AV1

An open-source video codec developed by the Alliance for Open Media

Informação e Codificação
2024/2025

universidade de aveiro

deti universidade de aveiro
departamento de eletrónica, telecomunicações e informática

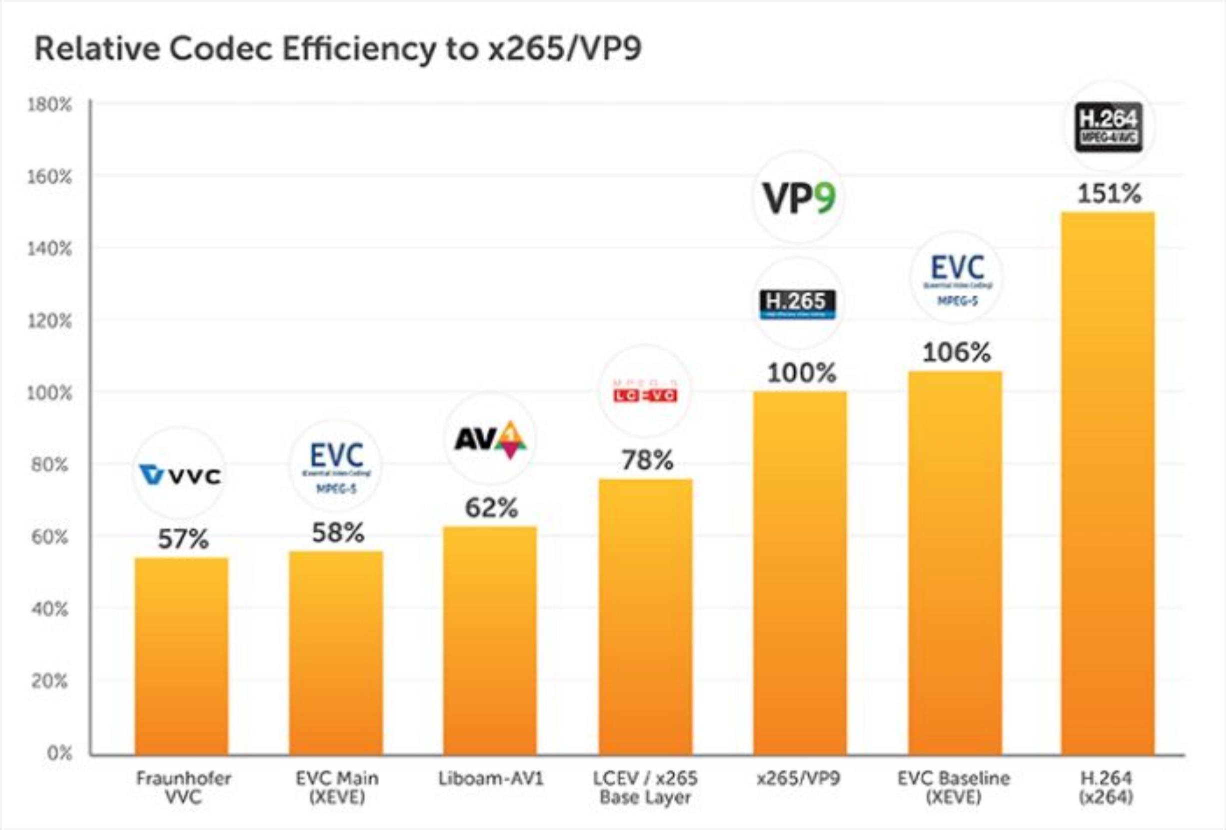
Joaquim Andrade - 93432
Leandro Rito - 92975
Tiago Mendes - 108990

What is AV1?

