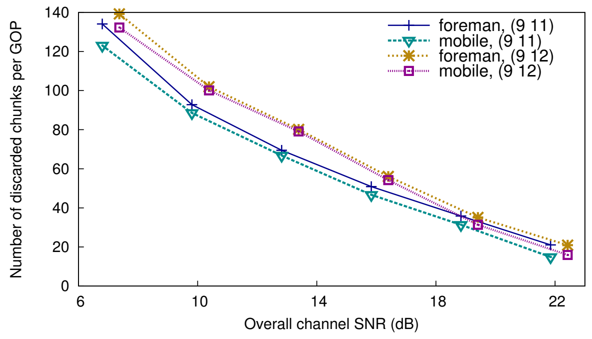

Figure 6: ParCast discards source chunks if needed.

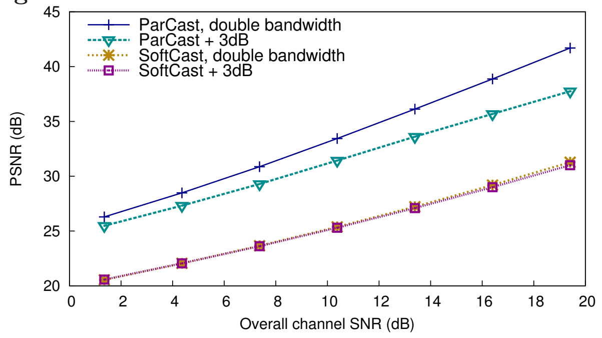

Figure 7: Effects of extra bandwidth.

dom order. Again, all variants use the source aware power allocation only.

Figure 4 shows that matching alone could improve the performance by around 3.5 dB even without precoding, and by 8.1 dB with precoding. Note that the bars for 'without matching' should not be compared directly across videos or channels due to their random nature.

There are some interesting similarities between Figure 3 and Figure 4. The *without precoding, with Hadamard* bar in Figure 3 is similar to the *without precoding, without match* bar in Figure 4. The former is slightly better due to Hadamard mixing the data for different subcarriers. The *with precoding, with Hadamard* bar in Figure 3 and the *with precoding, without match* bar in Figure 4 are also similar, but the latter could be better if the data chunks happen to match subchannels *on the same subcarrier*.

Joint source-channel power allocation. With precoding and matching, we next compare ParCast with source aware power allocation only and with joint source-channel allocation. Note that, in effect, the source-only allocation in SoftCast assumes all subchannels are equivalent, which would correspond to precoding with VS^{-1}, where V and S are from the SVD of $H = U\ S\ V^H$. Therefore, we also consider this case, i.e., precoding with VS^{-1} combined with source-only allocation. We use the ground-truth CSI for these experiments, and defer the discussion on the effects of CSI inaccuracy to the next subsection.

Figure 5 shows that, depending on the channel characteristics, the non-weighted joint power allocation outperforms the source-only approach by 0.6 to 0.9 dB. However, precoding with VS^{-1} is similar to using H^{-1}, and is especially bad for a link with a large singular value spread. As expected, the trend is the same for both videos for the same channel, since this comparison highlights the benefit from leveraging channel details.

The non-weighted joint power allocation is most effective under three conditions: (1) the source chunks exhibit a large energy variation, (2) the MIMO-OFDM channel is very spatially correlated and frequency selective, and (3) CSI is fairly accurate. The first condition is often met. The latter two result in a large variation between subchannel gains, which is a challenging case for MIMO-OFDM to work well. The joint power allocation strategy is thus a way to take advantage of otherwise challenging conditions, although the second condition is a double-edged sword when CSI is inaccurate.

The SNR weighted approach is most helpful at low SNR by skipping the worst subchannels and the corresponding source chunks. Figure 6 shows the number of chunks discarded per 4-frame GOP as the channel SNR lowers, assuming there is no spare bandwidth. Recall that a chunk-subchannel pair is discarded if its λ/s^2 is too low (Section 3.4). This is more affected by the chunk energy at high channel SNR, and by the subchannel gain at low channel SNR.

Extra bandwidth. Finally, we study the PSNR of ParCast and SoftCast when given extra bandwidth. Figure 7 shows the results for 16-frame *mobile* and Link (9 12). The trends for other videos and links are similar and hence omitted.

