AV2 Video Codec Architecture

Andrey Norkin
Netflix
QoMeX'25

NETFLIX

ALLIANCE FOR OPEN MEDIA

Thanks to all AOM contributors who provided the material

Slides based on material from:

Debargha Mukherjee, Urvang Joshi, Bohan Li, Lester Lu, Kruthika Sivakumar, In-Suk Chong, Joe Young, **Google**

Ryan Lei, **Meta**

Liang (Leo) Zhao, Madhu Peringassery Krishnan, Tianqi Liu, Jayasingham Adhuran, **Tencent**

Andrey Norkin, **Netflix**

Van Luong Pham, Yeqing Wu, Hilmi Egilmez, Han Gao, **Apple**

On behalf of AOM Codec Working Group

AOM Workshop @ QoMeX'2025

Alliance for Open Media

- 2015: Alliance for Open Media (AOM) was born
  - Several major companies joined forces to create an industry consortium
- 2018: AV1 release
  - AV1 - the first major codec from AOM, was released
  - Starting point was a combination of VP10, Daala, Thor codecs
  - Used extensively today by Netflix, YouTube, Meta and other streaming video service providers
- 2020:
  - AOM started exploratory work towards a **next-generation video codec**

Codec Working Group

- 2021-25
  - In the last four years, many new tools have gone through the CWG review process and have been chosen as candidates
    - **Rigorous scrutiny** for hardware decoding complexity
  - A new version of AVM is released ~ every 4 months
  - Perceptual tests being conducted

- Contributing Companies:
  - **Tools and discussions**: Alibaba, Amazon, Apple, Broadcom, Google, Intel, Meta, Netflix, Nvidia, Oppo, Samsung, Tencent, Visionular
  - **Participating in HW review**: AMD, Realtek

Recent AV2 results

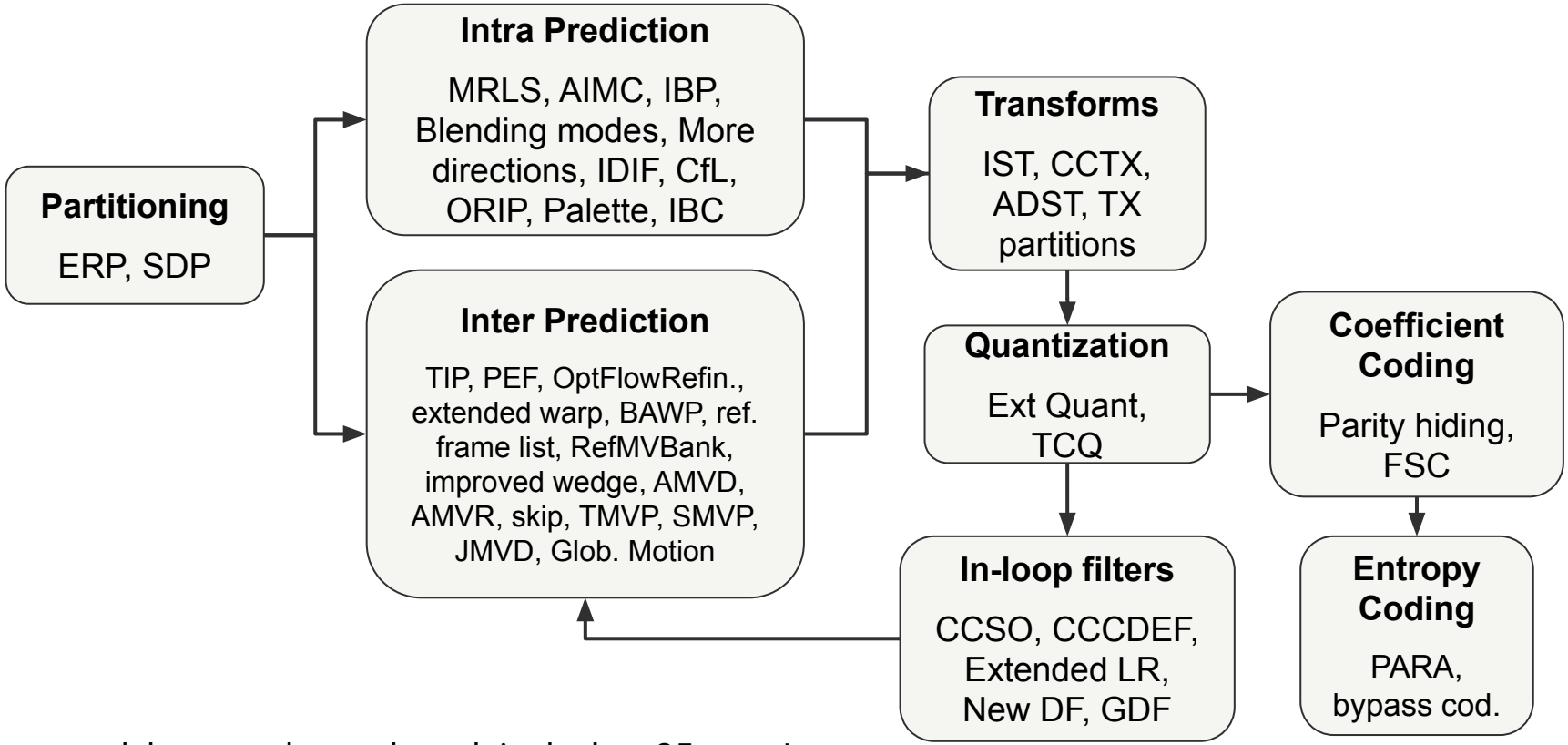
- Current status:
  - Latest anchor: AVM-v11.0.0
  - Almost all lower level coding tools finalized
  - High-Level Syntax work underway
- Summary of Objective Metrics:
  - Anchor: (modified) AV1
  - Test: AVM-v11.0.0