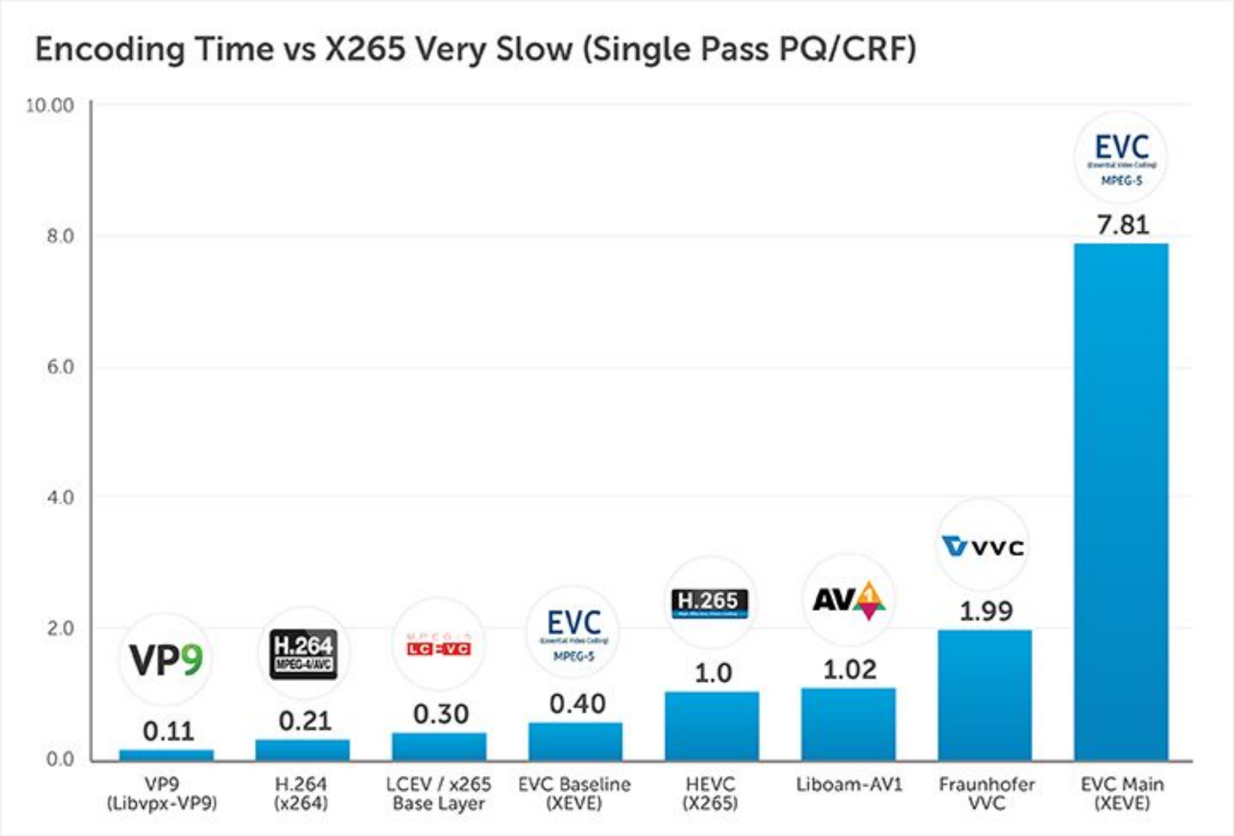
What is AV1?

- “AOMedia Video 1 (AV1) is an open, royalty-free video coding format initially designed for video transmissions over the Internet”
- It was created in 2018 by the Alliance for Open Media, a consortium of corporations including names like Amazon, Apple, Google, Intel, Meta, Microsoft, Netflix and Nvidia
- Their goal was to create a new video codec as a successor to VP9 and an alternative to HEVC

How does it Compare to other Formats?