The '+3dB' lines show the performance of both schemes with exact bandwidth but twice the power. SoftCast can only repeat the same data when given double the bandwidth, which is equivalent to using twice the power but no extra bandwidth. In contrast, ParCast benefits from a subchannel diversity gain when given extra bandwidth, although the gain is less pronounced at low SNR when some bad subchannels (and source chunks) would be skipped anyway.

5.4 Comparison against alternatives

Overall. We first compare ParCast, Omni-MPEG, and Layered SVC over all stationary 3×3 links, with precoding and delayed CSI feedback. This represents the most common scenario. Figure 8 shows that ParCast often achieves a much higher PSNR than the other schemes, especially for *football*, which is a very challenging case for traditional video compression techniques. We next zoom into individual traces and video sequences.

Channel dimension. For ideal channels, with equal gain on all subchannels, an $N \times N$ MIMO system would scale the SISO capacity by approximately N times, or reduce the bandwidth requirement by a factor of N. However, this N factor is hard to achieve for real channels. In fact, more antenna may not help if not used optimally. We therefore compare the performance of all three schemes for 2×2 and 3×3 configurations for the same source-destination pair, both with delayed CSI. Note that the bandwidth requirement for 3×3 is only $\frac{2}{3}$ of that for 2×2 links.

We see in Figure 9 that ParCast normally achieves comparable performance for both 2×2 and 3×3, whether with a large subchannel gain spread or not. This shows that ParCast can harness the capacity scaling potential of the additional channel dimensions. The slight drop from 3×3 to 2×2 is due to different numbers of chunks and how good the 2×2 part compares with its 3×3 counterpart. Layered SVC often benefits from 3×3 by being able to use a denser modulation for the enhancement layer than in 2×2. Omni-MPEG is more sensitive to the channel detail. The 2×2 channel for (9 11) is *nicer* than its 3×3 counterpart by allowing high rates to be selected at a higher accuracy, and hence the PSNR is much higher for 2×2 than for 3×3.

It is interesting to note that ParCast often outperforms

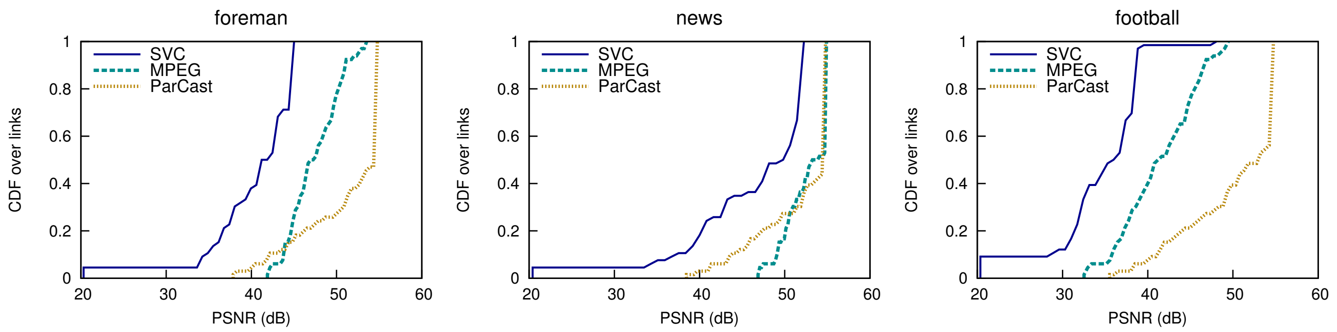

Figure 8: PSNR for H.264, SVC, and ParCast for all stationary traces.

