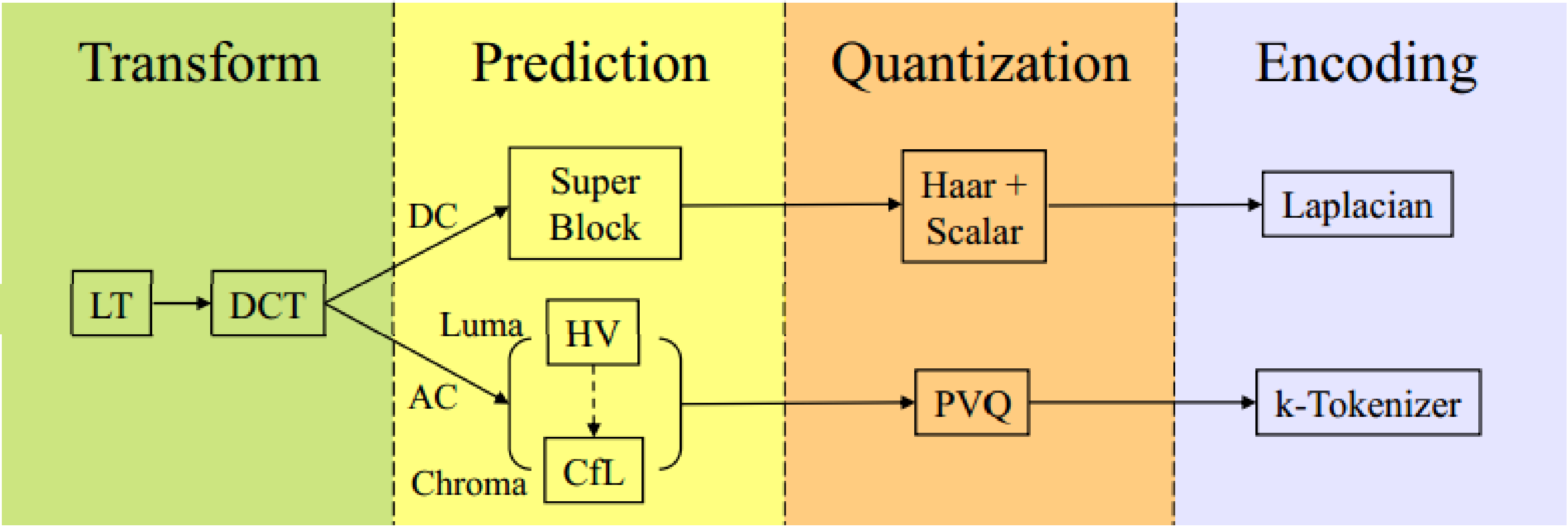
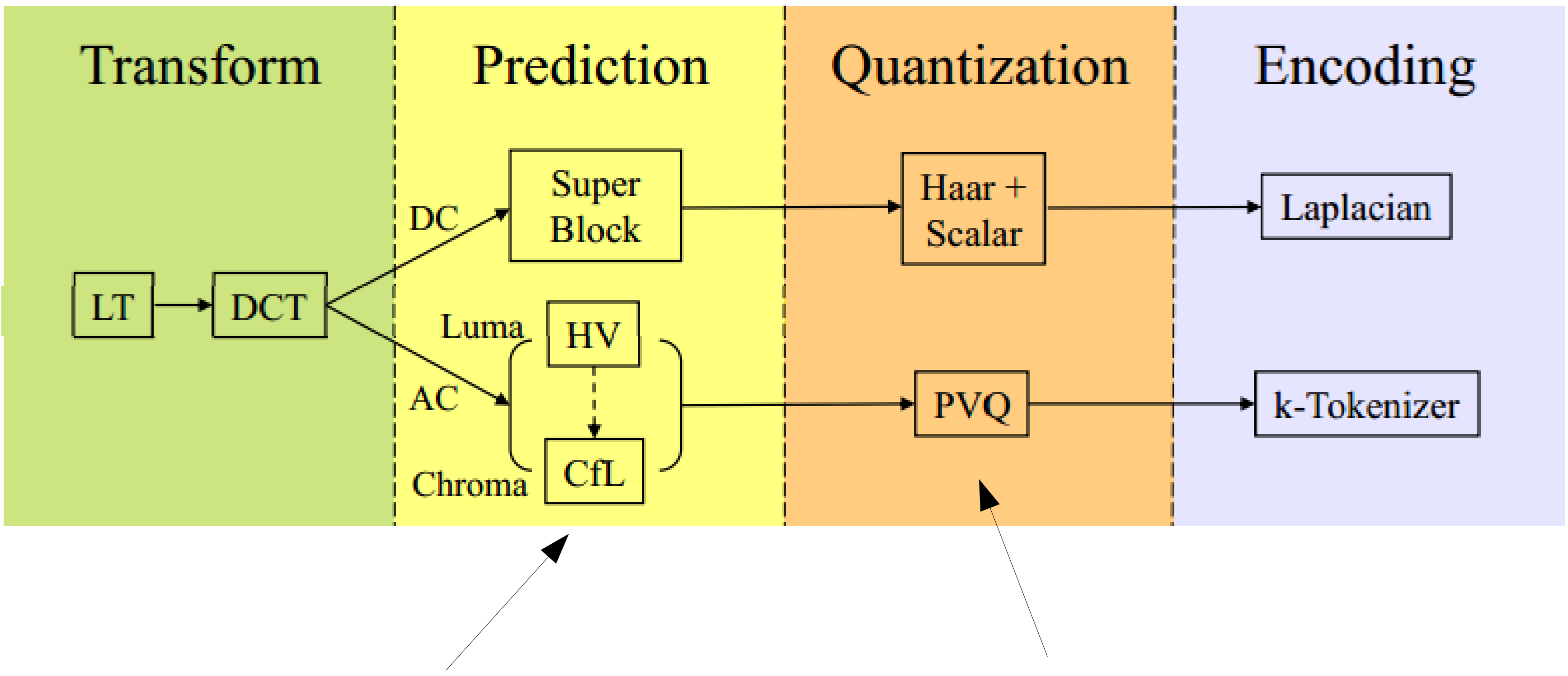
Daala

- Daala is a high-efficiency video codec designed for internet applications
- Technical differences (so far)
  - Lapped Transforms
  - Perceptual Vector Quantization
  - Chroma from Luma Prediction
  - Overlapped Block Motion Compensation
  - Paint Deringing Filter
  - Multisymbol arithmetic coding

