A CROSS-LAYER DESIGN FOR SCALABLE MOBILE VIDEO

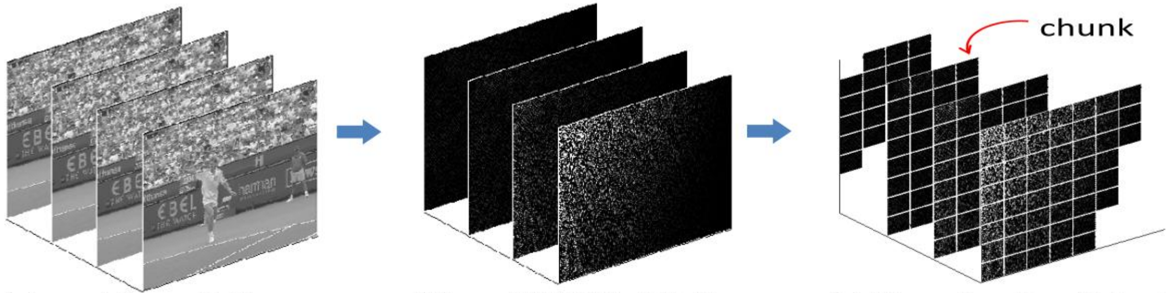
Szymon Jakubczak , Dina Katabi

ABSTRACT

- There are two limitations for today's mobile video
  - Cannot reduce bandwidth consumption by wireless broadcast
  - Lacks robustness to wireless interference and errors
- SoftCast - change the network stack to act like a linear transform
  - Video quality commensurate with its channel quality
  - Increases robustness to interference and errors

SoftCast

- Error-Resilient Compression
  - SoftCast compresses the video using a weighted 3-dimensional DCT transform
  - Transmit all the non-zero chunks
  - Sort the chunks in decreasing order of their energy and picks chunks as possible to fill the bandwidth



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

SoftCast(2)

- Error Protection
  - Scaling the magnitude of the DCT components in a frame
  - Let $x_i[j], j = 1 \dots N$, be a random variables drawn from a distribution $\mathcal{D}_i$ and remove its mean with zero mean, and variance λ_i
  - The mean of $\mathcal{D}_i$ will send as metadata

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

SoftCast(3)

- Resilience to Packet Loss
  - Each SoftCast slice is a linear combination of all chunks
  - SoftCast produces these slices by multiplying the chunks with the Hadamard matrix
  - Hadamard matrix is an orthogonal transform composed entirely of +1s and -1s

Encoder

- The encoding process can then be represented as

$$Y = HGX = CX$$

- G is a diagonal matrix with the scaling factors, H is the Hadamard matrix

Decoder

- Use Linear Least Square Estimator (LLSE) to estimate DCT components

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

- At high SNR(small noise, the entries in Σ approach 0 $X_{LLSE} \approx C^{-1}Y$

- The loss of a packet corresponds to the absence of a row in Y

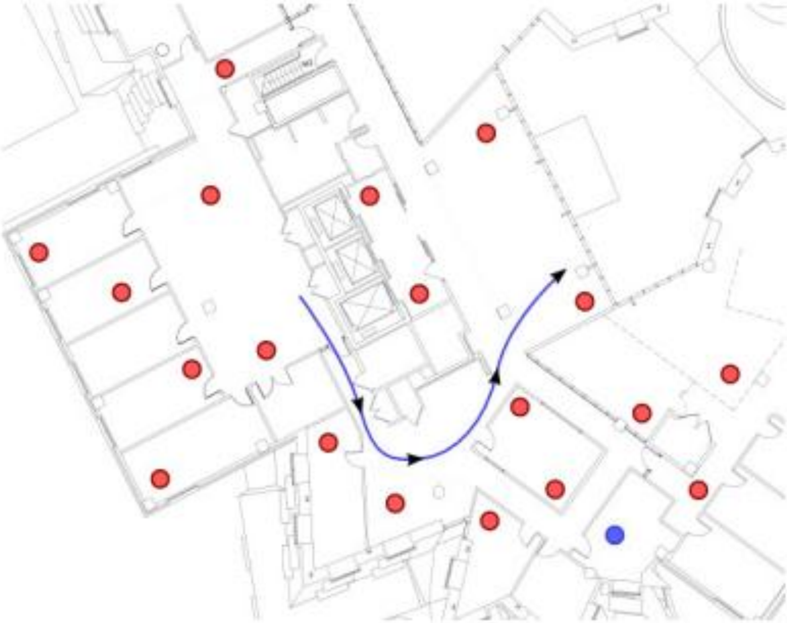
$$X_{LLSE} = \Lambda_x C_{*i}^T (C_{*i} \Lambda_x C_{*i}^T + \Sigma_{(*i,*i)})^{-1} \hat{Y}_{*i}.$$