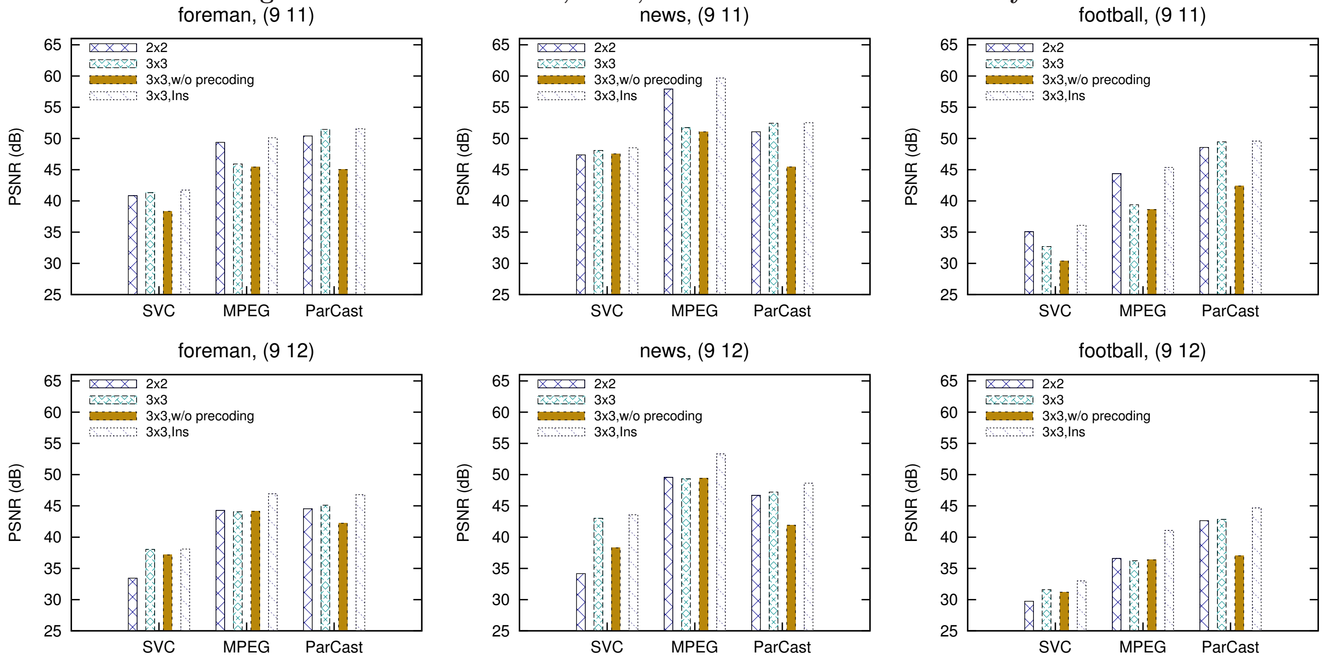

Figure 9: Effects of channel dimension and CSI accuracy.

Omni-MPEG by a few dBs, except for the video *news*. This is because the digital modulations used for Omni-MPEG cannot easily achieve the channel capacity. In contrast, the analog modulation approach in ParCast is very helpful in MIMO-OFDM situations to fully utilize the diverse channel gains without much overhead. Otherwise, the transmitter would also need to signal to the receiver how to demodulate each subchannel, and even a FARA [17] like scheme may not work well. *news* is fairly compressible and favors traditional compression approaches. Even the non-capacity-achieving 802.11n like rates suffice for such compressible videos.

CSI accuracy. The main steps of ParCast, precoding, matching, and power allocation, all depend on the SVD of the channel matrix, which is sensitive to the CSI accuracy. Therefore, we next assess the effects of CSI accuracy on the overall performance. We first study the 3×3 stationary traces and then consider the mobile trace in the next part. Figure 9 compares the performance with delayed (the 3×3 bar) and instantaneous CSI (labeled 3×3*, Ins*).

For the stationary traces, CSI inaccuracy is mainly caused by estimation noise and precision errors. The inaccuracy in our trace indicates the amount of error margin manageable on commodity NICs.

The issue with delayed CSI is that precoding would not be able to entirely decorrelate spatial streams. Our data show that such spatial correlation generally weakens the strongest spatial subchannel, but strengthens the weakest slightly. The larger the singular value spread, the larger the effects of inaccurate CSI. This could cause modulation selection to be too optimistic, which affects both Omni-MPEG and Layered SVC. Inaccurate modulation selection could cause decoding errors. Therefore, the Omni-MPEG performance is about the same with or without precoding (3×3 vs 3×3*, w/o precoding* in Figure 9).

In contrast, ParCast does not adapt the encoding of the chunk after selecting a subchannel, and the effective subchannel gain variation would just translate to slight variation in the mean squared error for the pixel. Hence, ParCast can degrade gracefully at a subchannel level. In particular, since Link (9 11) has a smaller singular value spread, the PSNR difference between ParCast with instantaneous CSI (the ground truth) and delayed CSI is only about 0.1 dB regardless of the video. For Link (9 12), the PSNR drop for ParCast due to inaccurate CSI is between 1.4 and 1.8 dB for the three videos. The bulk of that is due to precoding. If we use the ground-truth CSI for matching and power allocation, then precoding with delayed CSI causes ParCast to lose 1.1 to 1.4 dB in PSNR. The matching step alone loses about 0.2 dB in a similar comparison. SNR weighted power allocation alone is hardly affected, but can interact with the