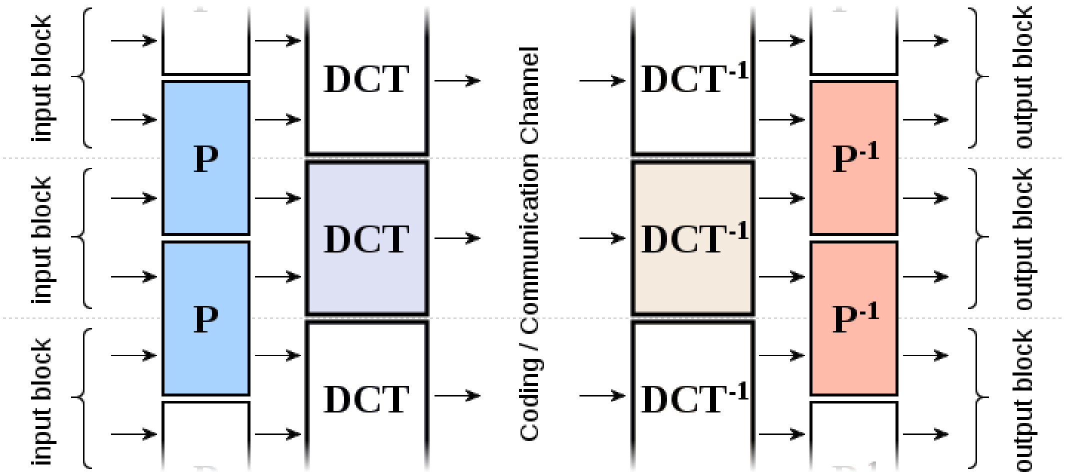
Still Image Encoding



Still Image Encoding



Lapped Transforms



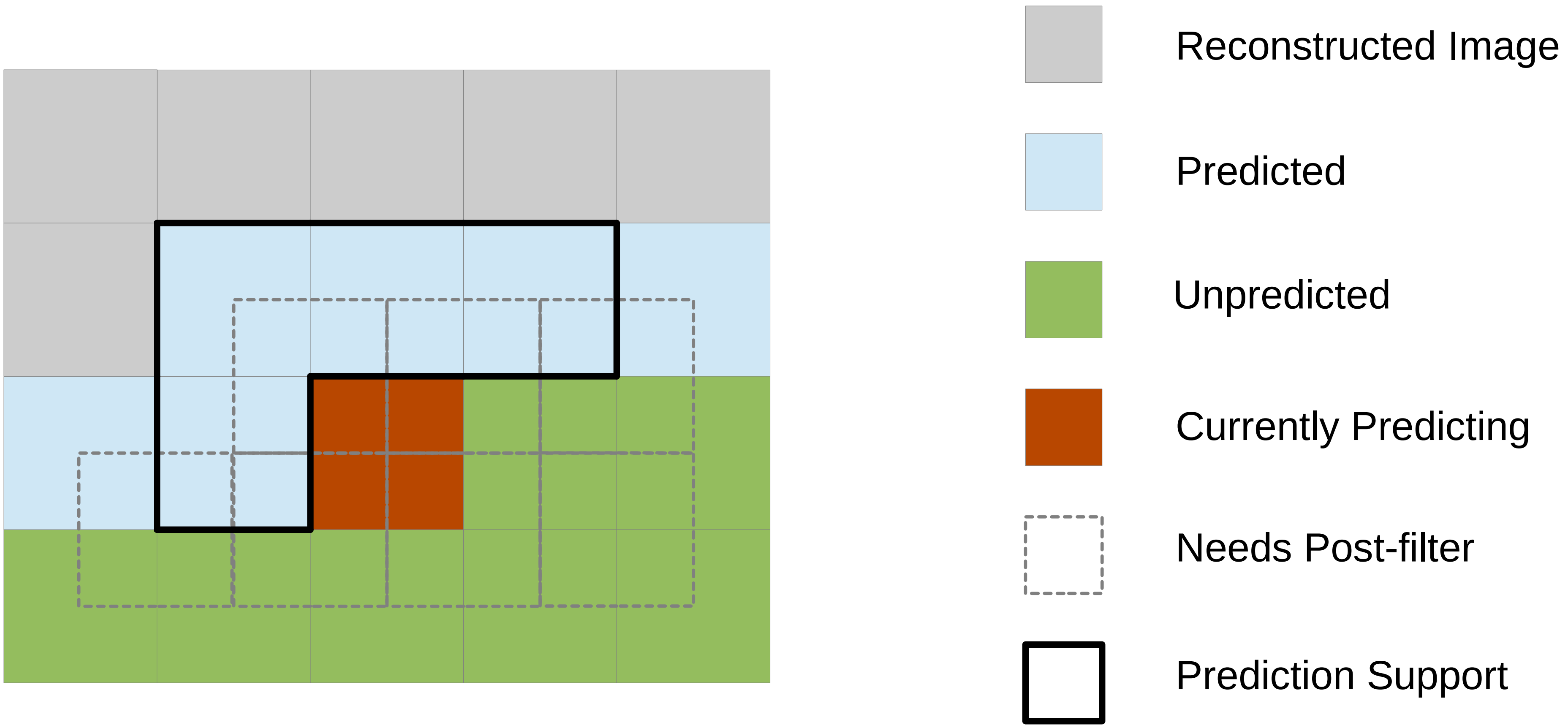
Lapped Transforms

- No more blocking artifacts, without loop filter
- Computationally cheaper than wavelets
- Better compression than DCT / wavelets
- Doesn't completely disrupt block based infrast.

subset-1	4x4	8x8	16x16
KLT	12.47 dB	13.62 dB	14.12 dB
DCT	12.42 dB	13.55 dB	14.05 dB
CDF 9/7	13.14 dB	13.82 dB	14.01 dB
LT-KLT	13.35 dB	14.13 dB	14.40 dB
LT-DCT	13.33 dB	14.12 dB	14.40 dB

Decoding an Intra Frame with Lapped Transforms

Neighboring blocks:



Perceptual Vector Quantization

- Separate “gain” (contrast) from “shape” (spectrum)
  - Vector = Magnitude × Unit Vector (point on sphere)
- Potential advantages
  - Better contrast preservation
  - Better representation of coefficients
  - Free “activity masking”
    - Can throw away more information in regions of high contrast (*relative* error is smaller)
    - The “gain” is what we need to know to do this!

Simple Case: PVQ without a Predictor

- Scalar quantize gain
- Place *K* unit pulses in *N* dimensions
  - Only has (*N* - 1) degrees of freedom

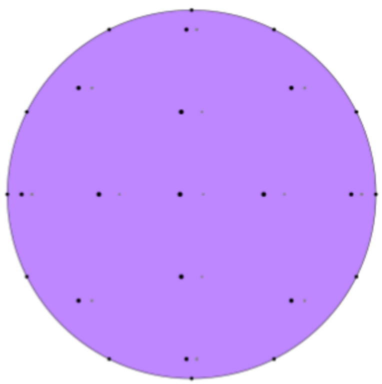
$$\mathbf{y} \in \mathbb{Z}^N : \sum_{i=0}^{N-1} |y_i| = K$$

- Normalize to unit L_2 norm

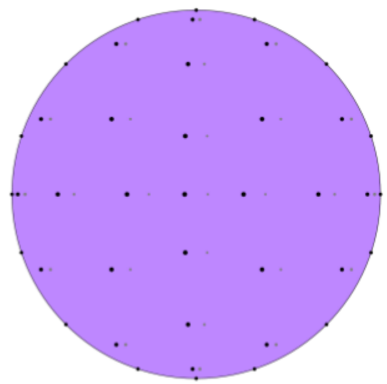
$$\mathbf{u} = \mathbf{y} / \|\mathbf{y}\|_{L2}$$

- *K* is derived implicitly from the gain

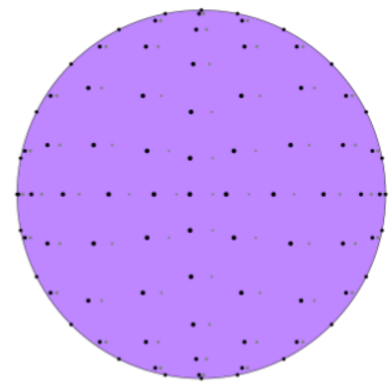
Codebook for *N*=3 and different *K*



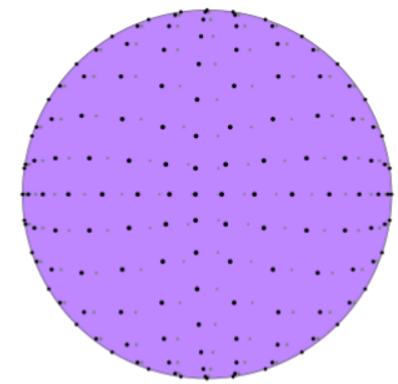
5.25 bits (K=3)



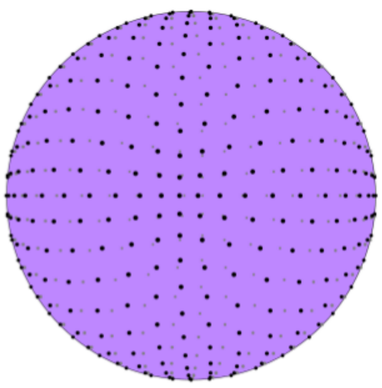
6.04 bits (K=4)



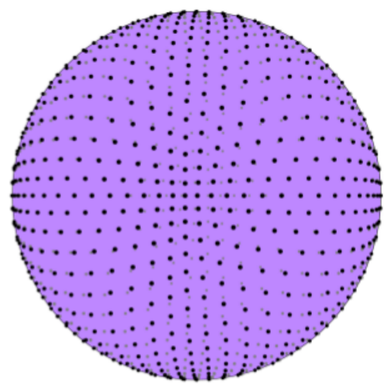
7.19 bits (K=6)



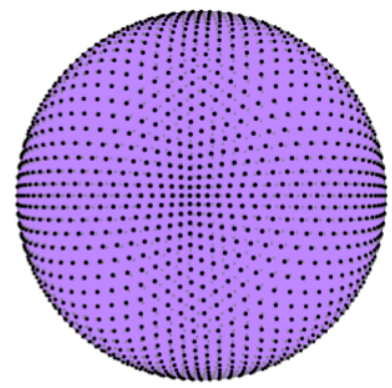
8.01 bits (K=8)



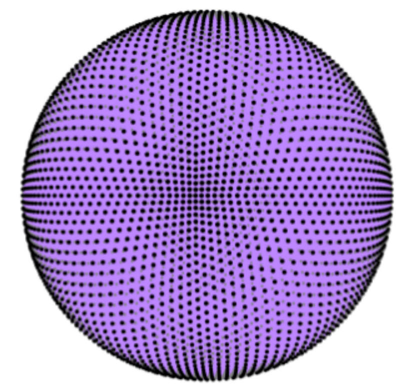
8.92 bits (K=11)



10.00 bits (K=16)