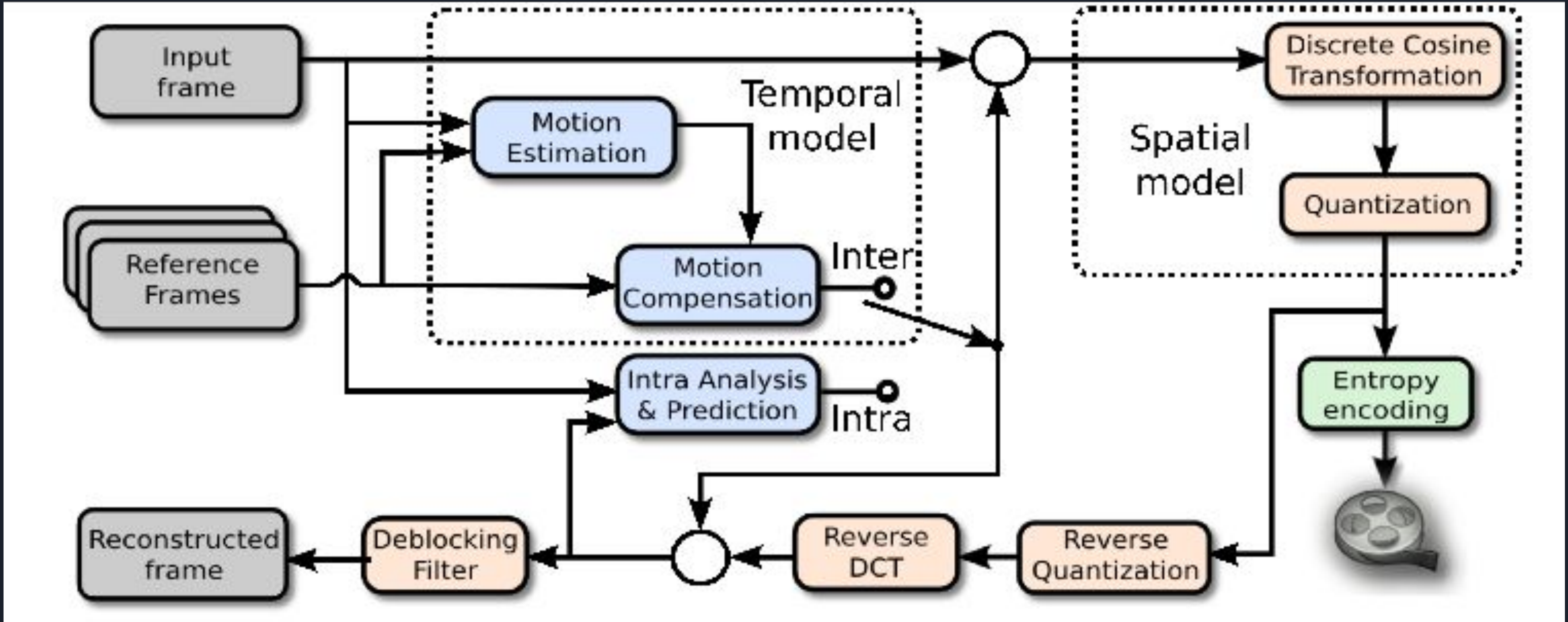


How does it Compare to other Formats?



How does it Compare to other Formats?