Implementation

- Use the GNURadio codebase to build a prototype of SoftCast
- Physical Layer
  - Implementation leverages the OFDM implementation in the GNURadio
  - The transmitter's PHY passes SoftCast's packets directly to OFDM

Implementation (2)

- Video Coding
  - Implemented SoftCast in Python (with SciPy)

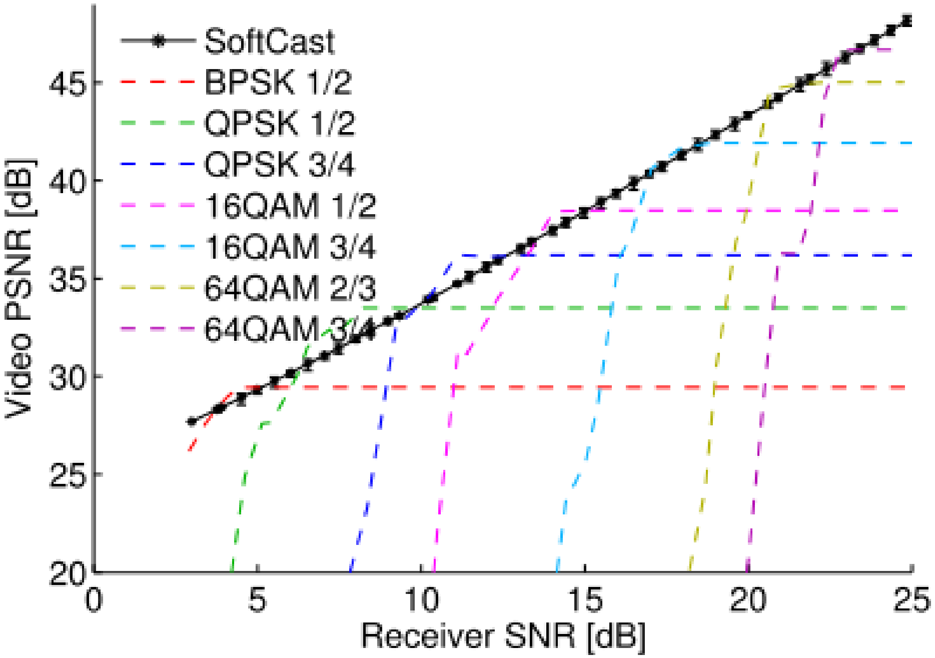


Evaluation environment

- Testbed: in the 20-node GNURadio testbed
- Modulation and Coding: SoftCast is transmitted directly over OFDM
- Wireless Environment: The carrier frequency is the same as that of 802.11b/g
- Metric: compare the schemes using the Peak Signal-to-Noise Ratio (PSNR)

