The Daala Video Codec Project
Next-next Generation Video

Timothy B. Terriberry

Mozilla & The Xiph.Org Foundation

- Patents are no longer a problem for free software
  - We can all go home

- Except... not quite

Carving out Exceptions in OIN

program or (b) a user interface specifically designed to implement such capability.

Subject to the limitations with respect to Sony and Philips as set forth below, to the extent that any of the Linux Environment Components identified in this edition contain audio, still video and/or motion video Codecs, such Codecs shall be deemed not to constitute a Linux Environment Component unless expressly set forth in Table 0 below. For the purpose of the foregoing, "Codec" shall mean a component of a program capable of performing transformations of a video, still image or

(Table 0 contains one Xiph codec: FLAC)

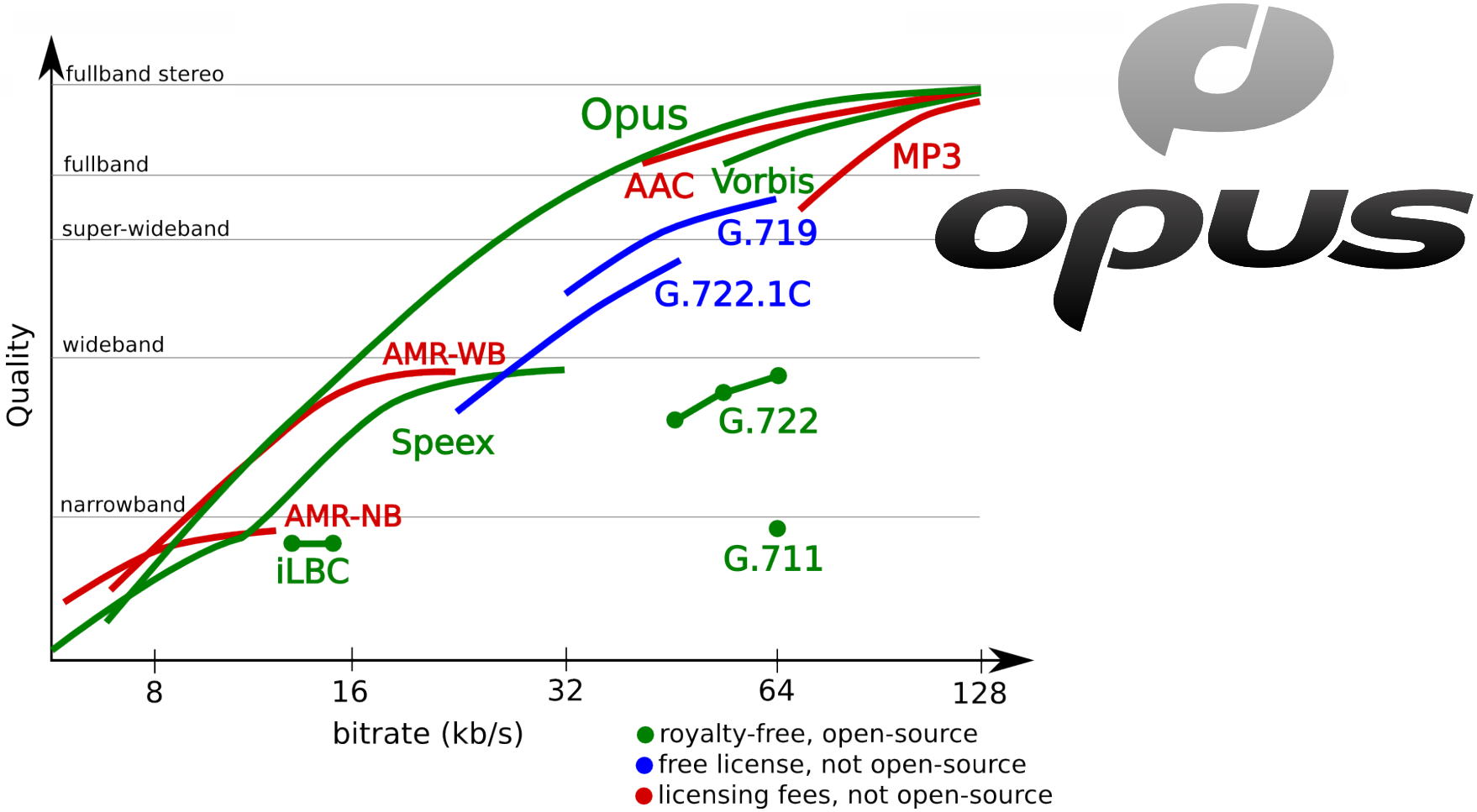
Why This Matters

- Encumbered codecs are a billion dollar toll-tax on communications
  - Every cost from codecs is repeated a million fold in all multimedia software
- Codec licensing is anti-competitive
  - Licensing regimes are universally discriminatory
  - An excuse for proprietary software (Flash)
- Ignoring licensing creates risks that can show up at any time
  - A tax on success

The Royalty-Free Video Challenge

- Creating good codecs is hard
  - But we don't need many
  - The best implementations of patented codecs are already free software
- Network effects decide
  - Where RF is established, non-free codecs see no adoption (JPEG, PNG, FLAC, …)
- RF is not enough
  - People care about different things
  - Must be better on *all* fronts

We Did This for Audio



The Daala Project

- Goal: Better than HEVC *without* infringing IPR
- Need a better strategy than “read a lot of patents”
  - People don’t believe you
  - Analysis is error-prone
    - Try to stay *far* away from the line, but...
    - One mistake can ruin years of development effort
    - See: H.264 Baseline

Strategy

- Look for some elements common to broad classes of patents
  - Only need to avoid one element in a patent claim to be able to say “we don’t do that”
- Replace with fundamentally different techniques
  - Higher risk/higher reward than incremental changes
  - Can avoid vast swaths of IPR
  - Creates new challenges others haven’t solved
- Still have to read a lot of patents

Fundamentally Different

- Identified four key areas we can avoid
  - “Displaced Frame Difference” (motion compensation)
  - Adaptive loop filters (deblocking)
  - Spatial prediction (“intra”)
  - Binary arithmetic coding (specifically, context modeling)

Displaced Frame Difference

- Motion Compensation
  - Copy blocks from an already encoded frame (offset by a motion vector)
  - Subtract from the current frame
  - Code the residual

Input

⊖

Reference frame

=

Residual

Displaced Frame Difference

- The “displaced frame difference” (DFD) is the term of art for that residual
- Not in and of itself patentable!
  - At least, not anymore...
- But found as *one* element of nearly *all* patent claims on motion compensation

What We Do Instead

- “Perceptual” Vector Quantization
- Based on work in Opus designed to *preserve energy* (film grain, fine details, etc.)