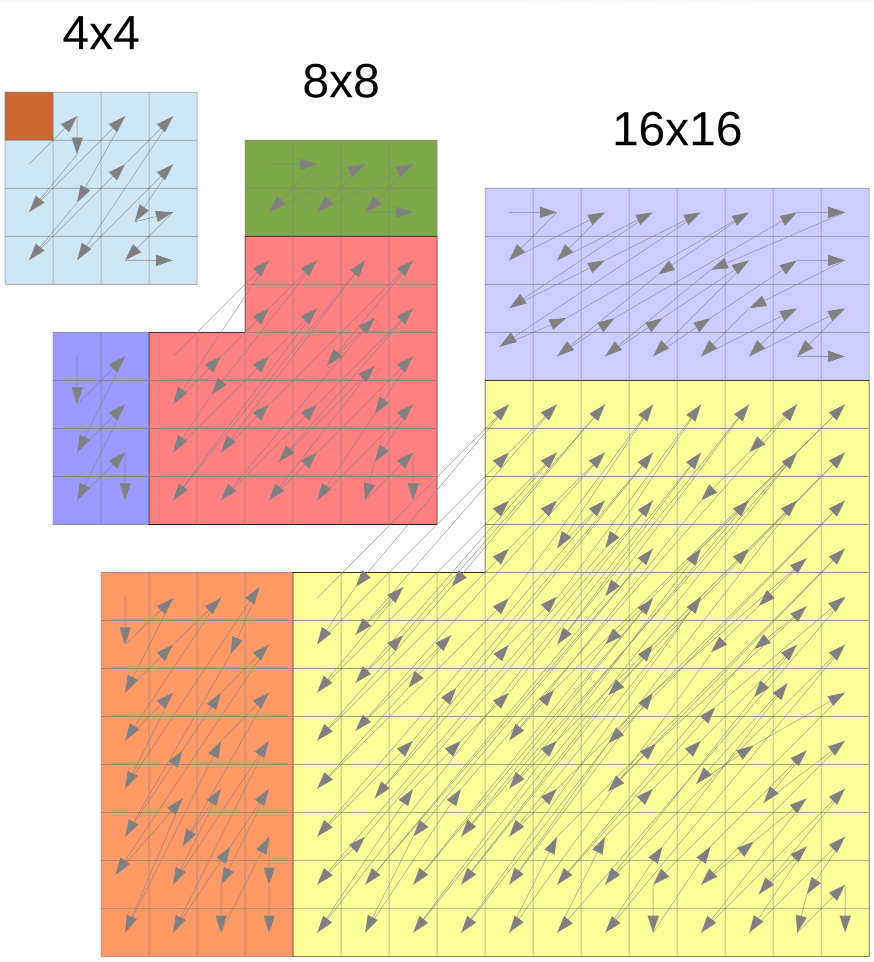


11.05 bits (K=23)

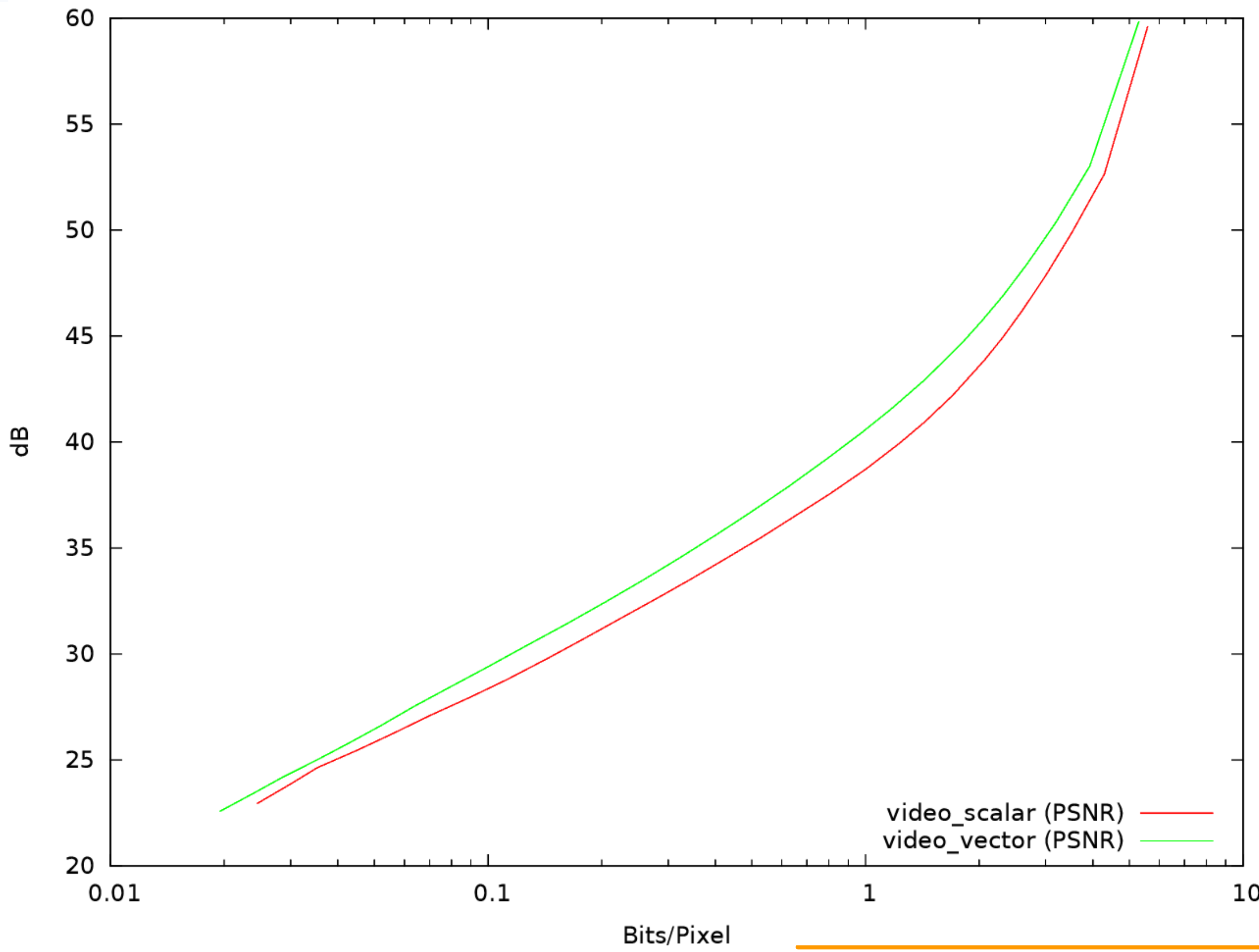


12.00 bits (K=32)

Band Structure



Results (PVQ vs Scalar)



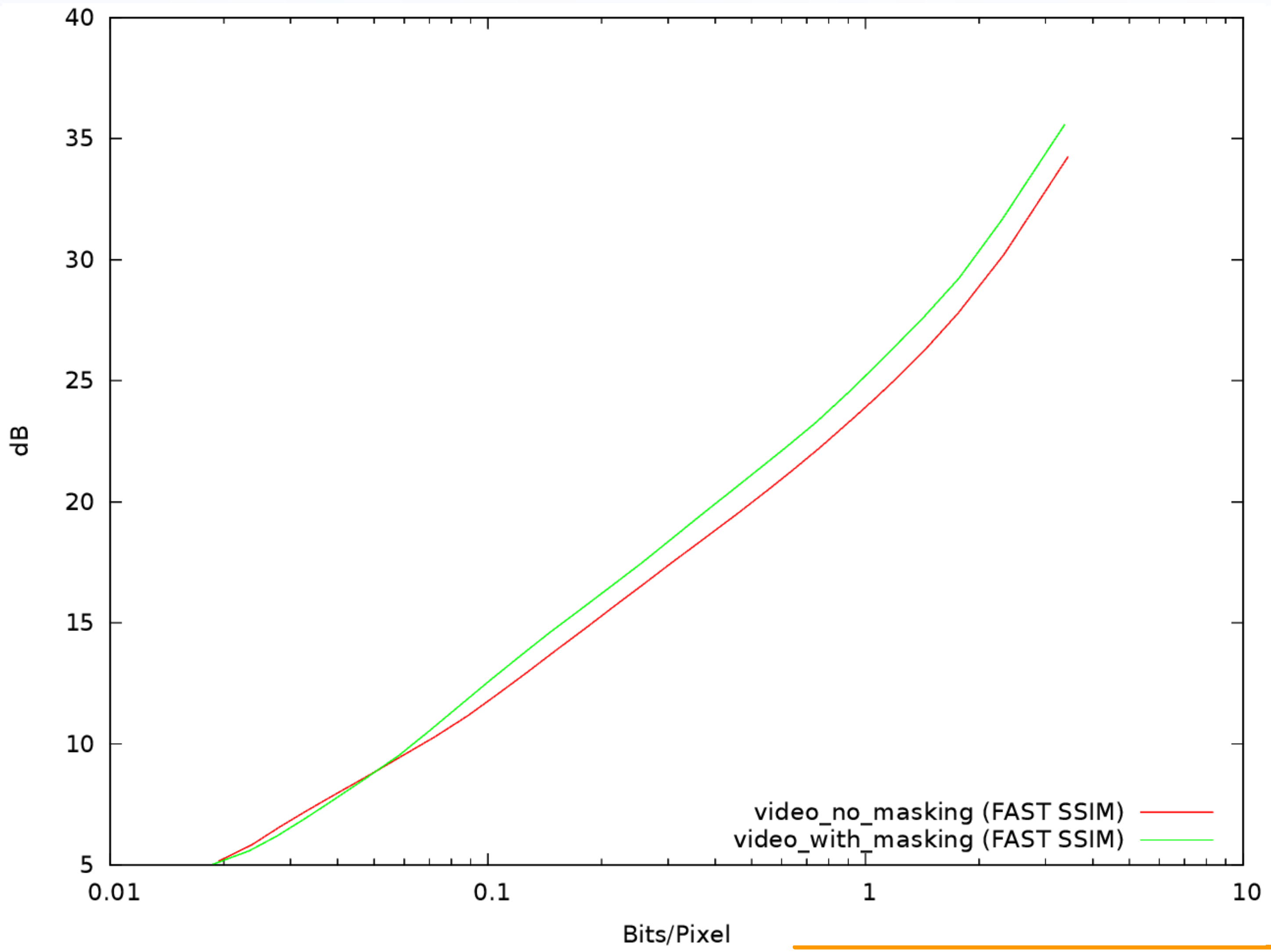
Activity Masking

- Goal: Use better resolution in flat areas
  - Most codecs require explicit QP signaling (MB)
  - PVQ allows implicit signaling based on gain (band)
- Changes how *K* is computed from the gain
- Gain quantized using a non-linear scale