Config	PSNR-Y	PSNR-U	PSNR-V	PSNR-YUV	VMAF
AI	-17.50%	-34.63%	-35.47%	-19.38%	-20.76%
RA	-27.11%	-41.91%	-41.30%	-28.63%	-32.59%
LD	-21.10%	-38.02%	-37.05%	-22.75%	-23.47%

* PSNR-YUV uses (14:1:1)/16 weights for Y, U, and V

AV2 framework

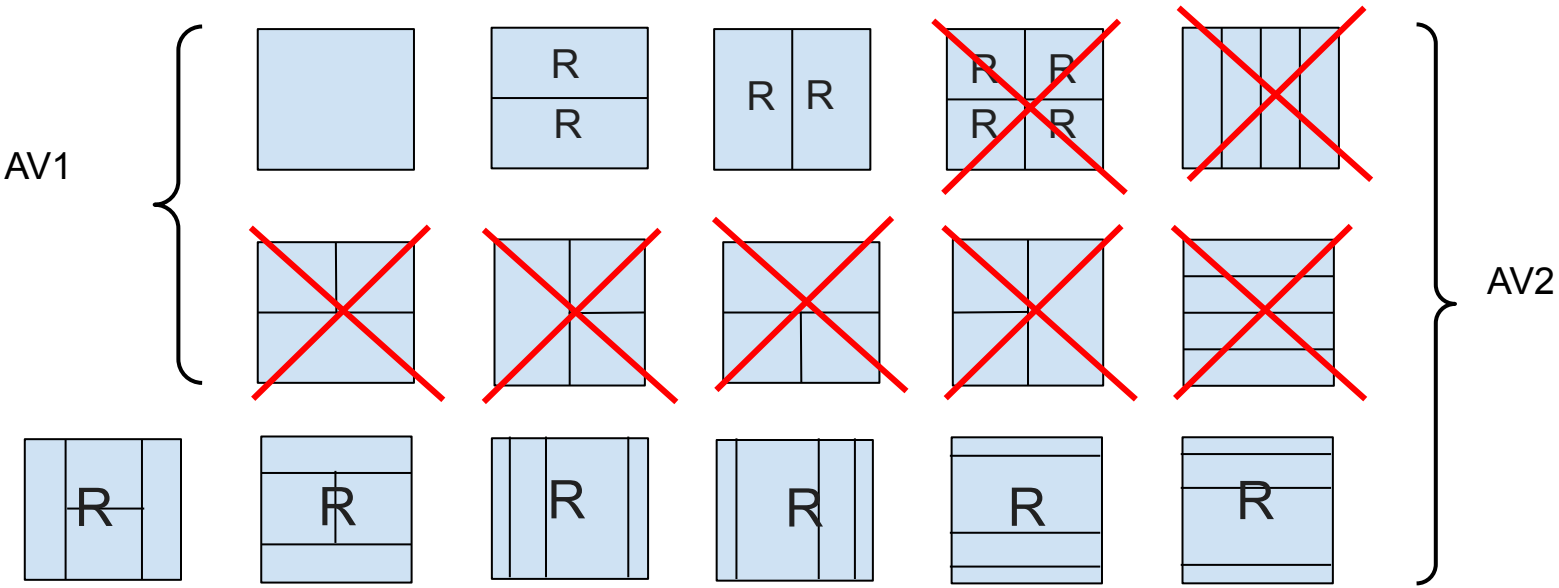
AV2 framework and tools examples



Framework has not changed much in the last 35 years!