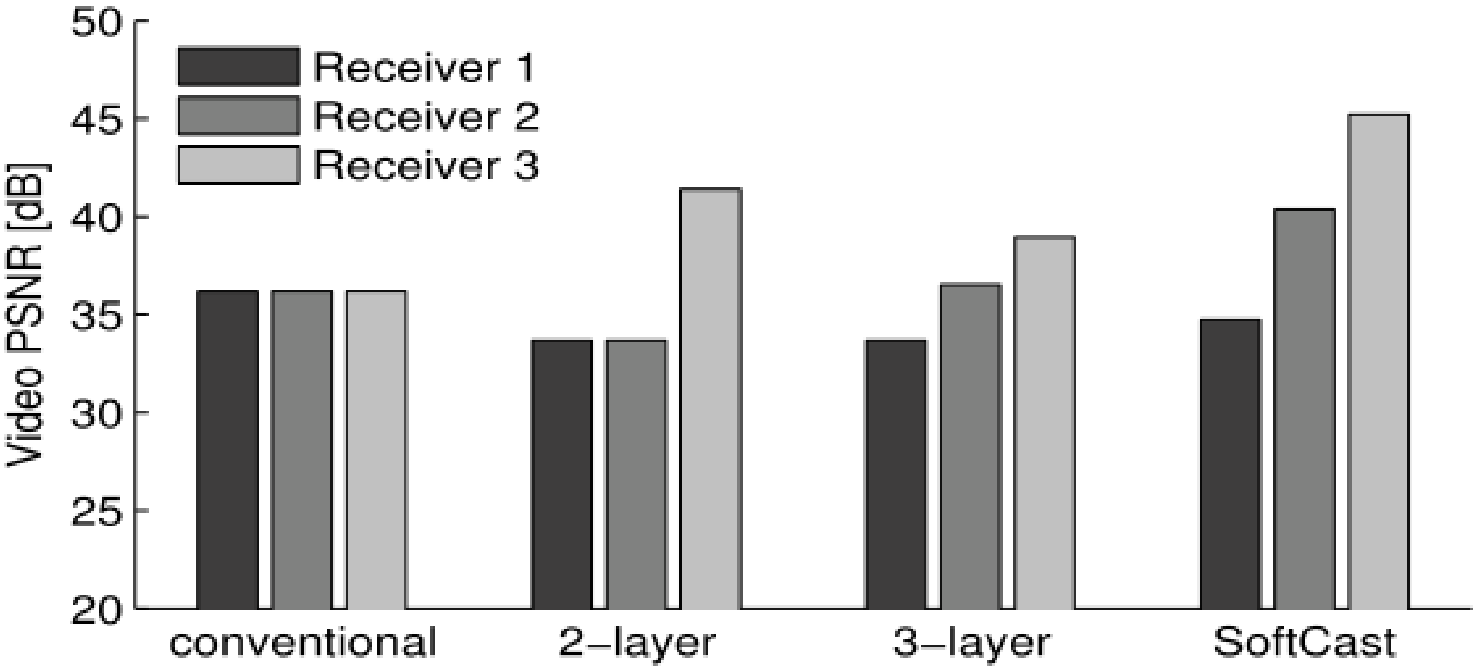
Evaluation

- Performance of SoftCast (in black) vs. single-layer MPEG4

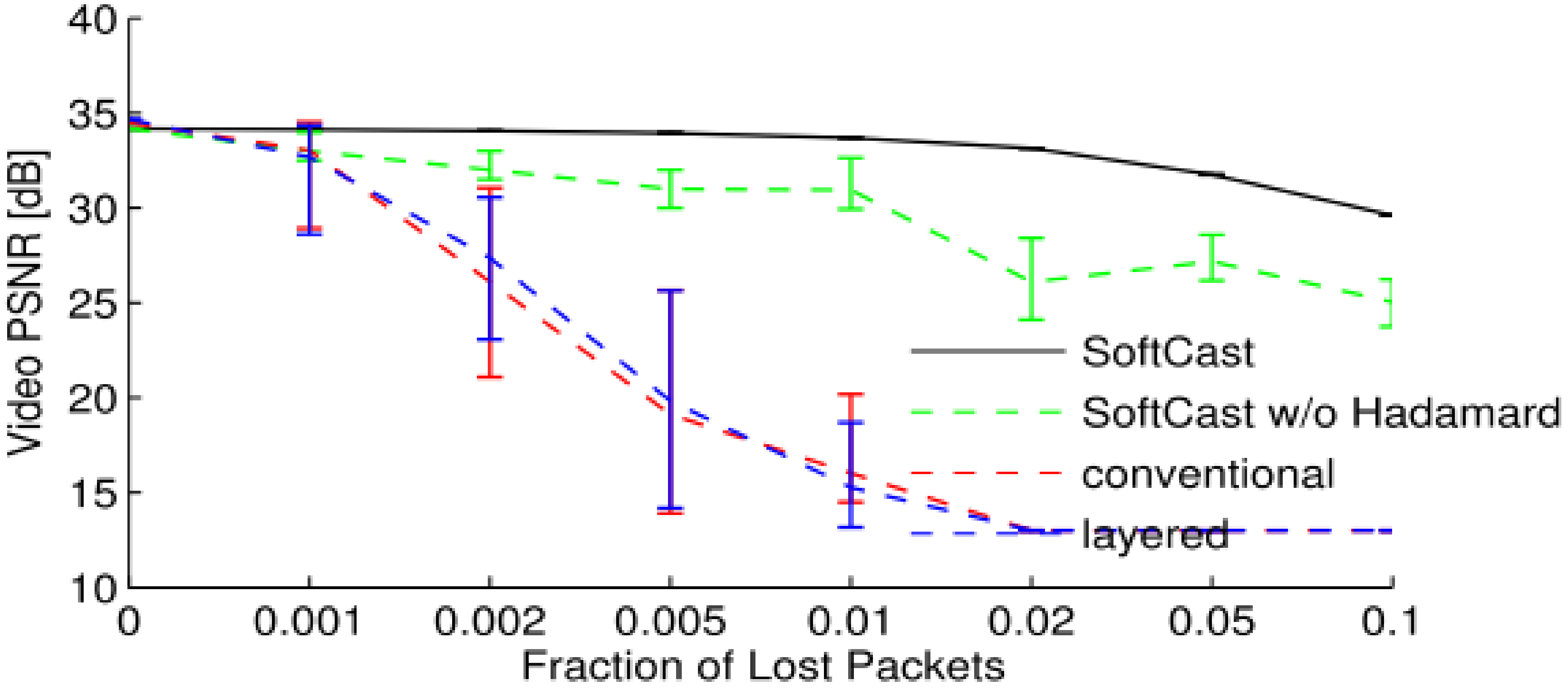


Evaluation of multicast

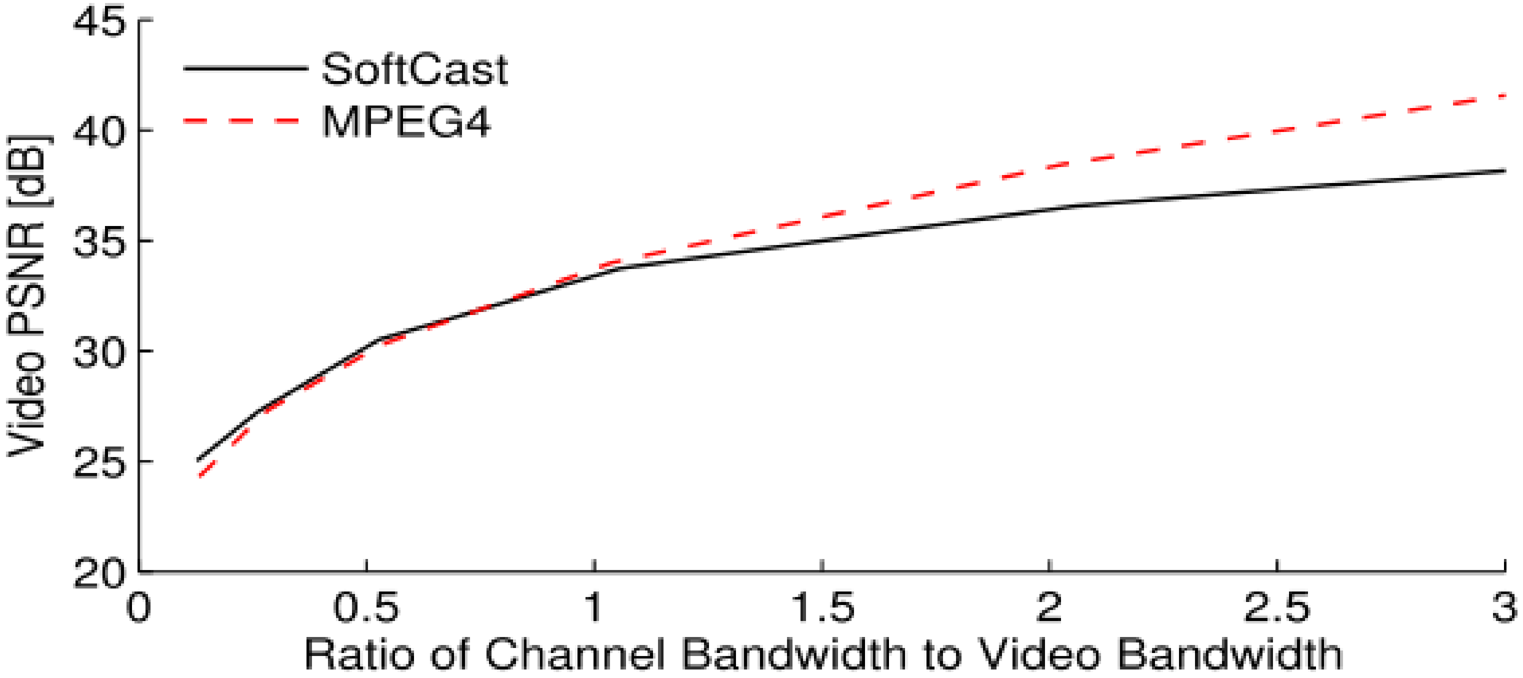
- The receivers’ SNRs are 11 dB, 17 dB, and 22 dB.



Evaluation of robustness to Packet Loss



Impact of available wireless bandwidth



Conclusion

- SoftCast adopts an integrated design for video and PHY layer coding
- Making the whole network stack act as a linear transform
- Improves video quality for multicast users, eliminates video glitches caused by mobility, and increases robustness to interference and channel errors.