No Activity Masking (54 kB)

Activity Masking (54 kB)

Results (Activity Masking)

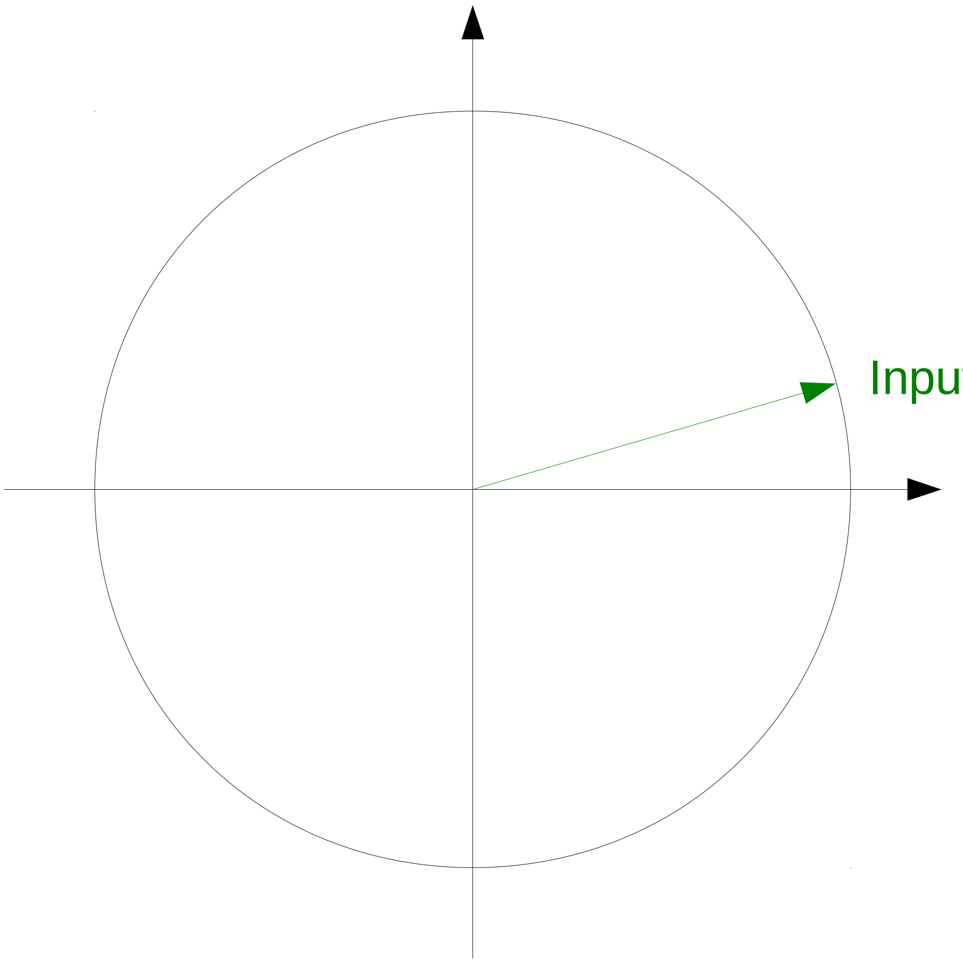


Using Prediction

- Subtracting and coding a residual loses energy preservation
  - The “gain” no longer represents the contrast
- But we still want to use predictors
  - They do a *really* good job of reducing what we need to code
  - Hard to use prediction on the shape (on the surface of a hyper-sphere)
- Solution: transform the space to make it easier