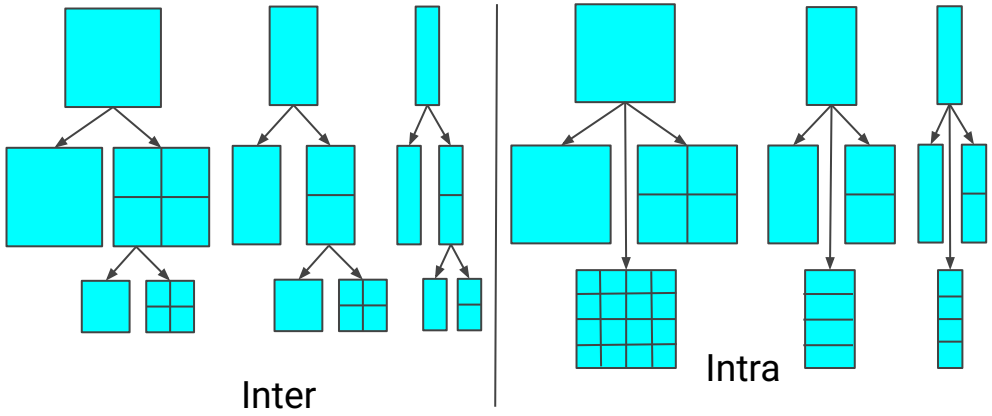
Extended Recursive Partitioning (ERP)

Recursive partitioning
128x128 (now 256x256) super-blocks down to 4x4 blocks

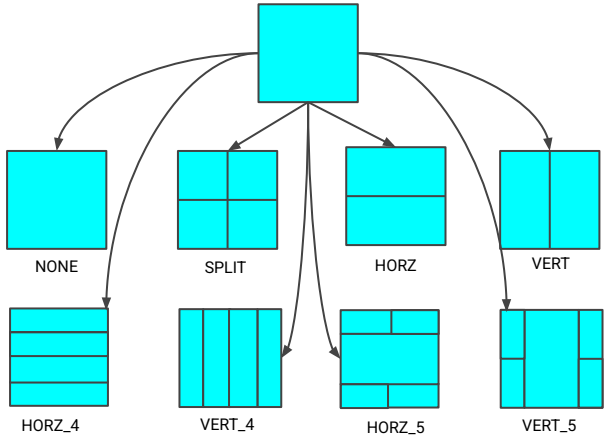


[1] Y. Chen, C.-Y. Tsai, U. Joshi, D. Mukherjee, "Extended recursive partitions", AOMedia document CWG-B084.
[2] U. Joshi, C.-Y. Tsai, Y. Chen, D. Mukherjee, "Uneven 4-way Partitions", AOMedia document CWG-D035.

Transform block partitioning



AV1 Transform Partition Scheme

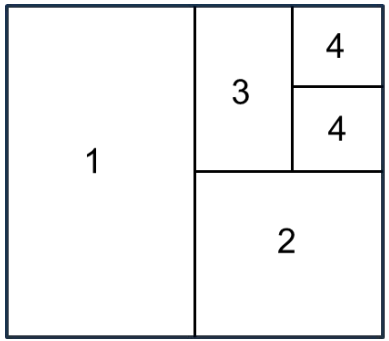


AV2 Transform Partition Scheme (inter/intra)

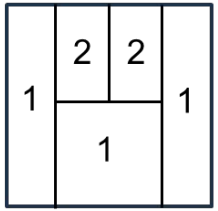
- Removal of recursive scheme used in AV1
- Same transform partition types used for square and rectangular transform blocks

Semi-decoupled partitioning (SDP)

- In AV1, partition structure is shared between luma and chroma
- In AV2 Semi Decoupled Partitions (SDP) are supported.
  - Shared luma & chroma partitioning for larger blocks
  - *Independent* luma & chroma partitioning at smaller block sizes (up to 64x64)