Perceptual Vector Quantization

- Separate “gain” (energy) from “shape” (spectrum)
  - Vector = Magnitude × Unit Vector (point on sphere)
- Potential advantages
  - Can give each piece different rate allocations
    - Preserve energy (contrast) instead of low-passing
  - 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!
  - Better representation of coefficients

What does PVQ have to do with DFDs?

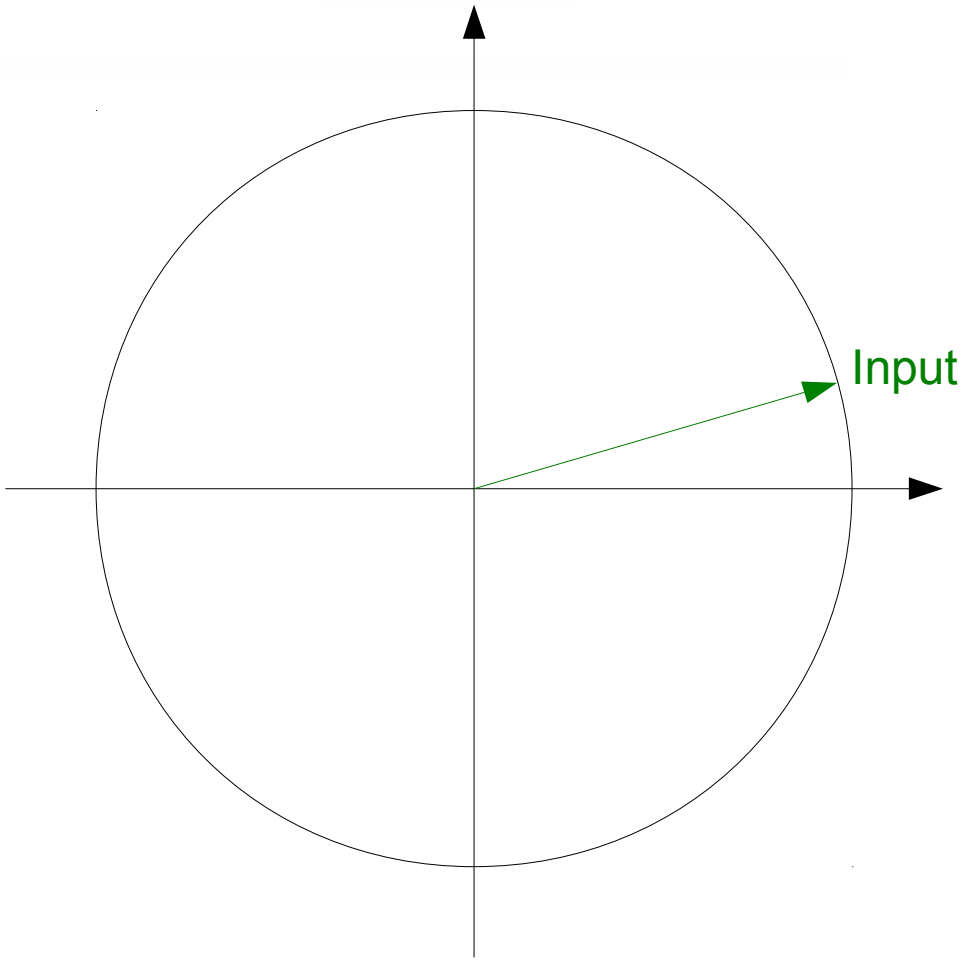
- Subtracting and coding a residual loses energy preservation
  - The “gain” no longer represents the energy of the original signal
- But we still want to use predictors
  - They do a *really* good job of reducing what we need to code

What Does Prediction Really Do?

- Prediction changes the *probability* of points near the predictor
  - Highly probable things are cheap to code
  - With DFDs, “highly probable” means “near zero”
- Predicting gains is easy
  - Subtract gain of predictor
- Enumerating points on a sphere near an arbitrary point (to model probabilities) is hard
  - Solution: *Transform* the space so we can single out points near the predictor