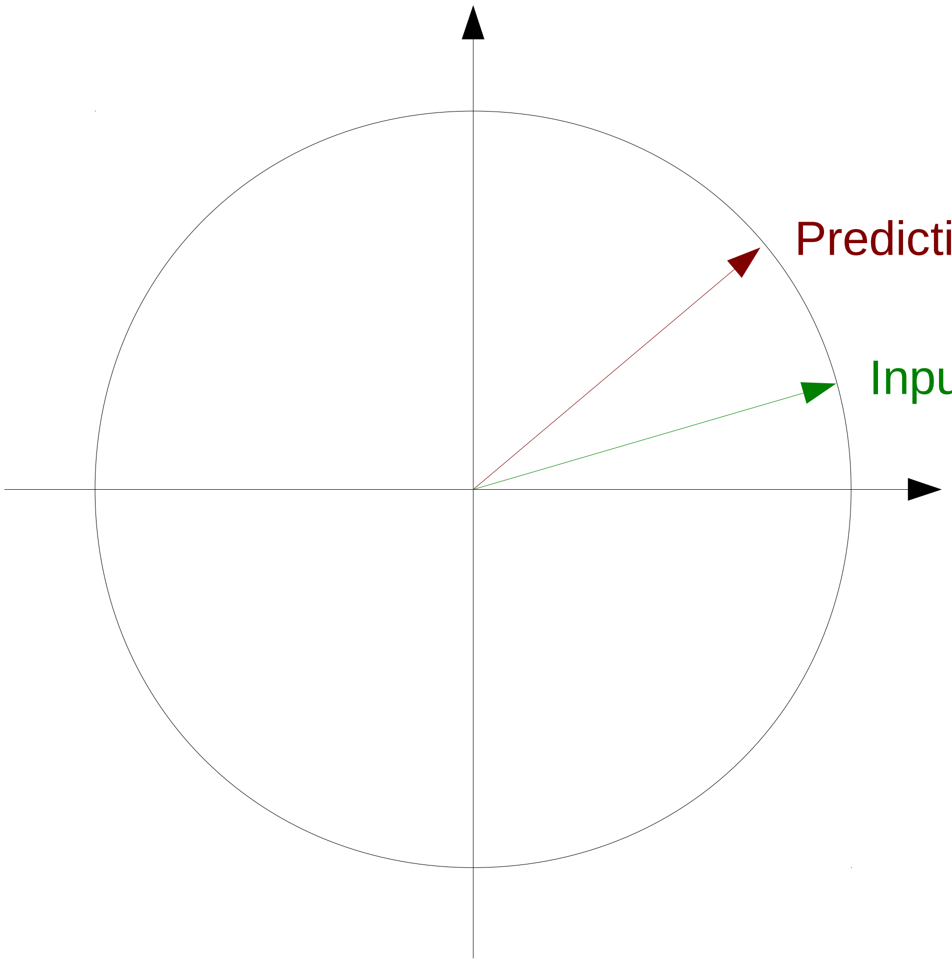
2-D Projection Example

- Input



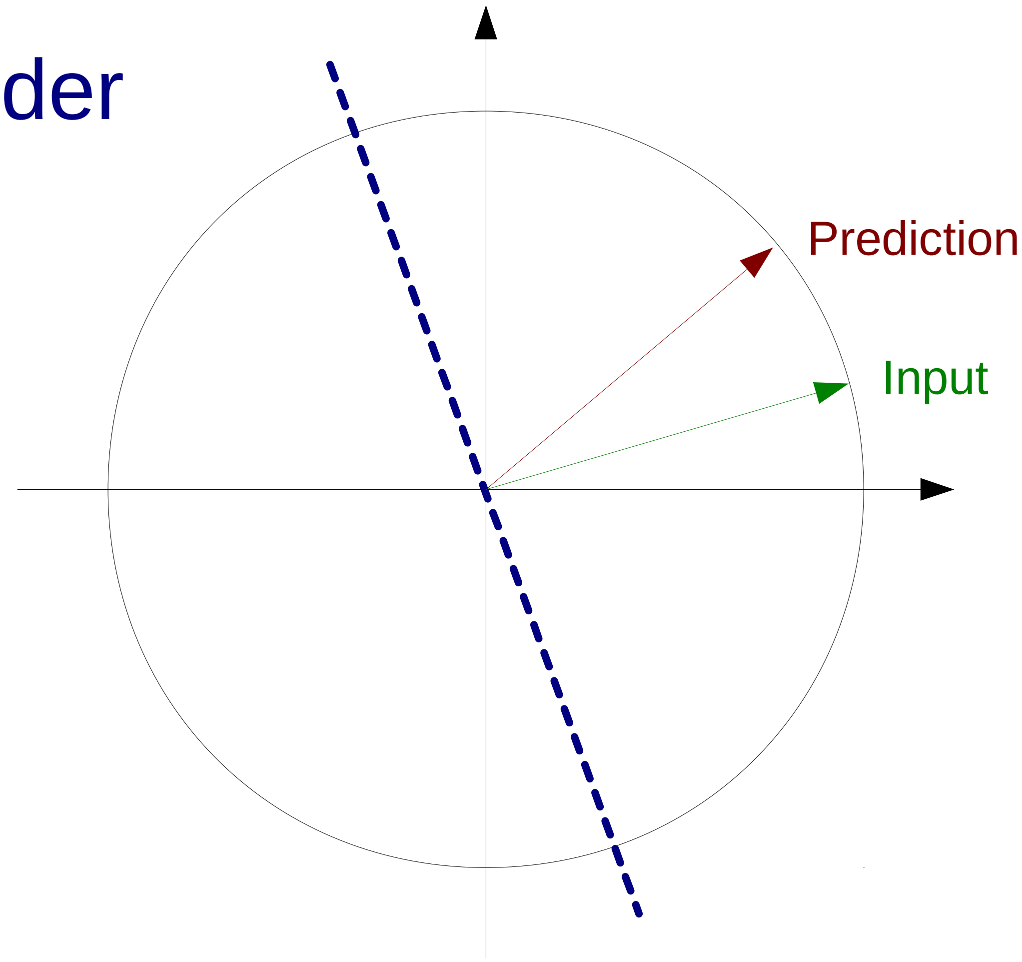
2-D Projection Example

- Input + Prediction



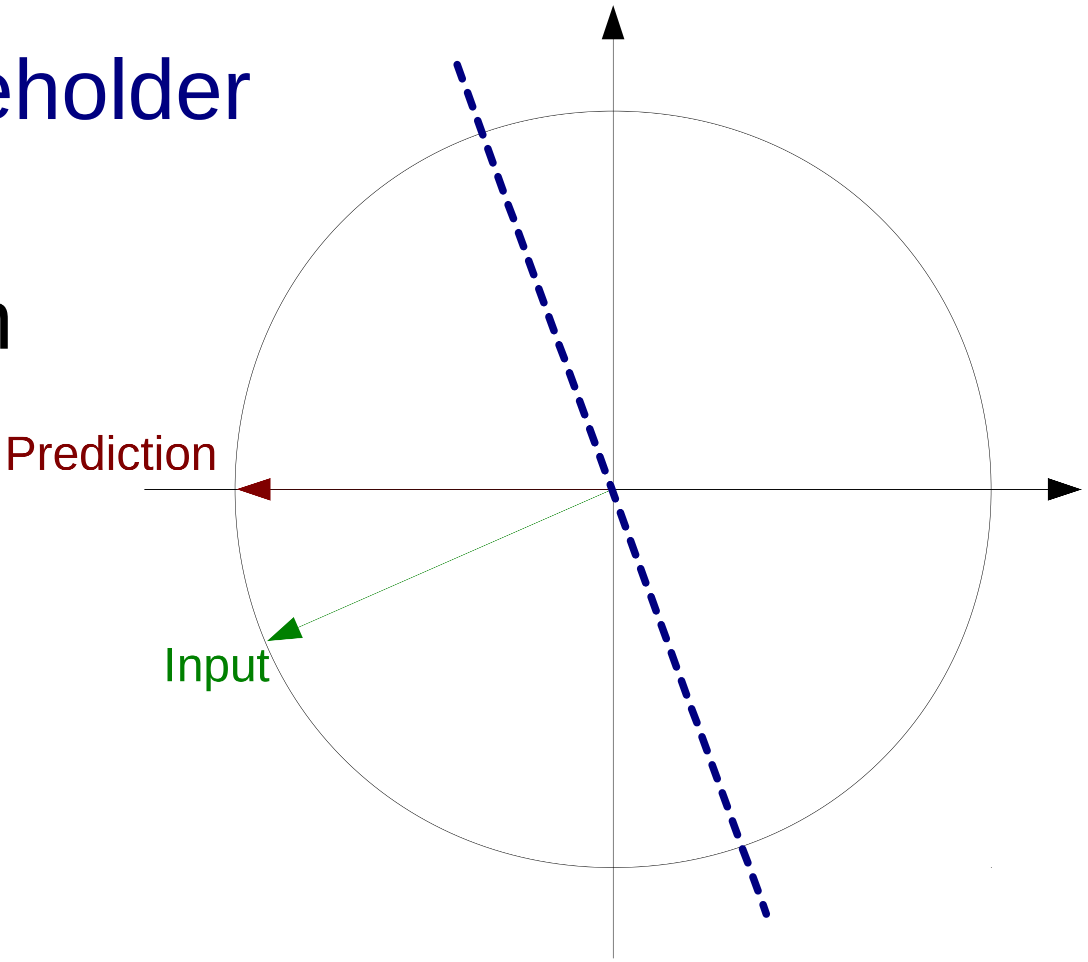
2-D Projection Example

- Input + Prediction
- Compute Householder Reflection



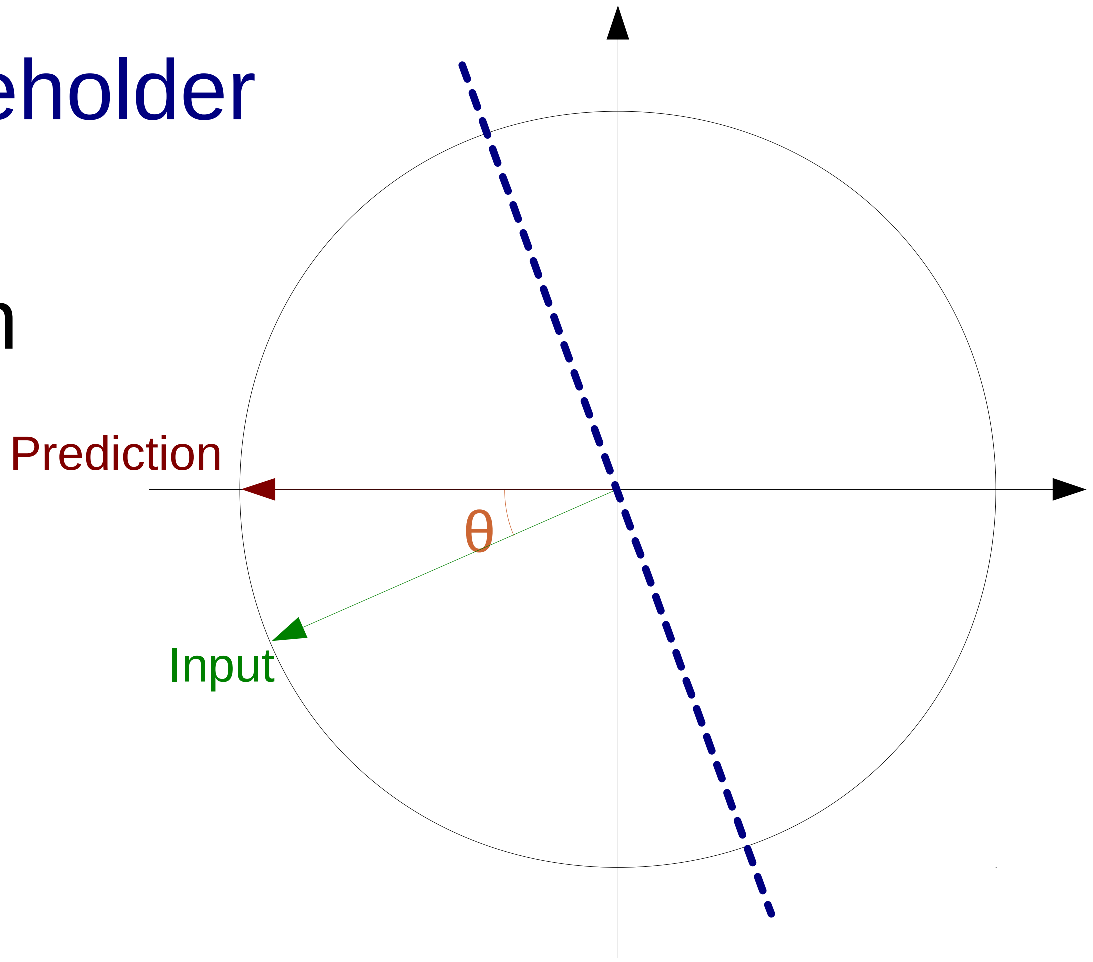
2-D Projection Example

- Input + Prediction
- Compute Householder Reflection
- Apply Reflection



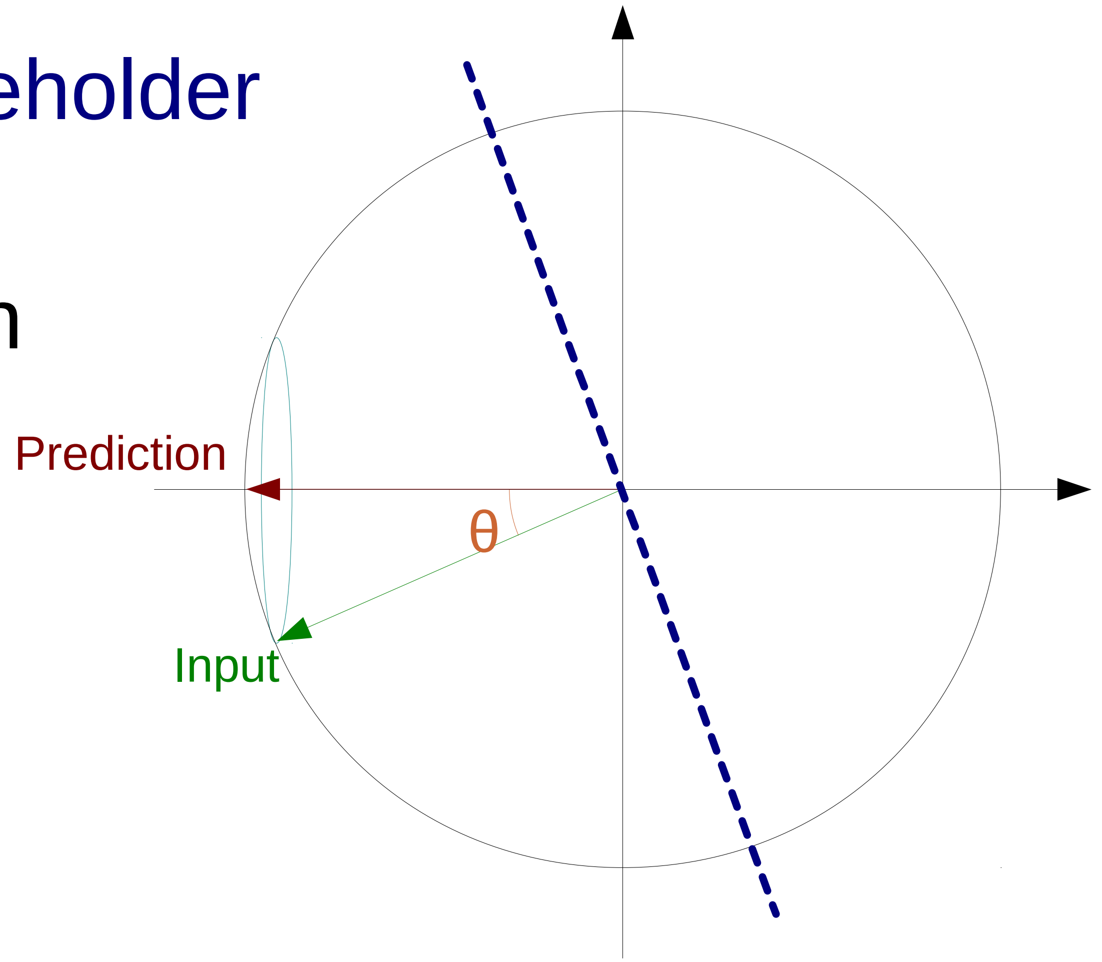
2-D Projection Example

- Input + Prediction
- Compute Householder Reflection
- Apply Reflection
- Compute & code angle



2-D Projection Example

- Input + Prediction
- Compute Householder Reflection
- Apply Reflection
- Compute & code angle
- Code other dimensions



What does this accomplish?

- Creates another "intuitive" parameter, θ
  - "How much like the predictor are we?"
  - $\theta = 0 \rightarrow$ use predictor exactly
- Remaining N-1 dimensions are coded with VQ
  - We know their magnitude is gain*sin(θ)
- Instead of subtraction (translation), we're scaling and reflecting
  - This is *nothing* like computing a DFD

To Predict or Not to Predict...

- $\theta \geq \pi/2 \rightarrow$ Prediction not helping
  - Could code large θ*'s*, but doesn't seem that useful
  - Need to handle zero predictors anyway
- Current approach: code a “noref” flag
  - Jointly coded with gain and θ

Spatial Prediction of Chroma

- In 4:2:0 image data, chroma is 50% of luma
- Chroma predicted spatially by signalling a directional mode
  - Reconstructed neighbors must be available to decode a block
  - Limited to predicting from current color plane
- Cross-channel correlation not exploited
- Does not work with codecs using lapped transforms!

Predicting Chroma from Luma

- Key insight: YUV conversion de-correlates luma and chroma globally, but local relationship exists [1]
- Both encoder and decoder compute linear regression:

$$\alpha = \frac{N \cdot \sum_i L_i \cdot C_i - \sum_i L_i \sum_i C_i}{N \cdot \sum_i L_i \cdot L_i - \left(\sum_i C_i\right)^2} \qquad \beta = \frac{\sum_i C_i - \alpha \cdot \sum_i L_i}{N}$$

- Use reconstructed luma coefficients to predict coincident chroma coefficients:

$$C(u,v) = \alpha \cdot L(u,v) + \beta$$

- Not selected for HEVC due to 20-30% increased complexity

[1] S.H. Lee & N.I. Cho: "Intra prediction method based on the linear relationship between the channels for YUV 4:2:0 intra coding" ICIP 2009, pp. 1033-1036

Adapting Chroma from Luma to the Frequency Domain

- Key insight: LT and DCT are both linear transforms so similar relationship exists in frequency domain
- Both encoder and decoder compute linear regression using 4 LF coefficients from Up, Left and Up-Left
- Use reconstructed luma coefficients to predict coincident chroma coefficients:

$$C_{DC} = \alpha_{DC} \cdot L_{DC} + \beta_{DC}$$

$$C_{AC}(u,v) = \alpha_{AC} \cdot L_{AC}(u,v)$$

- Still expensive, but cost constant with block size

Block Size	SD-CfL		FD-CfL	
	Adds	Mults	Adds	Mults
N x N	4*N+2	8*N+3	2*12+5	4*12+5
4 x 4	18	35	29	53
8 x 8	34	67	29	53
16 x 16	66	131	29	53

Example

Original uncompressed image

Example

Reconstructed luma with predicted chroma using FD-CfL

PVQ Prediction with CfL

- Consider prediction of 15 AC coefficients of 4x4 Cb
- The 15-dimensional predictor $\mathbf{r}$ is scalar multiple of coincident reconstructed luma coefficients $\hat{\mathbf{x}}_L$

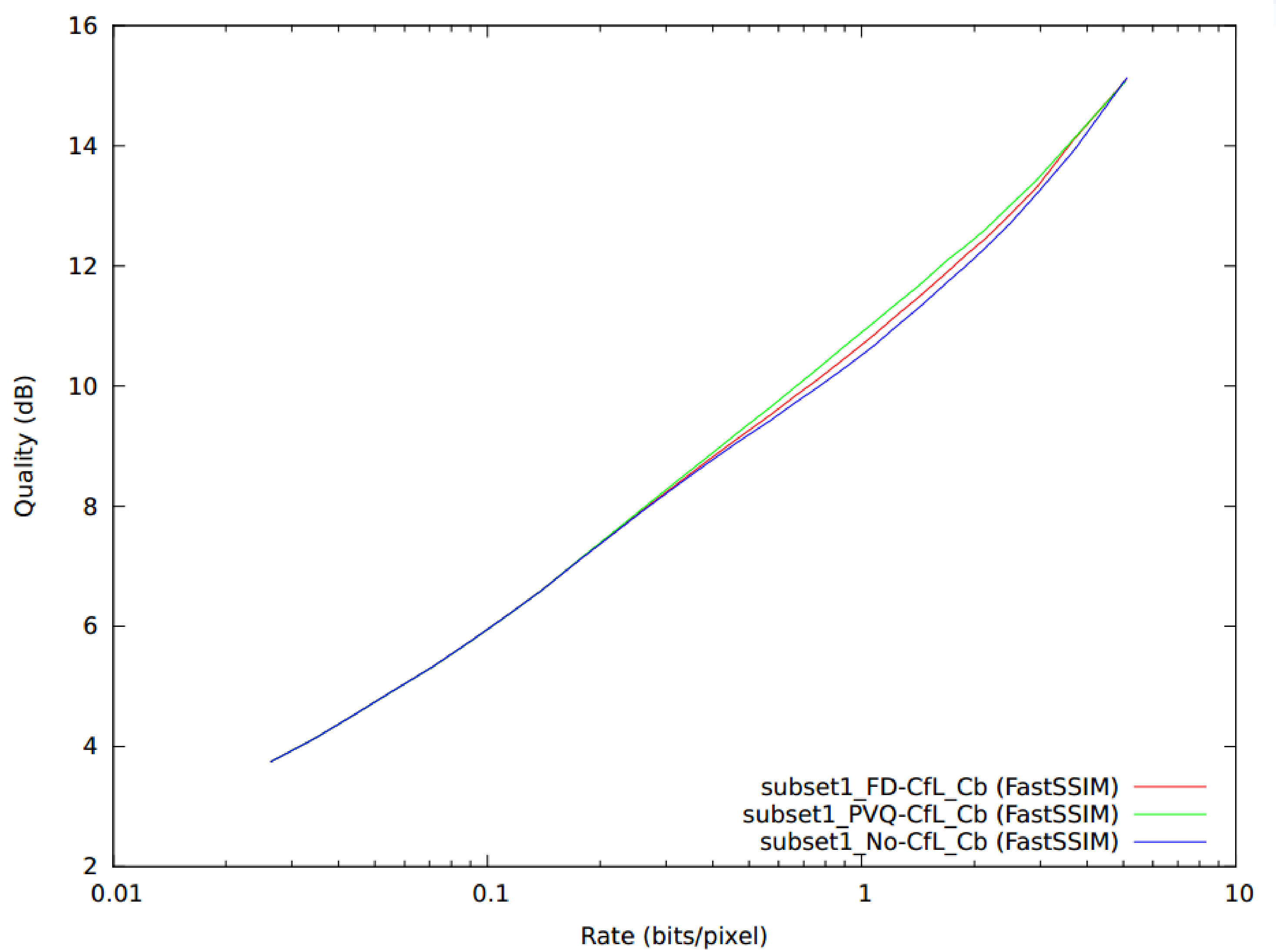
$$C_{AC}(u,v) = \alpha_{AC} \cdot L_{AC}(u,v) \implies \mathbf{r} = \alpha_{AC} \cdot \hat{\mathbf{x}}_L$$

- Thus “shape” predictor is almost exactly $\hat{\mathbf{x}}_L$

$$\frac{\mathbf{r}}{\|\mathbf{r}\|} = \frac{\alpha_{AC} \cdot \hat{\mathbf{x}}_L}{\|\alpha_{AC} \cdot \hat{\mathbf{x}}_L\|} = \operatorname{sgn}(\alpha_{AC}) \frac{\hat{\mathbf{x}}_L}{\|\hat{\mathbf{x}}_L\|}$$

- Only difference is *direction* of correlation!