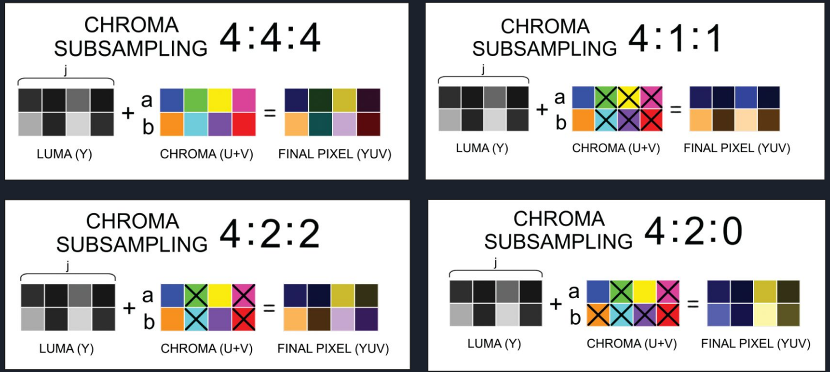
Processing stages of an AV1 encoder

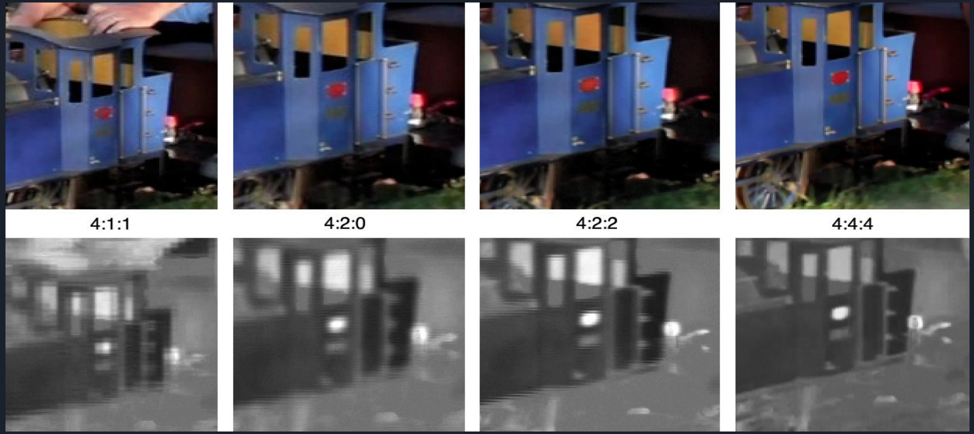


How AV1 Works:
Image Representation

Chroma and Luma vs RGB

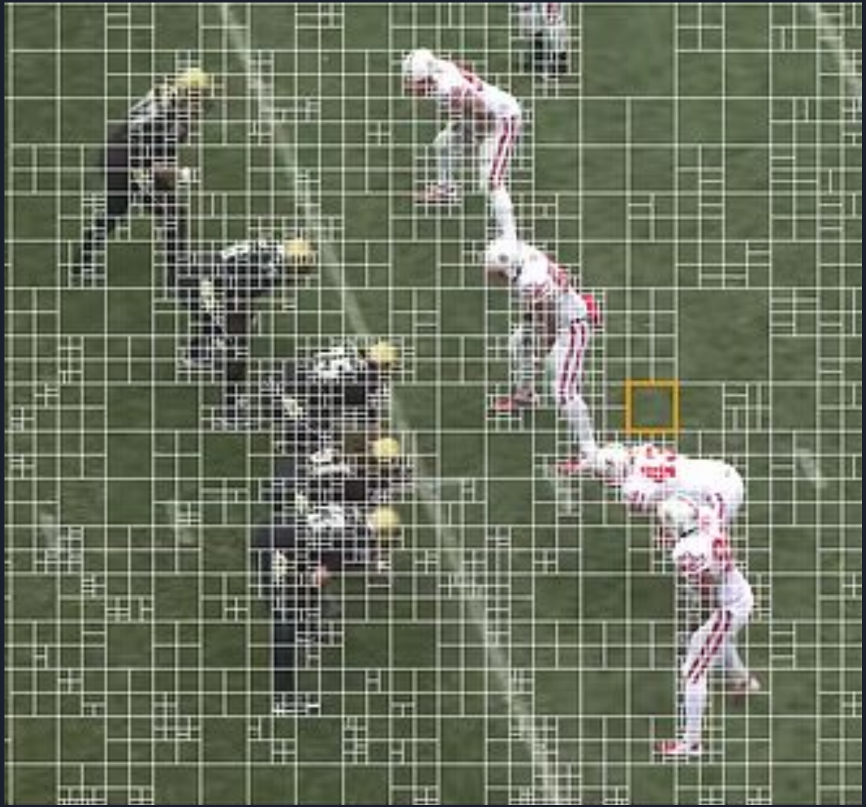
- Before getting started on AV1-specific details it's interesting to note that AV1, like some other codecs, does not use RGB to store pixel data.
- Instead it uses Luma to store the brightness of each pixel and Chroma to store its color as two values.
- This is useful because it allows for Chroma Subsampling - using less bits to store color than brightness - as our eyes are less sensitive to color changes than brightness ones.





Block Partitioning

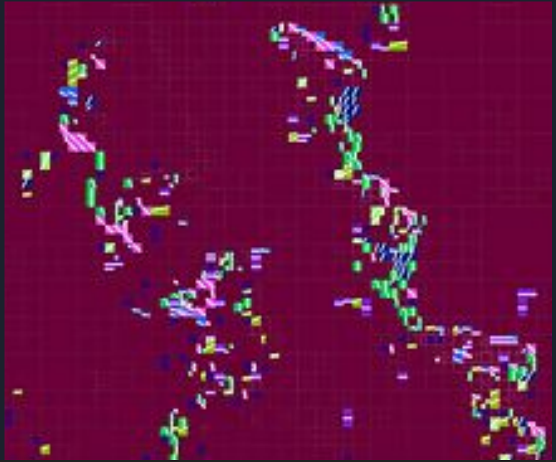
- The first step in modeling the information inside each frame is to divide it into small "**Blocks**"
- Each block's size depends on its level of detail
- Note how larger blocks are assigned to simple chunks of grass in this example, whereas smaller blocks are used for the players

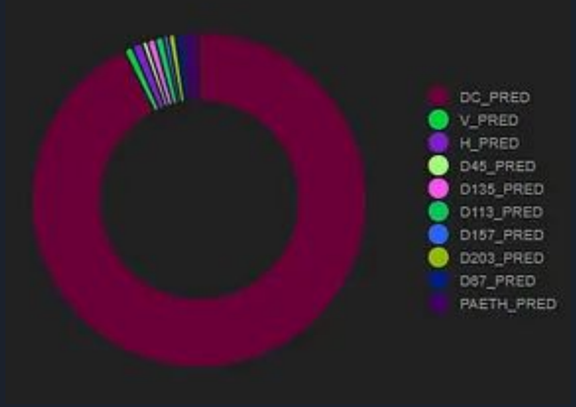


How AV1 Works:
Intra Frame Coding

Intra Prediction: Spatial Redundancy

- When coding Intra Frames, the created blocks are coded based on their relations to their neighbours
- In the example, the contents of the green grass blocks are predicted based on the boundary pixels of the adjacent ones
- The supported modes of Intra Prediction are:
  - Directional Intra Prediction
  - Non-directional Smooth Intra Prediction
  - Recursive Intra Prediction
  - Intra Block Copy
  - Chroma from Luma
  - Color Palette





How AV1 Works:
Inter Frame Coding

Inter Prediction: Motion Compensation

- Inter prediction tries to predict the motion of moving object by tracking it across frames
- Motion vectors are used to determine the new position of the block from a reference frame
- The encoder can choose a reference frame for the computation of the motion vector, from a buffer of 8 possible reference frames
- The encoder can also compensate rotation and scaling by using a matrix transformation instead of a vector