Video	No delay		10ms		100ms		1s		5s	
	Ours	MPEG	Ours	MPEG	Ours	MPEG	Ours	MPEG	Ours	MPEG
foreman	48.9	46.8	43.6	43.5	42.2	43.3	41.8	43.2	41.4	42.7
mobile	42.8	40.5	37.4	35.6	36.0	35.3	35.6	35.2	35.2	34.6
news	50.5	52.2	44.8	48.4	43.2	48.2	42.7	48.1	42.3	47.5
bus	45.9	42.7	40.1	37.7	38.7	37.3	38.3	37.1	37.7	37.1
football	50.6	43.7	44.7	38.4	43.2	38.1	42.8	37.9	42.3	37.1

Table 1: PSNR (dB) for videos given different CSI delay, ParCast vs MPEG.

precoding step adversely to cause more PSNR degradation. In particular, the non-weighted joint power allocation evaluated previously can be worse than the source-only allocation.

As we saw in the previous subsection, matching chunk with subchannel is more important than additional channel-aware power allocation. Therefore, even without very accurate CSI, matching alone could still help as long as the relative subchannel gain order is not affected. Similarly, matching the base layer with a good subchannel also helps Layered SVC, but different layers carry different number of bits and require different modulations. Omni-MPEG over 802.11n means that the implicit unequal protection achieved in the entropy coding stage (Section 2) is not aligned with the channel, even with MIMO precoding. What does help is that precoding reduces the difference between the SNRs on the corresponding spatial subchannel across subcarriers.

Even though delayed CSI makes precoding less effective, it is still better than not precoding at all for both Layered SVC and ParCast. Without precoding, each received signal on an antenna has interference from other spatial streams. Reducing such interference could significantly improve the decoding performance.

Mobility. We next use ParCast and Omni-MPEG to deliver 200-frame video sequences over the mobile trace[3]. Inaccurate CSI is a main feature of the trace, but likely due to changing channel as opposed to estimation noise. If the channel itself changed too much, the matching would be wrong in ParCast, and power allocation based on the wrong matching would hurt further. We consider CSI delay periods of 0 (No delay), 1 packet (10 ms), 10 packets (100 ms), 100 packets (around 1 s), and 500 packets (around 5 s). Trials suggest that the joint power allocation in ParCast and precoding for Omni-MPEG are sometimes too aggressive for the mobility case. Therefore, we only use the source-aware power allocation for ParCast[4] and no precoding for MPEG.

Table 1 summarizes the results. The error margin is about ± 0.6 dB for MPEG and ± 0.9 dB for ParCast for any delay period. We see that the main difference is again between whether there is delay. Omni-MPEG degrades less with CSI delay, though it cannot benefit from more accurate CSI information. ParCast performance degrades slightly more.

If we compare the ParCast PSNRs for 1s delay with no delay, matching alone incurs about 1 dB loss, and the remaining loss can be attributed to inaccurate precoding. The bulk of the drop is likely caused by the estimation error on the Intel NICs. Sora based experiments show PSNR degradation of 1 to 2 dB with delayed CSI.

6. DISCUSSION AND FUTURE WORK

Limitation of analog modulation. Analog modulation lacks error-detection or correction capabilities. However, several design decisions in ParCast can mitigate these.

First, ParCast is designed for heterogeneous MIMO-OFDM channels. Our results suggest analog modulation can work with MIMO-OFDM in complementary ways. At low SNR, MIMO-OFDM provides better reliability and error resilience via spatial and frequency diversity. At high SNR, analog modulation is an easy mechanism to harness the channel capacity. The latter is more useful a setting for MIMO for getting higher rates, which motivates HD video streaming.

Second, ParCast encodes video frames independently, so that the distortion for any GOP does not affect other GOPs. Distributed video coding has a similar property and could be an alternative. If instead, the video frames are encoded based on one another as the B- and P-frames in MPEG-4, errors might accumulate over frames due to the distortion in reconstructing the reference frame.

Third, the DCT and precoding steps in ParCast aim to remove correlation between any source chunks and between subchannels. Without either, there would interference between received signals. The LLSE decoder would need to first cancel out the inter-signal interference during decoding and incur performance loss.