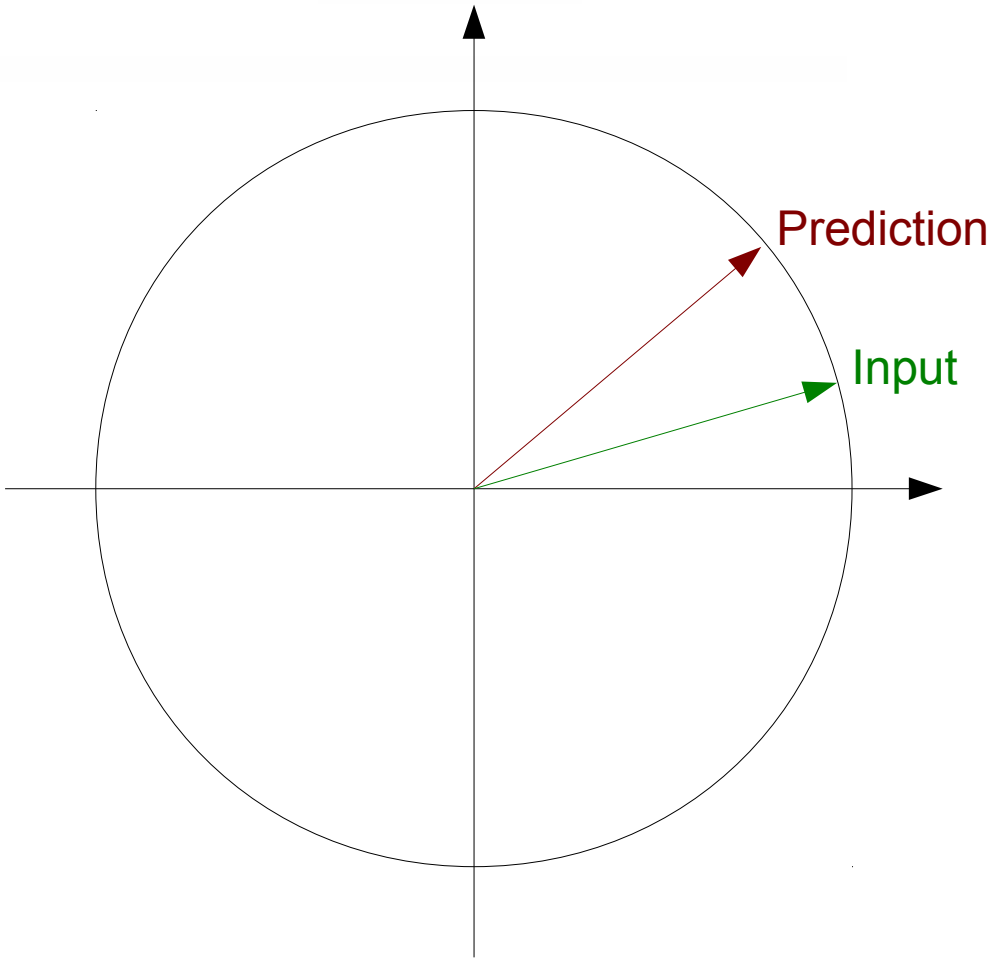
2-D Projection Example

- Input



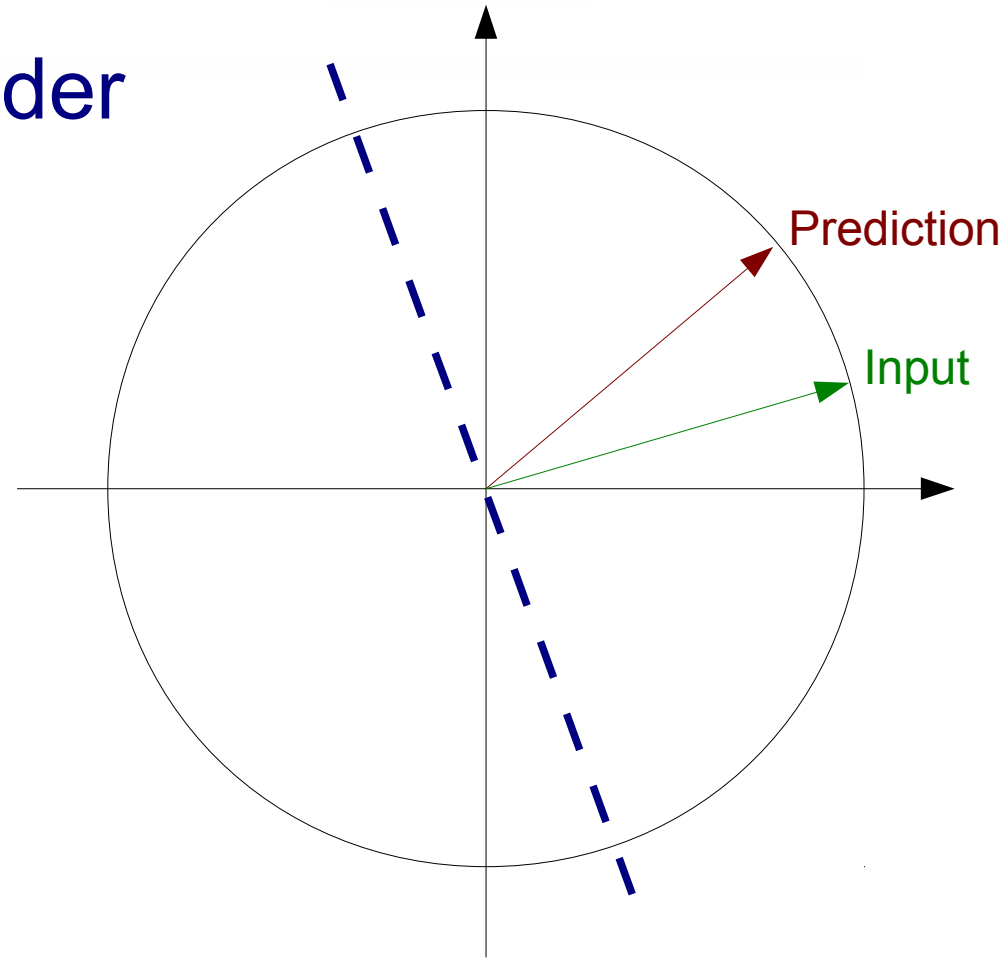
2-D Projection Example

- Input + Prediction



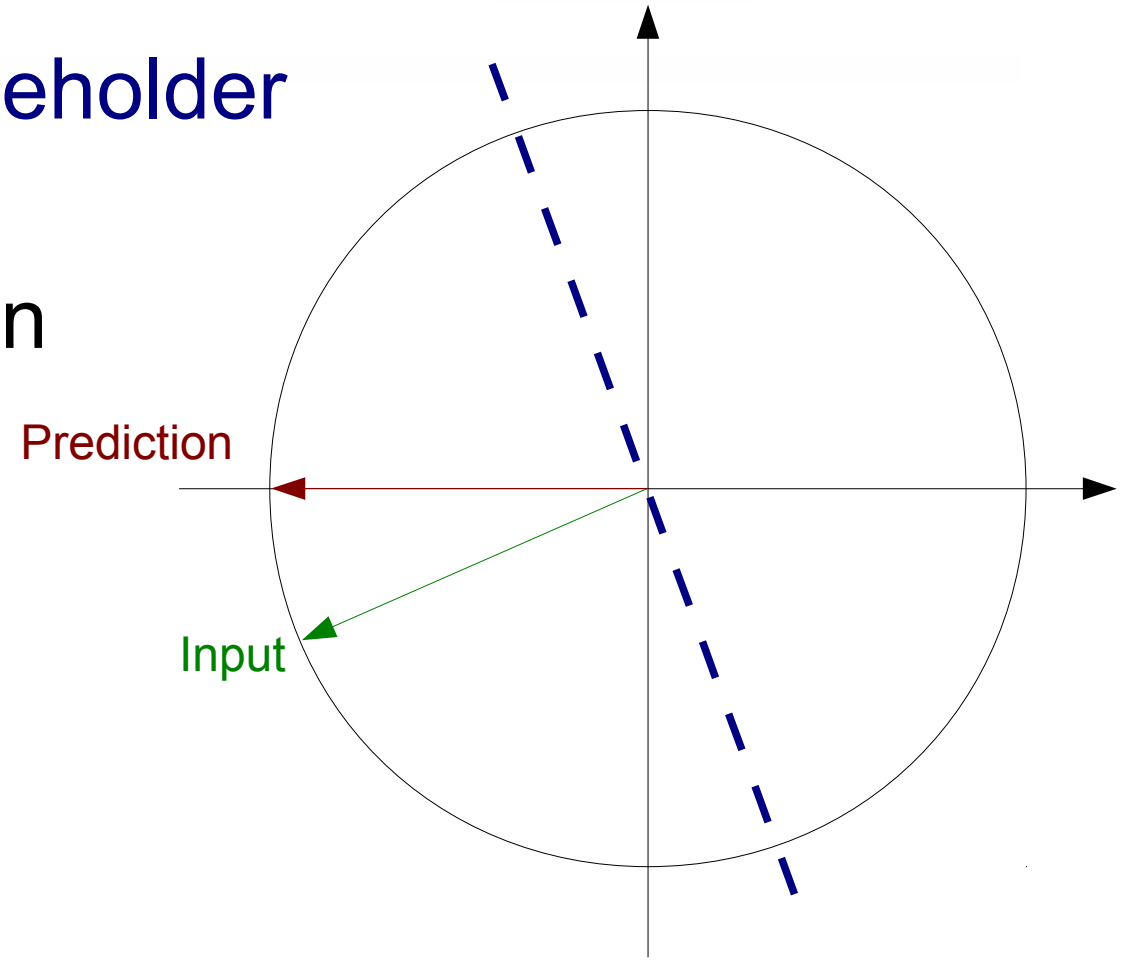
2-D Projection Example

- Input + Prediction
- Compute Householder Reflection



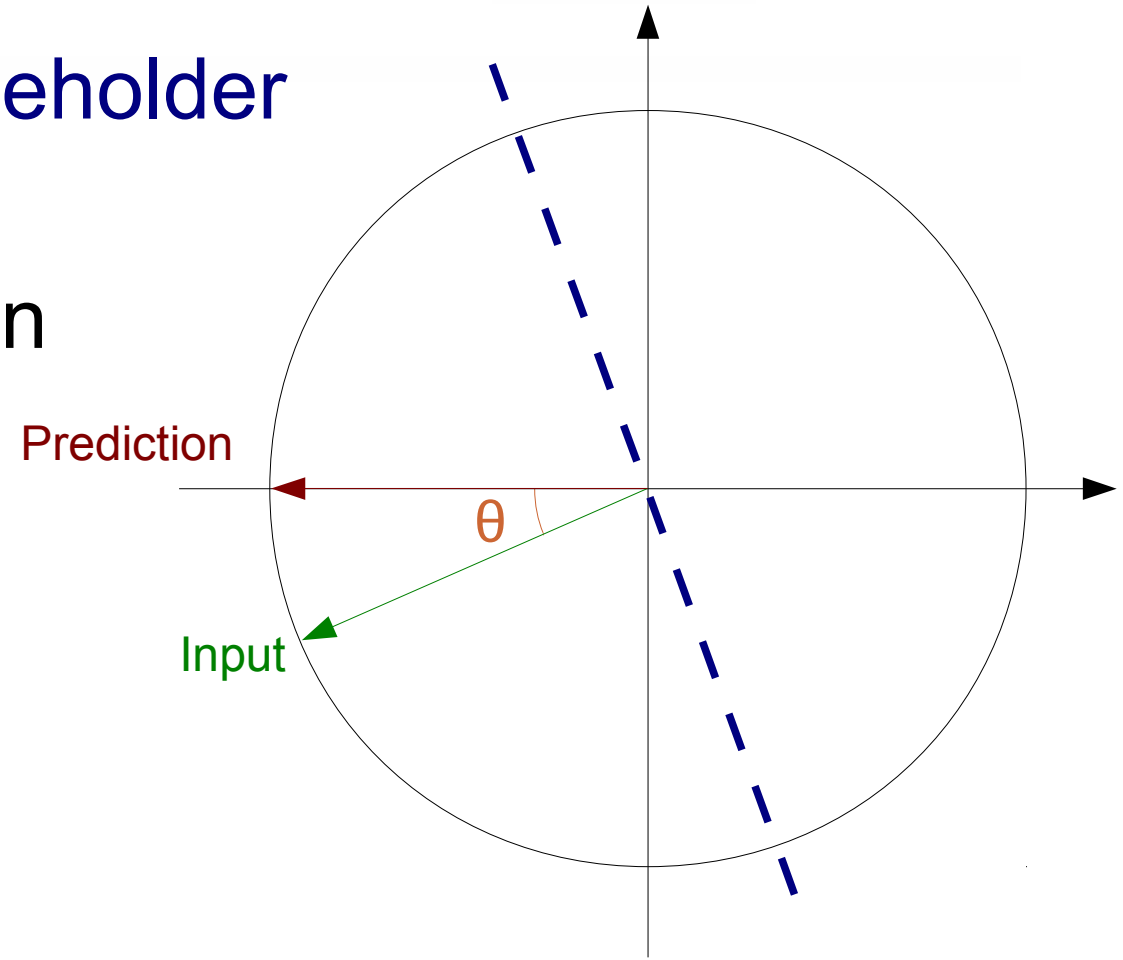
2-D Projection Example

- Input + Prediction
- Compute Householder Reflection