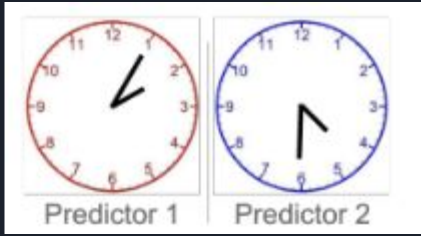
Inter Prediction: Compound Prediction

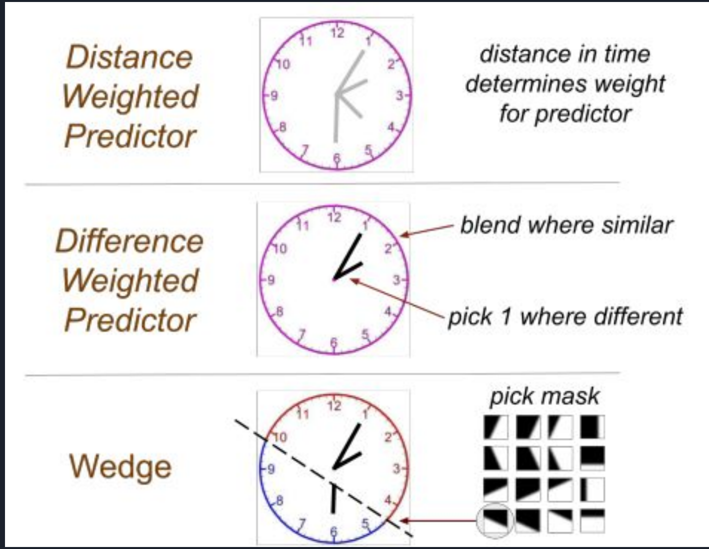
- The motion compensated predictions from two reference frames can be combined to better predict the current frame. These references can be past frames, future frames or both.
- It is most common for one reference frame to be in the past and another in the future, so that the decoder can perform interpolation (bi-directional compound). In contrast, if both are past or future frames, the decoder must perform extrapolation (unidirectional compound), which is less accurate.
- The contributions of each reference are then combined by assigning different weights to each one of them:

$$P(x, y) = w(x, y) * R_1(x, y) + (1 - w(x, y)) * R_2(x, y)$$

Inter Prediction: Compound Prediction

- The weight assigned to the prediction from each one of the two reference frames can be determined in one of 4 ways, which the encoder must specify.
- They are:
  - COMPOUND_AVERAGE
  - COMPOUND_DIST
  - COMPOUND_DIFF
  - COMPOUND_WEDGE

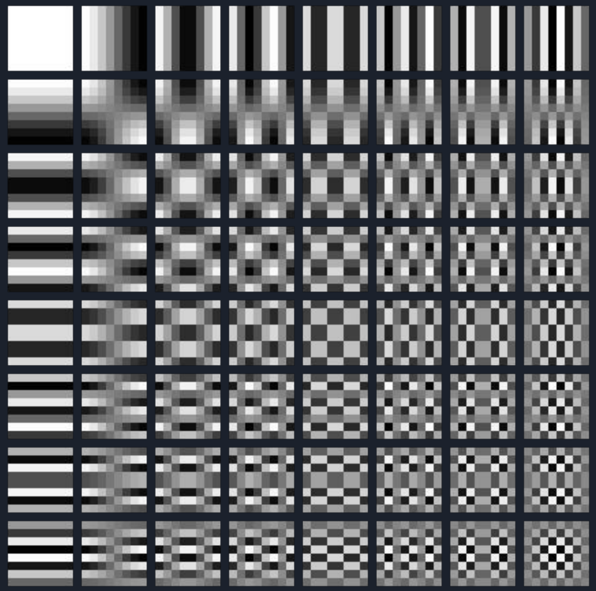




How AV1 Works: Residuals

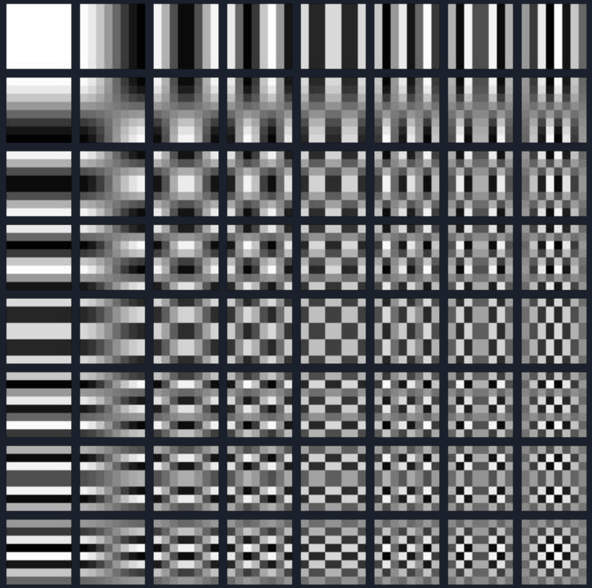
Residual Calculation and Transform Coding

- The Residual is calculated by subtracting the predicted frame data from the original frame, creating a difference signal that represents the error between prediction and actual values.
- The Residual pixel values are then transformed to the frequency domain using either one or a combination of the DCT, ADST, flipped ADST (FLIPADST), and identity transform (IDTX), resulting in a total of 16 2-D transform options.