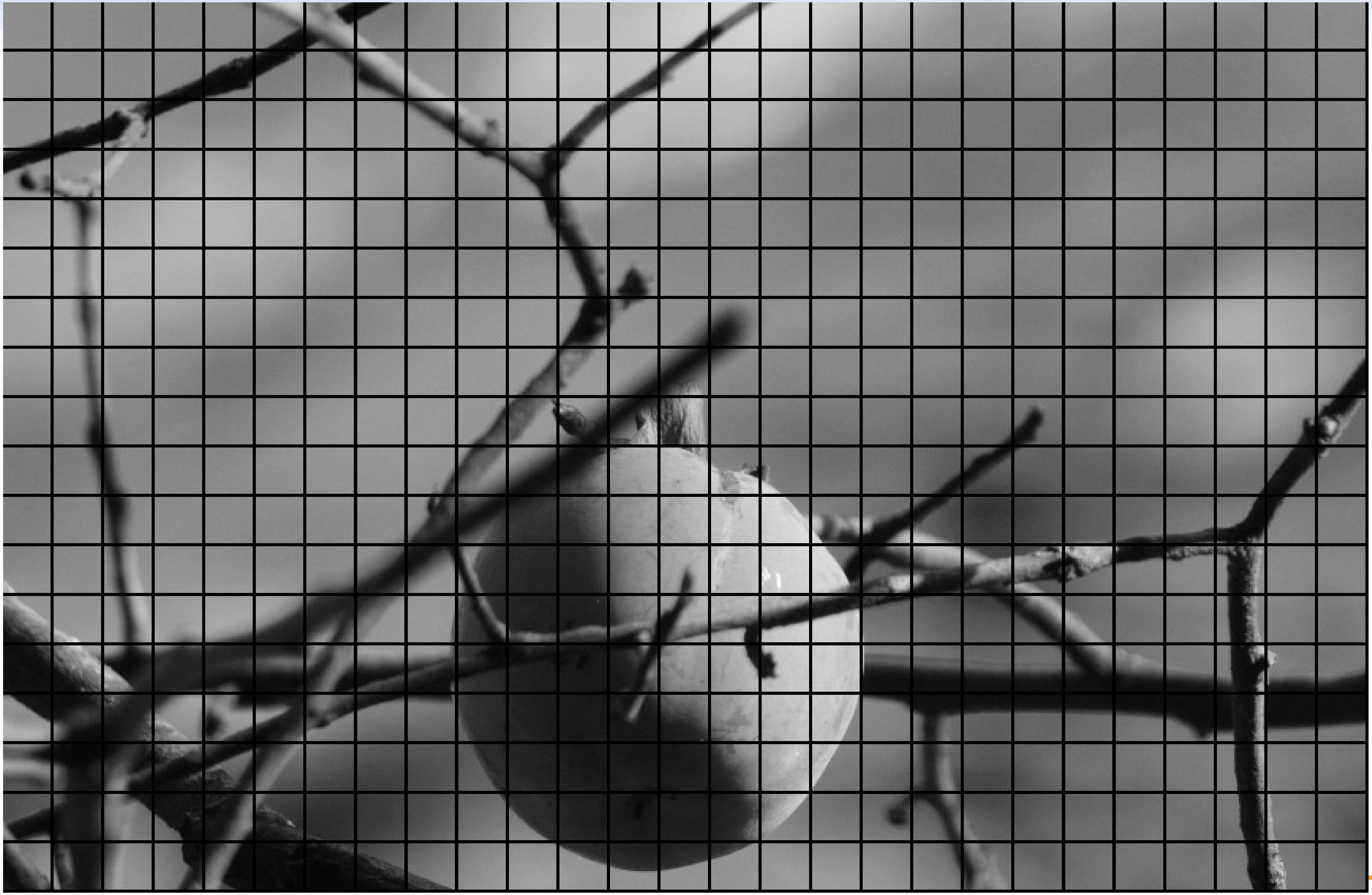
Results (FD-CfL v PVQ-CfL)



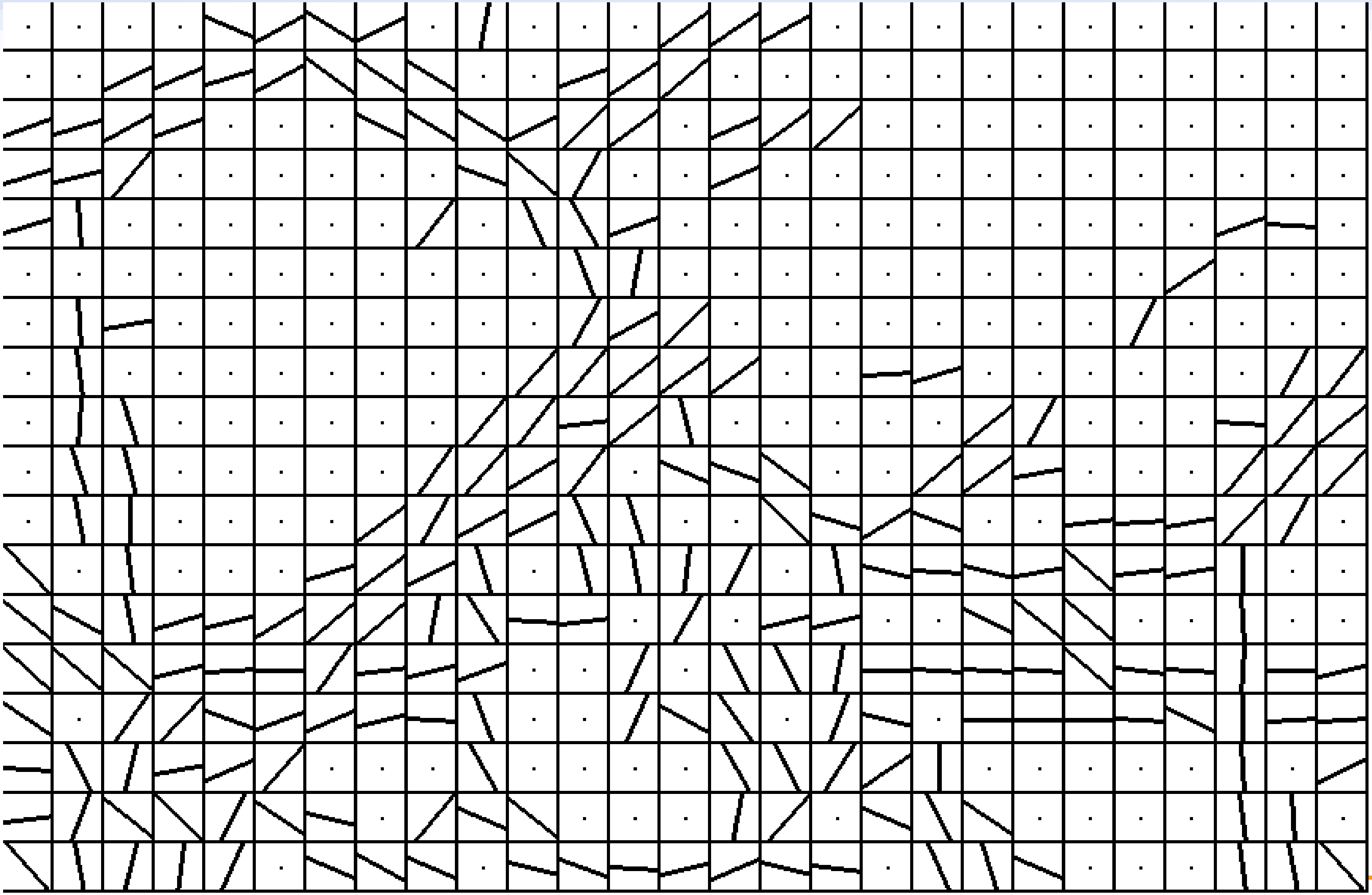
Paint De-Ringing Filter

- Larger support of lapped transforms increases ringing
- Proposed paint deringing filter directionally blends proportional to quantization noise
  1) Direction search (on reconstruction)
  2) Boundary pixel optimization
  3) Paint and blend