- Apply Reflection



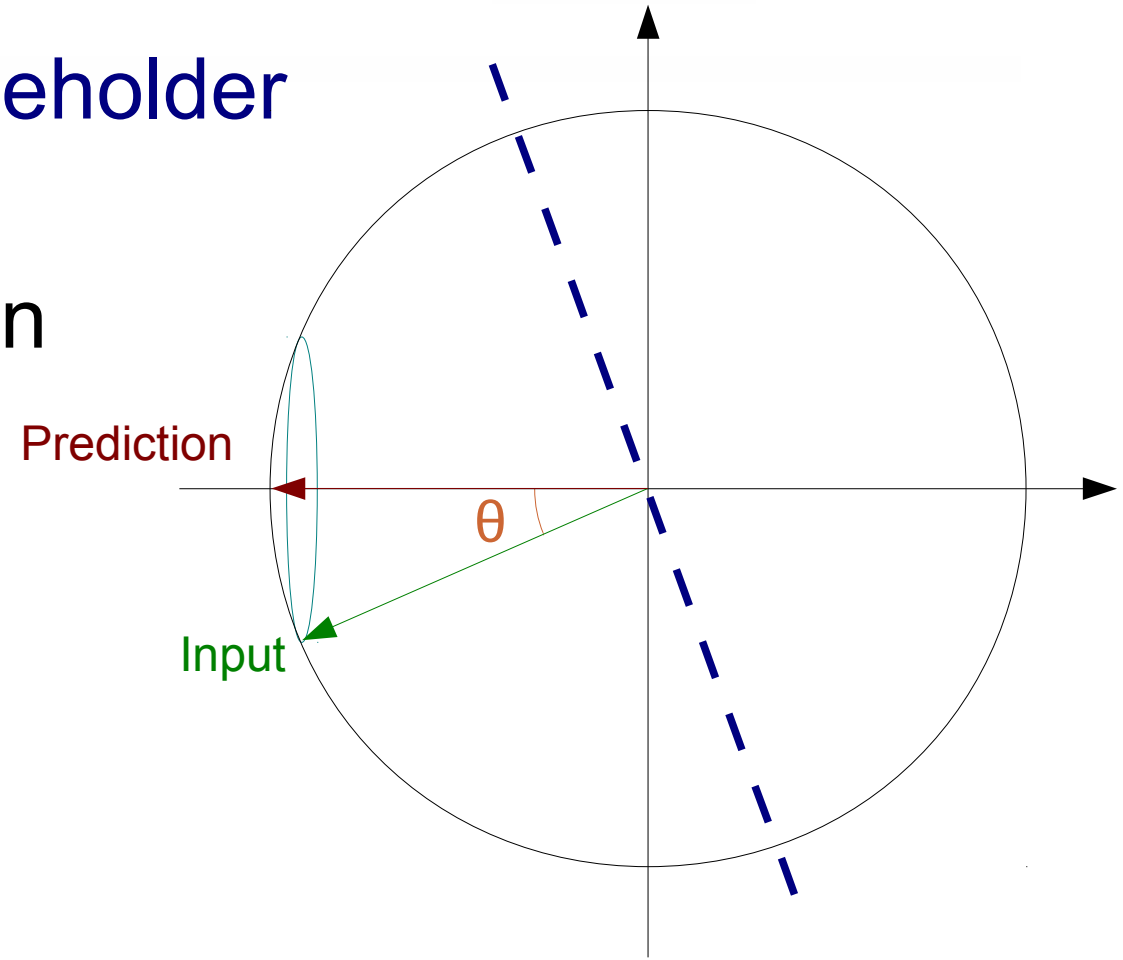
2-D Projection Example

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



2-D Projection Example

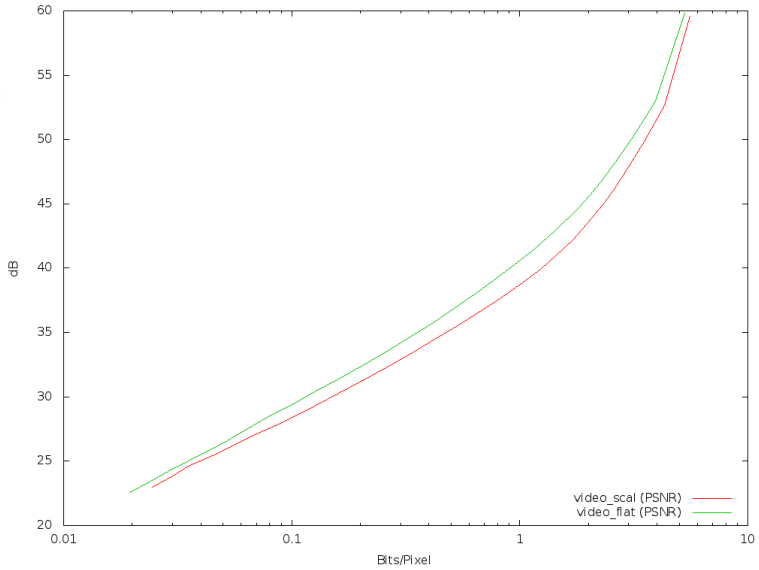
- Input + Prediction
- Compute Householder Reflection
- Apply Reflection
- Compute & code angle
- Code other dimensions



What does this accomplish?

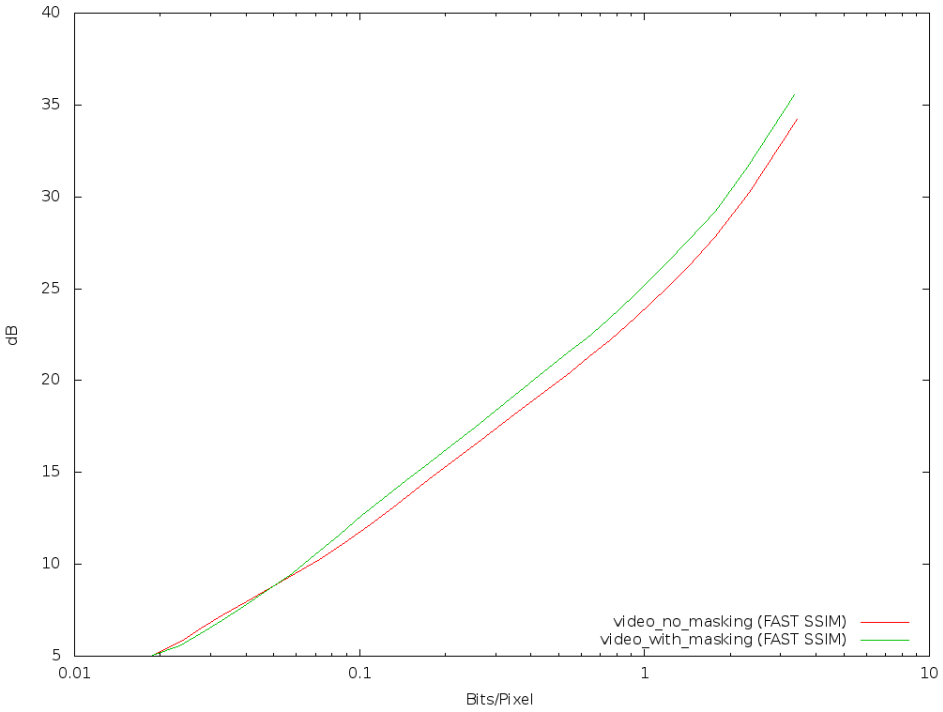
- Creates another “intuitive” parameter, θ
  - “How much like the predictor are we?”
  - $\theta = 0 \rightarrow$ use predictor exactly
- Remaining N-1 dimensions are coded with VQ
  - We know their magnitude is gain*sin(θ)
- Instead of subtraction (translation), we’re scaling and reflecting
  - Whatever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

And it works!