Source division. There are several other ways of dividing the source into layers to assign to subchannels, such as using the I-, B-, and P-frames in MPEG. The number of frames of each type is very source dependent and the size of different frame types can differ based on channel conditions. In contrast, the source coding stage of ParCast and SoftCast divides source independent of the source detail. Furthermore, ParCast allocates the same number of symbols to all subchannels, while the amount of decodable information naturally scales with the subchannel SNR. Therefore, the symbol modulation is also independent of the channel.

Source-channel dissimilarities. Since there is still disparity between the distribution of DCT coefficient energy and the subchannel gains as shown in Figures 1 and 2, there might still be more residual redundancy at the source than is effectively used at the channel. Moreover, 3D-DCT does not fully capture motion in the video. Further refinement to the 3D-DCT or LLSE could improve the overall performance. However, the former should be designed with the subchannel characteristics in mind similar to the joint source-channel power allocation step, whereas the latter could be more flexible based on actual channel conditions.

7. RELATED WORK

ParCast is related to work on unequal error protection, joint source-channel coding, and graceful degradation.

Unequal error protection (UEP) over MIMO-OFDM. As mentioned earlier, unequal error protection is often implicit in entropy coding. There have also been many explicit UEP schemes specifically for MIMO or MIMO-OFDM chan-

[3]Only 148 frames for *bus*, because the sequence is shorter.

[4]PSNRs with the SNR weighted joint power allocation are about 1 dB lower.

nels, for example, involving more forward error correction for the worst subchannels [20].

The fine-grained scalable (FGS) bit-stream extension of MPEG-4 can be used to generate many layers for different MIMO-OFDM subchannels [14]. However, FGS is rarely used in practice, and it assumes unequal digital modulation across subchannels, which is hard to implement in practice. Another proposal [26] is similar in principle, though with far fewer, coarse-grained layers and only the subchannels of a narrowband MIMO channel. Together with the scheme we studied earlier [25], these schemes mainly differ in how they divide the source into layers and the granularity of layers and subchannels used. Nevertheless, common to all, the layers are not independent, and an incorrectly decoded base layer would render all other layers useless.

ParCast is similar in matching importance source data with high-gain subchannels, but does so at a fine granularity and for independent source components. Unlike in any SVC based schemes, the importance and implicit redundancy levels of the source components in ParCast are *aligned*, i.e., the most important component also embeds the highest level of redundancy. Therefore, the power allocation process actually tries to better protect the less important components.

Graceful degradation. Hierarchical modulation is a common digital approach to mitigate glitches and reduce the cliff effects [21, 9]. It can be combined with multiple antennas, as in [16, 31]. These work typically focus on how to correctly decode part of the complex symbol.

Apex [23] recognizes that the bits in a regular QAM symbol have different error probability, and allocates important data to the bits with lower error probability.

Although ParCast is not explicitly optimized for graceful degradation, its linear codec performance degrades with noise in a manner similar to SoftCast.

Joint source-channel coding. Besides SoftCast and FlexCast, there have been a number of designs for lossy single-antenna channels (e.g., [24, 30, 27]), though few for MIMO. ParCast follows the same motivation to jointly consider source compression and error resilience at the channel, but specifically designs a scheme to leverage the properties of both the source and the MIMO-OFDM channel with a single code.

8. CONCLUSION

ParCast is motivated by non-uniform energy distribution at both the source and the channel. It sends independent source components on independent subchannels, with the more important components on higher-gain subchannels, joint source-channel power allocation, and analog modulation to optimally leverage source redundancy for error protection at the channel. We show that such a design can significantly outperform conventional digital schemes by turning challenging MIMO-OFDM links into opportunities.

9. ACKNOWLEDGMENTS

We sincerely thank Chong Luo, Ji-Zheng Xu, and the anonymous reviewers for invaluable feedback.