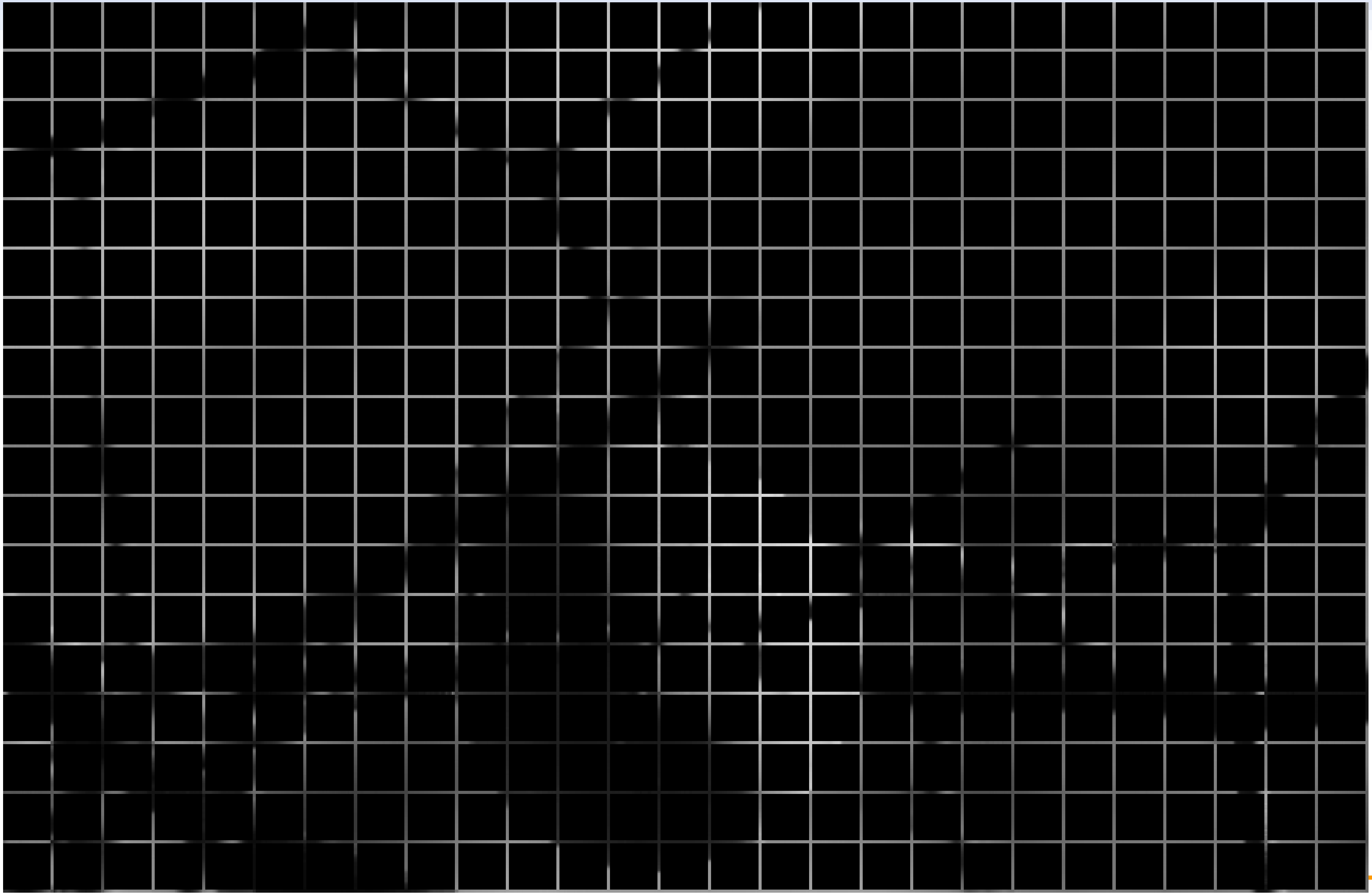
Paint (Block Partition)



Paint (Direction Search)

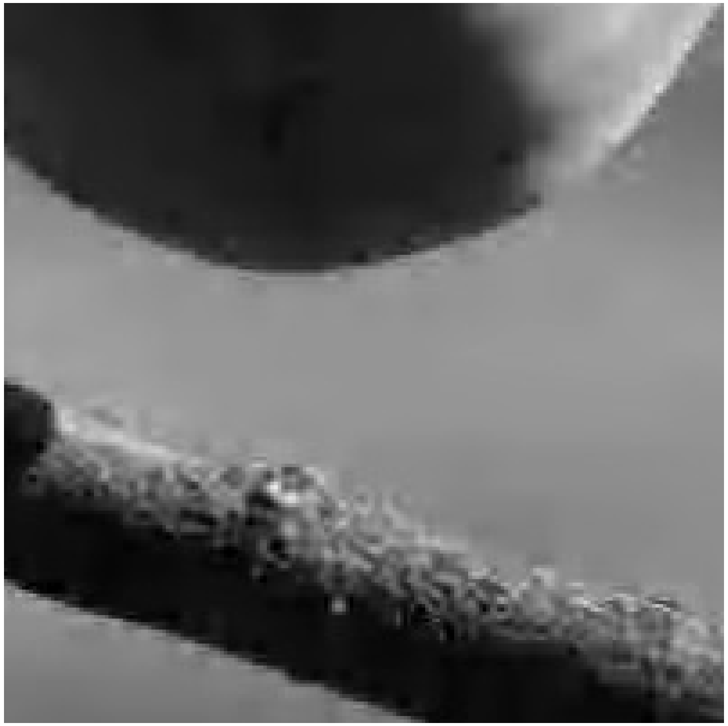


Paint (Boundary Optimization)

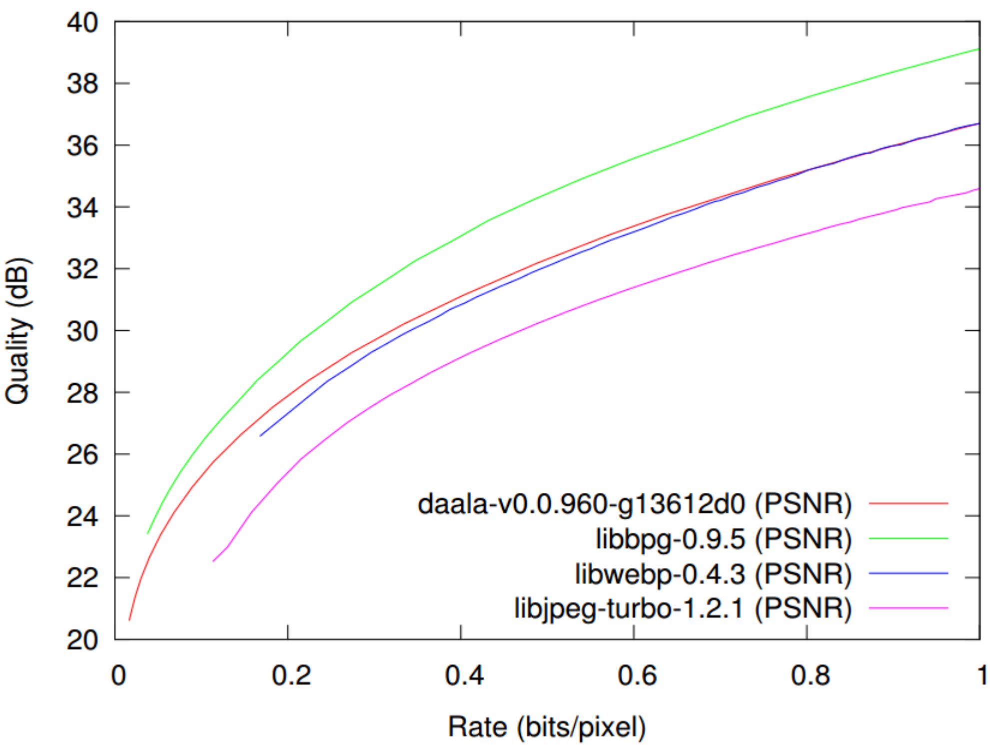


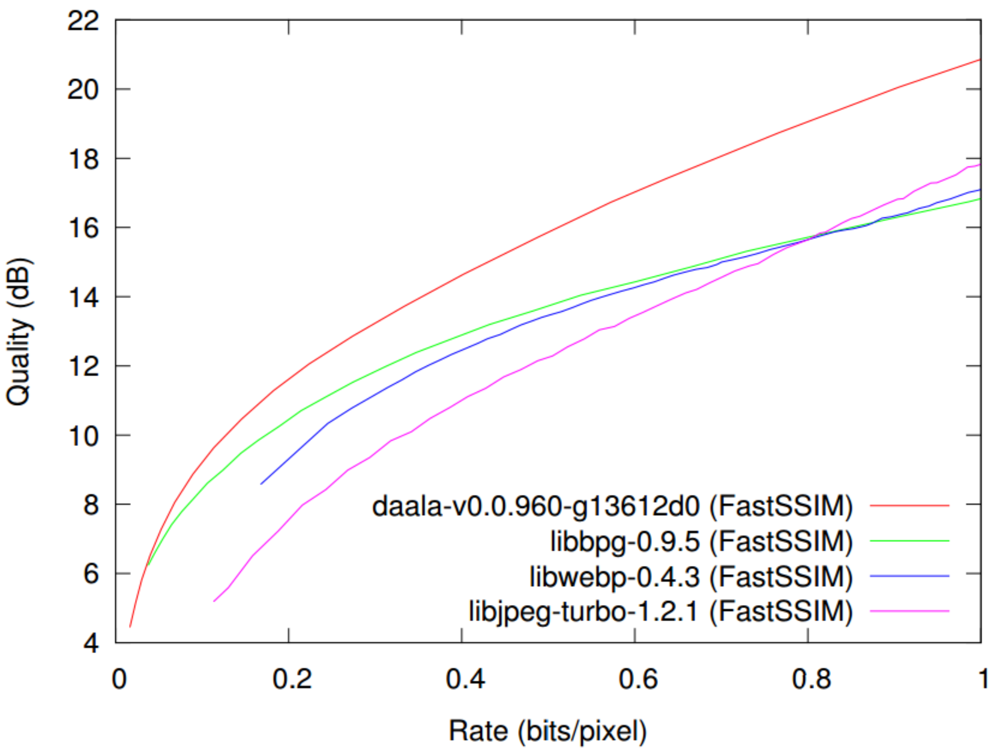
Paint (Bilinear Extension)

Blended Image



Results (PCS 2015 Images)





Resources

- Daala codec website: https://xiph.org/daala/
- Daala Technology Demos: https://people.xiph.org/~xiphmont/demo/daala/
- Git repository: https://git.xiph.org/
- IRC: #daala channel on irc.freenode.net
- Mailing list: daala@xiph.org

Questions?