PSNR for PVQ vs. Scalar Quantization (flat quantization, no activity masking)

FastSSIM for turning on activity masking



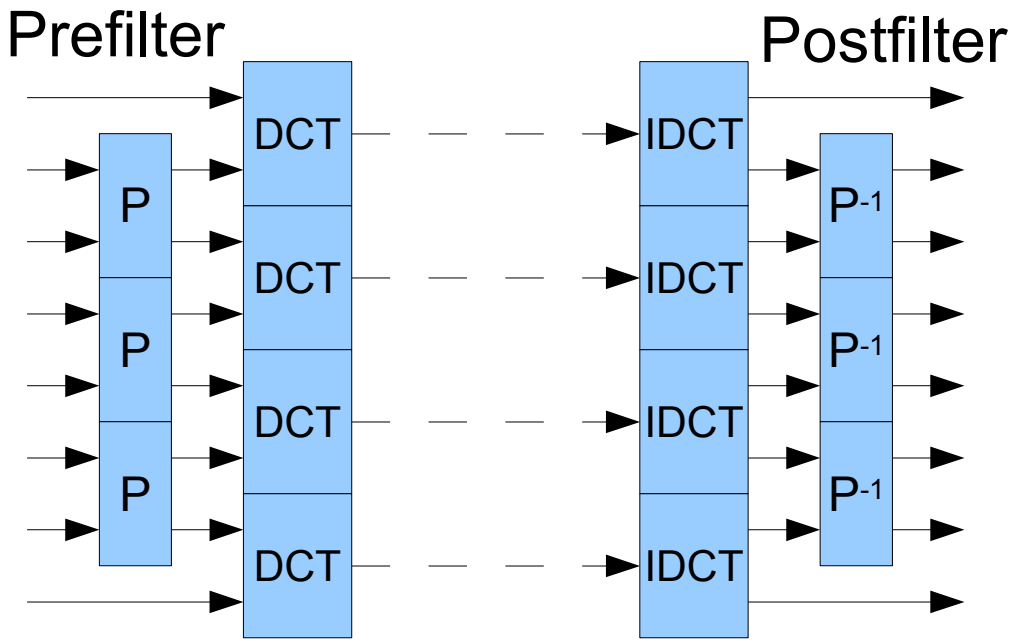
Other Differences...

Loop Filters

- “Loop filters” filter block edges to remove blocking artifacts
  - Adaptive: filter strength depends on the amount of difference across the block edge
  - Not invertible
- Simple filters used in H.263 (and Theora!)
  - Very simple to keep CPU cost low
- Since H.264 there’s been an explosion of complex filter designs
  - And patents

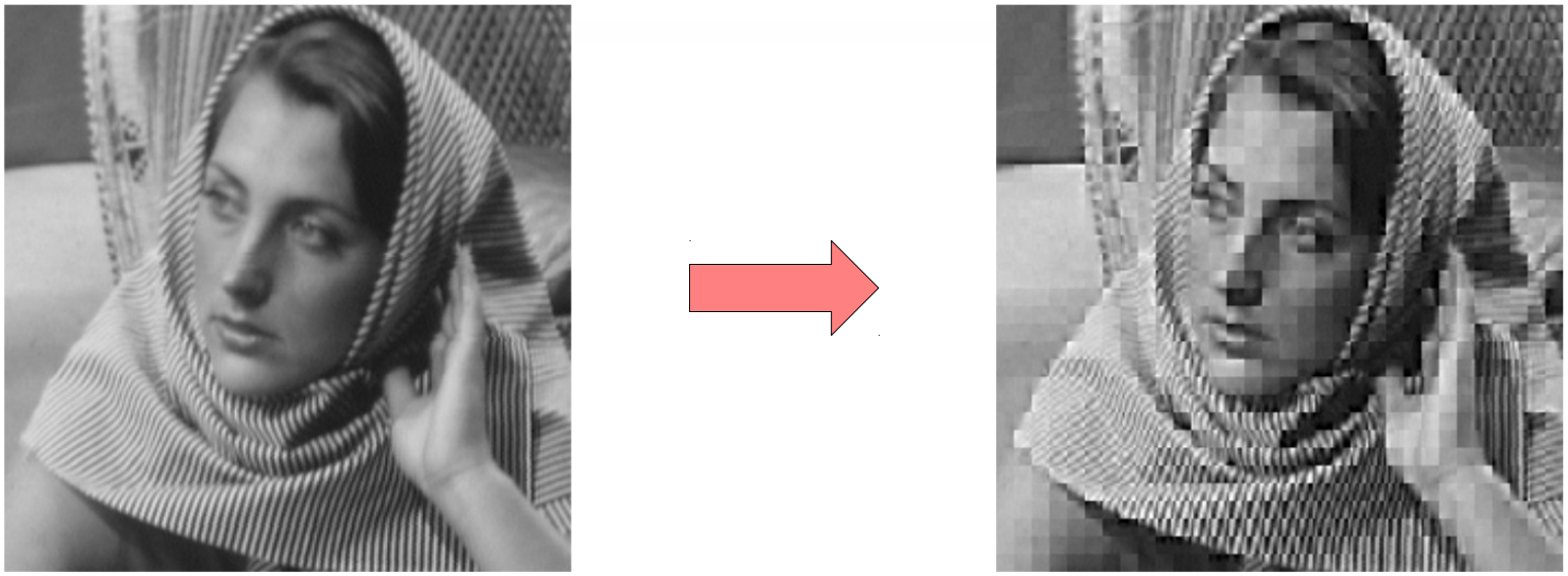
Lapped Transforms

- *Non*-adaptive, *invertible* deblocking post-filter
- Encoder applies the inverse (a blocking filter)
- Technique dates back to the 90's