Residual Calculation and Transform Coding

- These 4 transformation options are similar and differ only in the assumed boundary conditions. The Encoder chooses the one that will blend better with the surrounding blocks.
- This frequency transformation is done because our eyes are much less sensitive to higher frequency pattern than to lower frequency ones.
- The example image shows all DCT frequencies on an 8x8 pixel block. The DCT represents the pixel data as a combination of these. Note how the difference between the patterns on the lower right is negligible.



Quantization

- The Transform Coefficients are quantized to achieve the target bitrate.
- The encoder can remove more bits from the higher frequency components than from the lower ones, according to one of 15 predefined matrices.
- The encoder can also use more bits for the luma residuals than for the chroma.

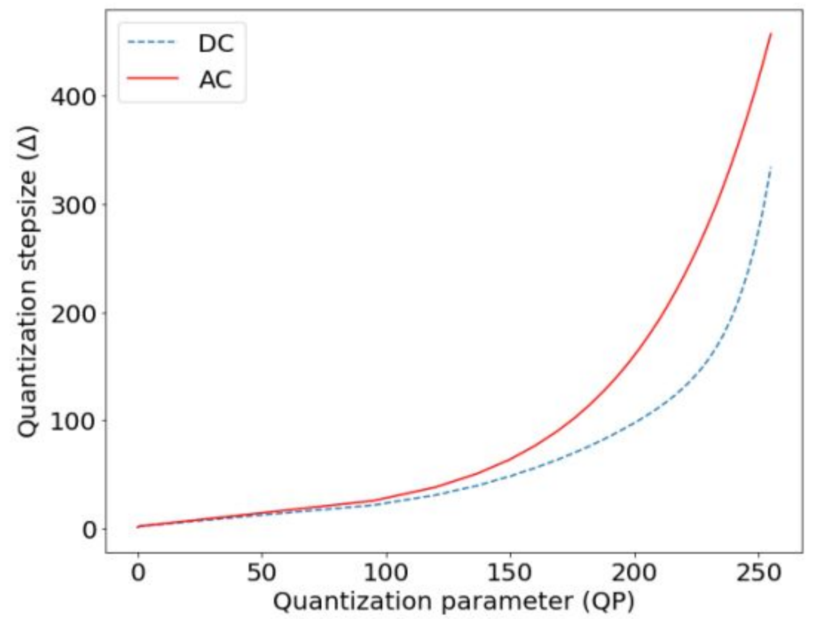


Fig. 25: The quantization parameter and quantization step size maps for DC and AC coefficients.

Entropy coding

- AV1 uses a non-binary arithmetic encoder which dynamically updates its probability model based on previously encoded symbols.
- To encode a given coefficient, one bit is reserved for its signal an then its magnitude is translated to symbols of types BR, LR or HR.
  - Base range (BR): The symbol contains 4 possible outcomes {0, 1, 2, > 2}, which are the absolute values of the quantized transform coefficient.
  - Low range (LR): It contains 4 possible outcomes {0, 1, 2, > 2} that correspond to the residual value over the previous symbols' upper limit.
  - High range (HR): The symbol has a range of [0, 2^15) and corresponds to the residual value over the previous symbols' upper limit.

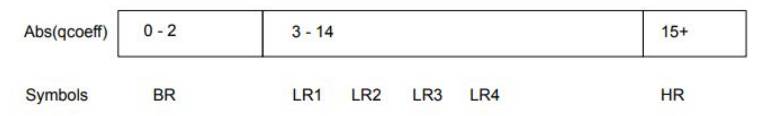


Fig. 28: The absolute value of a quantized transform coefficient V is decomposed into BR, LR, and HR symbols.