10. REFERENCES

[1] http://trace.eas.asu.edu/yuv/.
[2] ftp://ftp.tnt.uni-hannover.de/pub/svc/testsequences/.
[3] H.264/AVC JM Reference Software. http://iphome.hhi.de/suehring/tml/.
[4] High-Definition Wireless. AMIMON-WHDI Technology Overview. http://www.amimon.com/technology.shtml.
[5] IEEE Std. 802.11n-2009: Enhancements for Higher Throughput.
[6] SVC Reference Software. http://ip.hhi.de/imagecom_G1/savce/downloads/.
[7] Cisco visual networking index: Forecast and methodology 2010-2015. http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf, June 2011.
[8] S. Aditya and S. Katti. FlexCast: Graceful wireless video streaming. In *ACM MobiCom*, 2011.
[9] B. Barmada, M. Ghandi, E. Jones, and M. Ghanbari. Prioritized transmission of data partitioned H. 264 video with hierarchical QAM. *IEEE Signal Processing Letters*, 12(8):577–580, 2005.
[10] R. K. W. Chan and M. C. Lee. 3D-DCT quantization as a compression technique for video sequences. In *International Conference on Virtual Systems and MultiMedia*, 1997.
[11] D. Halperin, W. Hu, A. Sheth, and D. Wetherall. Predictable 802.11 packet delivery from wireless channel measurements. In *ACM SIGCOMM*, 2010.
[12] G. Hardy, J. Littlewood, and G. Pólya. Inequalities, (2nd ed.), Cambridge Mathematical Library, 1989.
[13] S. Jakubczak and D. Katabi. A cross-layer design for scalable mobile video. In *ACM MobiCom*, 2011.
[14] Z. Ji, Q. Zhang, W. Zhu, Z. Guo, and J. Lu. Power-Efficient MPEG-4 FGS Video Transmission over MIMO-OFDM Systems. In *IEEE ICC*, 2003.
[15] K. Lee and D. Petersen. Optimal linear coding for vector channels. *IEEE Trans. on Communications*, 24(12):1283–1290, 1976.
[16] Y. Noh, H. Lee, and I. Lee. Design of Unequal error protection for MIMO-OFDM systems. In *VTC*, 2005.
[17] H. Rahul, F. Edalat, D. Katabi, and C. G. Sodini. Frequency-aware rate adaptation and MAC protocols. In *ACM MobiCom*, 2009.
[18] H. Rahul, H. Hassanieh, and D. Katabi. SourceSync: A Distributed Wireless Architecture for Exploiting Sender Diversity. In *ACM SIGCOMM*, 2010.
[19] I. Richardson. *H. 264 and MPEG-4 video compression*, volume 20. Wiley Online Library, 2003.
[20] M. F. Sabir, R. Heath, and A. Bovik. An unequal error protection scheme for Multiple Input Multiple Output systems . In *Asilomar*, 2002.
[21] A. Schertz and C. Weck. Hierarchical modulation-the transmission of two independent DVB-T multiplexes on a single frequency. *EBU Technical review*, 2003.
[22] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the H. 264/AVC standard. *IEEE Trans. on Circuits and Systems for Video Technology*, 17(9):1103–1120, 2007.
[23] S. Sen, S. Gilani, S. Srinath, S. Schmitt, and S. Banerjee. Design and Implementation of an "Approximate" Communication System for Wireless Media Applications. In *ACM SIGCOMM*, 2010.
[24] M. Skoglund, N. Phamdo, and F. Alajaji. Hybrid Digital-Analog Source-Channel Coding for Bandwidth Compression/Expansion. *IEEE Trans. on Information Theory*, 52(8):3757–3763, 2006.
[25] D. Song and C. Chen. Maximum-throughput delivery of SVC-based video over MIMO systems with time-varying channel capacity. *Journal of Visual Communication and Image Representation*, 19(8):520–528, 2008.
[26] D. Song and C. W. Chen. Scalable H.264/AVC Video Transmission Over MIMO Wireless Systems With Adaptive Channel Selection Based on Partial Channel Information. *IEEE Trans. on Circuit Systems and Video Techonology*, 17(9), 2007.
[27] M. Stoufs, A. Munteanu, J. Barbarien, J. Cornelis, and P. Schelkens. Optimized scalable multiple-description coding and FEC-based joint source-channel coding: A performance comparison. In *IEEE 10th Workshop on Image Analysis for Multimedia Interactive Services*, 2009.
[28] K. Tan, J. Zhang, J. Fang, H. Liu, Y. Ye, S. Wang, Y. Zhang, H. Wu, W. Wang, and G. M. Voelker. Sora: High Performance Software Radio Using General Purpose Multi-core Processors. In *NSDI*, 2009.
[29] D. Tse and P. Viswanath. *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.
[30] Q. Xu, V. Stankovic, and Z. Xiong. Distributed joint source-channel coding of video using Raptor codes. *IEEE JSAC*, 25(4):851–861, 2007.
[31] G.-H. Yang, D. Shen, and V. Li. Unequal Error Protection for MIMO Systems with a Hybrid Structure. In *ISCAS*, 2006.