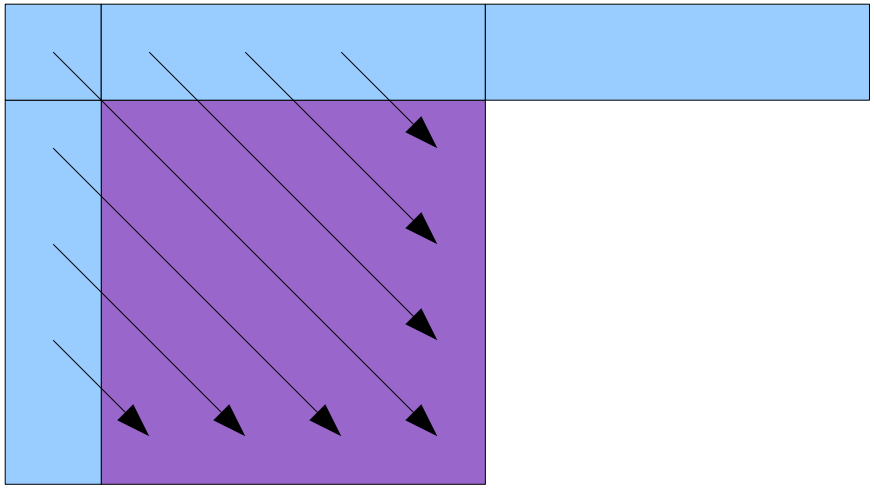
Blocking Filter

- Prefilter makes things blocky



Spatial (Intra) Prediction

- Predict a block from its causal neighbors
- Explicitly code a *direction* along which to copy
- Extend boundary of neighbors into new block along this direction



Intra Prediction with Lapped Transforms

- We can't copy pixels until we undo the lapping
  - We can't undo the lapping until we've predicted those pixels
- Don't copy pixels: copy transform coefficients
  - Currently just horizontal and vertical directions
  - Chroma (color) predicted from luma (brightness)
- Not as good, but we try to make up for it elsewhere (e.g., lapping itself)

Binary Arithmetic Coding

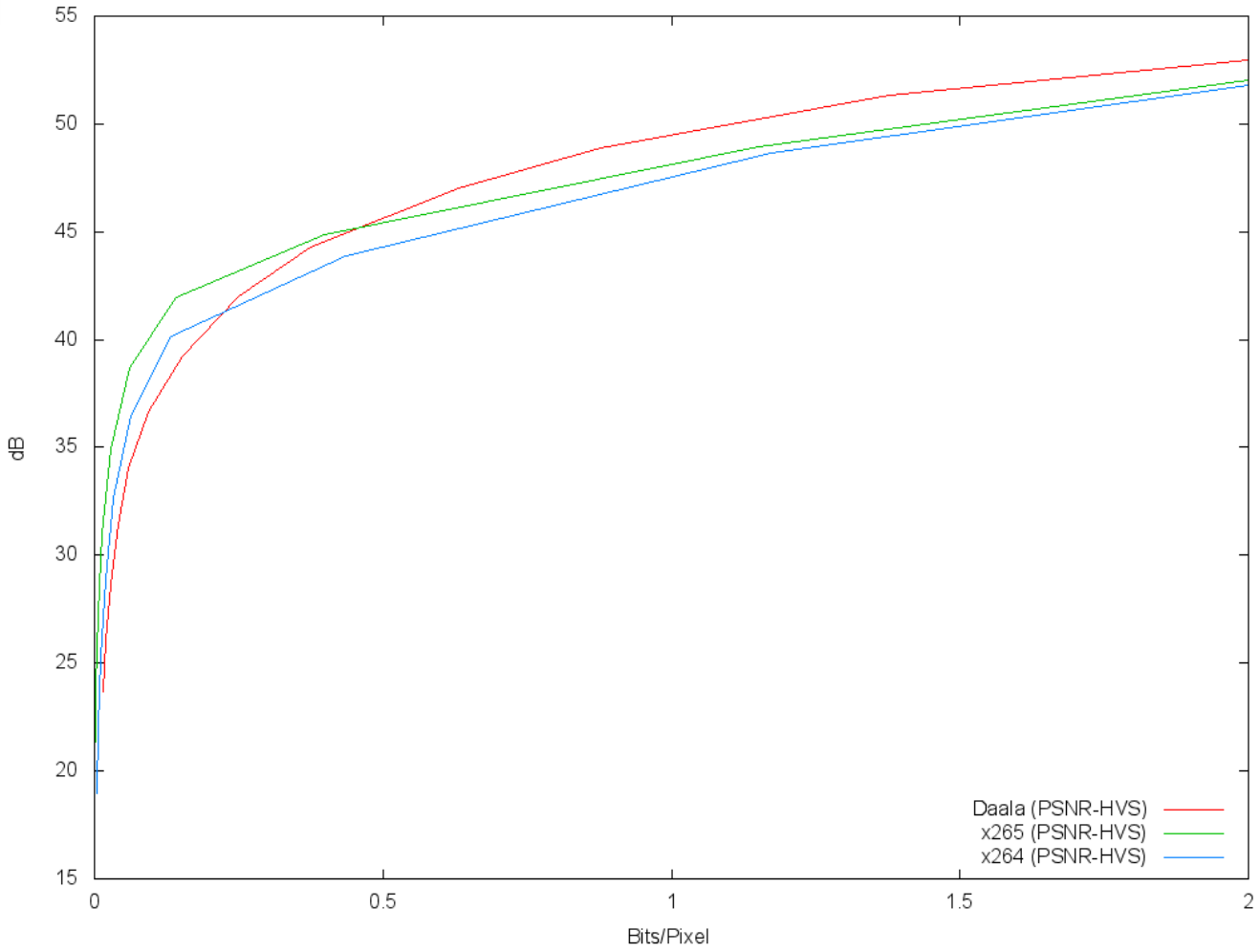
- Code only binary decisions
  - Actual cost in bits depends on probability
  - Very cheap to code 1 symbol
  - Need to code a lot of symbols (not parallelizable)
- Probability modeling
  - Simple 1-byte lookup tables
- Non-binary values
  - Various schemes for converting to binary decisions (“binarization”)