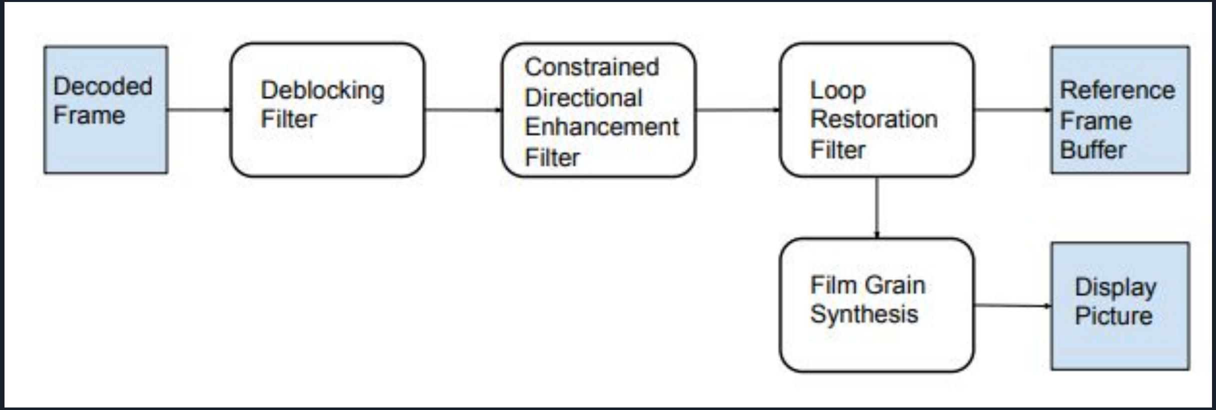
Entropy coding - Example

- If the magnitude of the coefficient we want to code, $|V| \in [0, 2]$, the BR symbol is sufficient to signal it and the coding of $|V|$ is terminated.
- Otherwise the outcome of the BR symbol will be "> 2", in which case an LR symbol is used to signal $|V|$.
- If $V \in [3, 5]$, this LR symbol will be able to cover its value and complete the coding. If not, a second LR is used to further code $|V|$. This is repeated up to 4 times, which effectively covers the range [3, 14].
- If $|V| > 14$, an additional HR symbol is coded with the remainder, $(|V| - 15)$.
- So, for example, the value $|V| = 3$ would be coded as: "BR>2", "LR=0"
- And the value $|V| = 18$ would be coded as: "BR>2", "LR>2", "LR>2", "LR>2", "LR>2", "HR=3"
- These symbols will then be passed on to the arithmetic encoder.

How AV1 Works:
Filters and Grain Synthesis

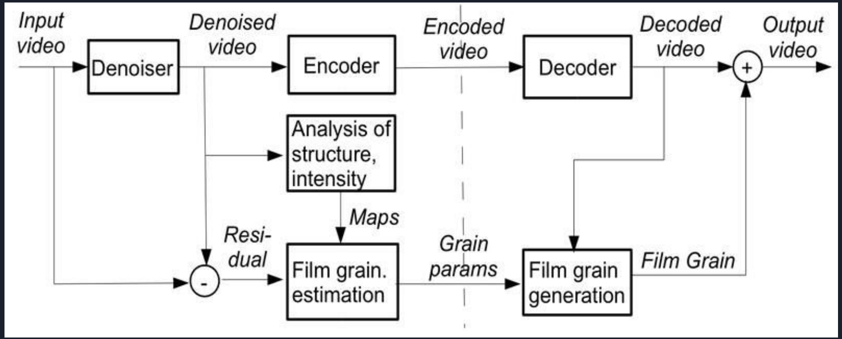
Decoding Filters

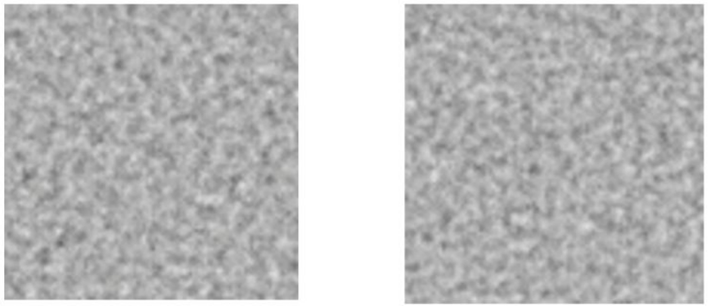
- The way AV1 models and encodes video may lead to some artifacts, namely around the block edges and around straight lines.
- In the decoder, there are 3 optional filters in place to rebuild the image as best as possible without any additional information.