Non-Binary Arithmetic Coding

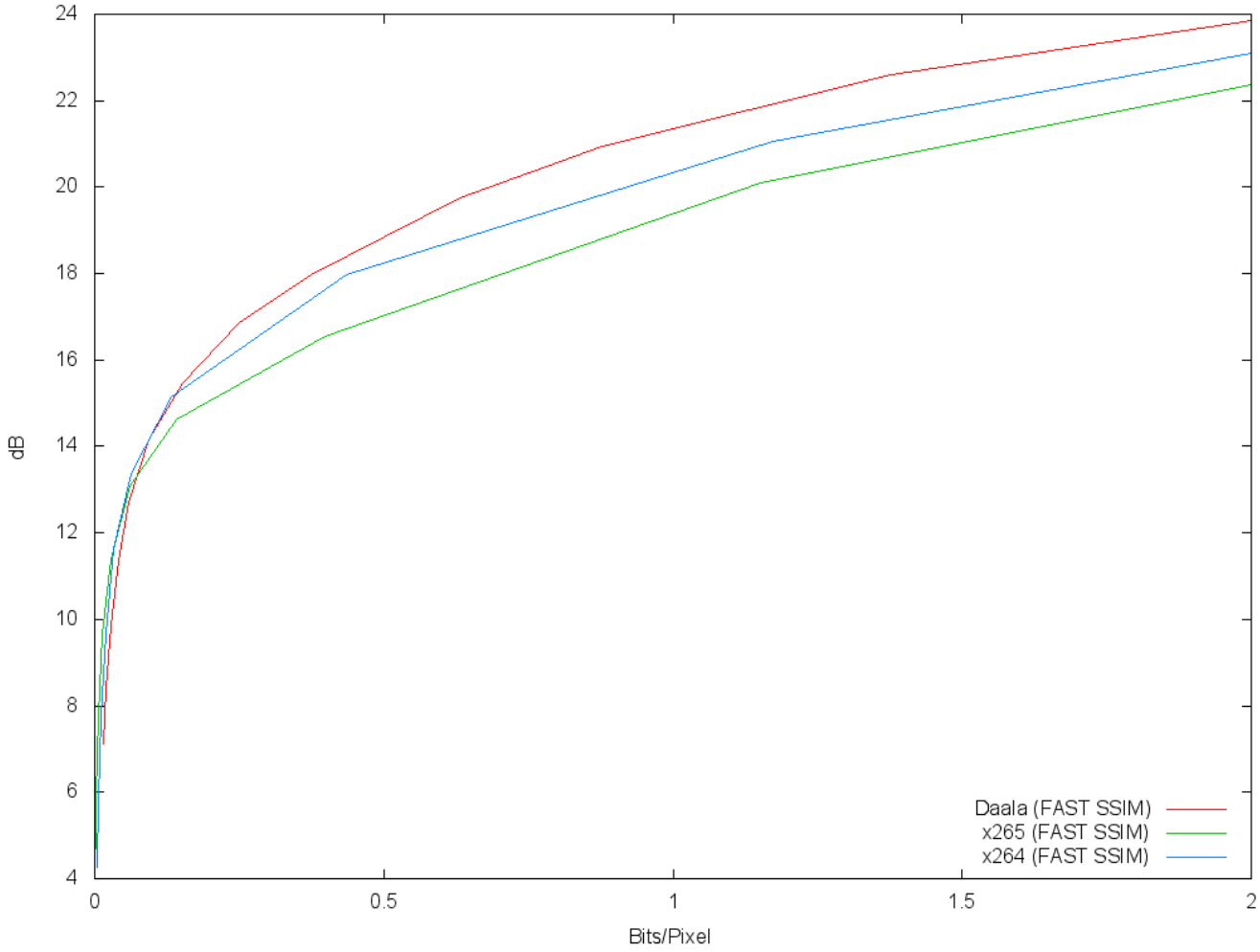
- Code values with up to *16* possibilities
  - Equivalent to 4 binary decisions
  - More expensive, but not 4x more expensive
    - A lot of overheads are *per-symbol*
  - Effectively parallel!
- One byte cannot model 16 probabilities
  - Use, e.g., *expected value* plus distribution shape (Laplace, Exponential) and compute on the fly
- Convert things to hex, not binary!
  - Often combine multiple values into one symbol

How Are We Doing?

PSNR-HVS-M Results on 19 Sequences

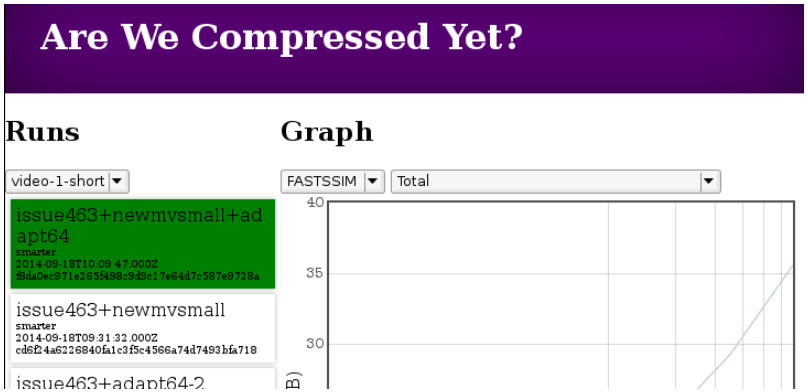


FastSSIM Results on 19 Sequences



Are We Compressed Yet?

- https://arewecompressedyet.com/
  - Will run metrics on any git commit (we're happy to add your repository, just ask)
  - Amazon EC2 instances, so results in a few minutes
  - Details on setup at https://wiki.xiph.org/AreWeCompressedYet



Daala Demo Pages

- https://people.xiph.org/~xiphmont/demo/
  - Next Generation Video: Introducing Daala, Part 1
  - Introducing Daala, Part 2: Frequency Domain Intra Prediction
  - Introducing Daala, Part 3: Time/Frequency Resolution Switching
  - Introducing Daala, Part 4: Chroma from Luma
  - Daala, Part 5: Painting Images for Fun and Profit
  - Daala, Part 6: Perceptual Vector Quantization
  - Daala Progress Update 20141223: Still Images

Questions?