Film Grain Synthesis

- AV1 has support for Film Grain Synthesis, a technique that is able to replicate video grain or "noise" by estimating its nature and intensity at the encoder and then regenerating it through a pseudorandom process at the decoder.
- This does not conserve the original information, but generates an image that looks identical





(a) Original grain (b) Synthesized grain

The End

Sources

https://medium.com/@nasirhemed/a-quick-overview-of-video-compression-and-av1-29dffbdb5cc4

https://arxiv.org/pdf/2008.06091

https://en.wikipedia.org/wiki/Chroma_subsampling?ref=hackernoon.com

https://en.wikipedia.org/wiki/AV1

https://netflixtechblog.com/bringing-av1-streaming-to-netflix-members-tvs-b7fc88e42320

https://www.wowza.com/blog/the-video-codec-landscape-2022

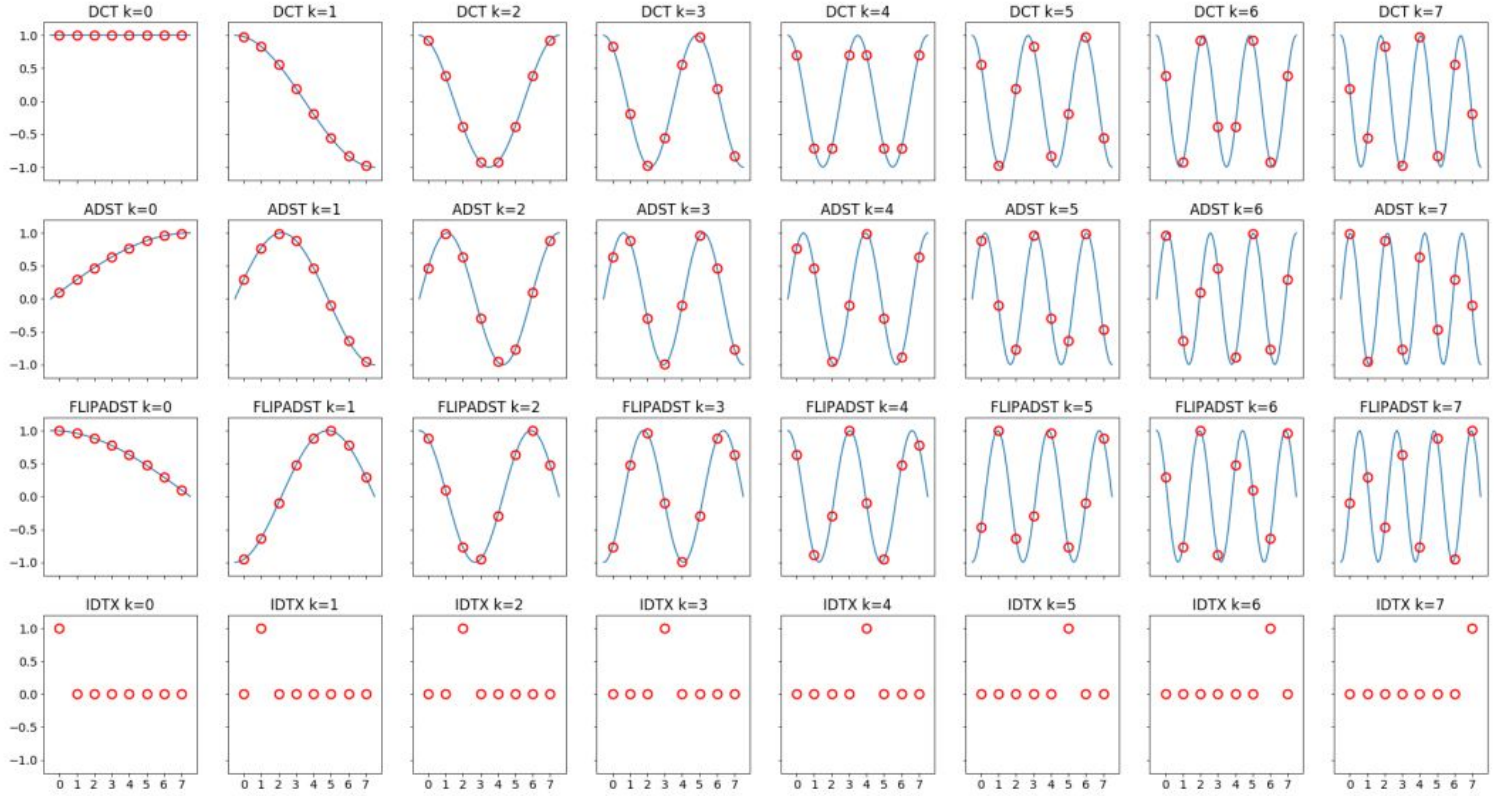


Fig. 24: Transform kernels of DCT, ADST, FLIPADST and IDTX for dimension $N = 8$. The discrete basis values are displayed as red circles, with blue lines indicating the associated sinusoidal function. The bases of DCT and ADST (a variant with a fast butterfly structured implementation) take the form of $cos(\frac{(2n+1)k\pi}{2N})$ and $sin(\frac{(2n+1)(2k+1)\pi}{4n})$ respectively, where n and k denote time index and the frequency index, taking values from $\{0, 1, ..., N-1\}$. FLIPADST utilizes the reversed ADST bases, and IDTX denotes the identity transformation.