Luma coding block partition

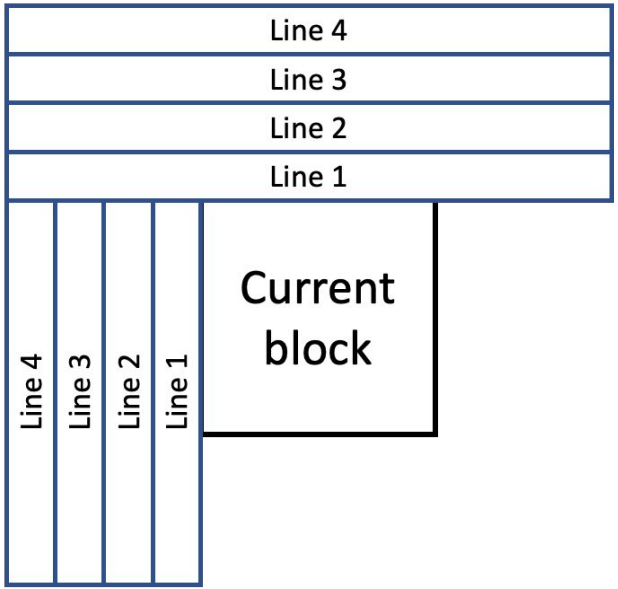


Chroma coding block partition

Intra prediction

Intra: MRLS and AIMC

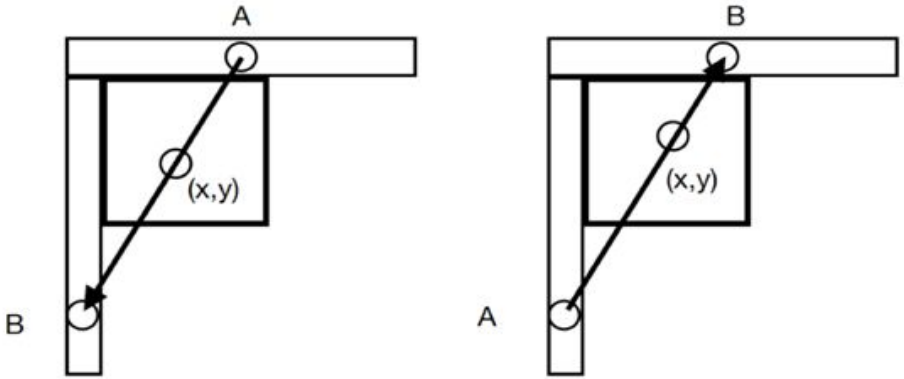
- Multiple Reference Line Selection (MRLS)
  - Select and signal one among 4 top and left reference lines for intra prediction
  - Applied to all directional prediction modes (luma)

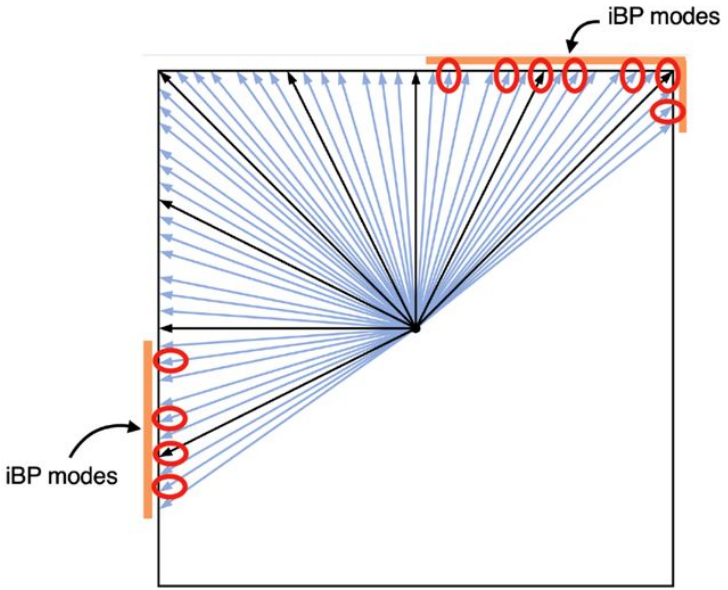


- Adaptive Intra Mode Coding (AIMC)
  - Rank available intra modes based on neighbor/colocated blocks intra mode info
  - Split the ranked modes into sets, then signal
    - mode set idx
    - mode idx within the set
  - Top ranked sets/modes are cheaper to signal

Intra Bi-Prediction

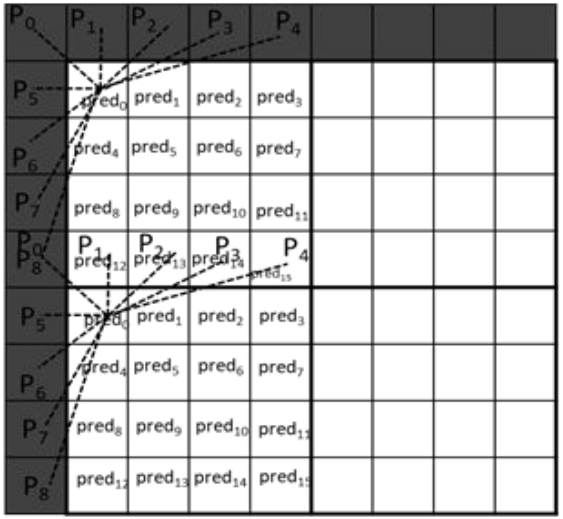
- Intra Bi-Prediction (IBP)
  - Prediction is the weighted average of A and B: *pred(x,y)* = *w* * A + (1 - *w*) * B
  - Weights defined based on (mode, x, y)
  - Adaptive on/off option based on the parity of intra prediction angles

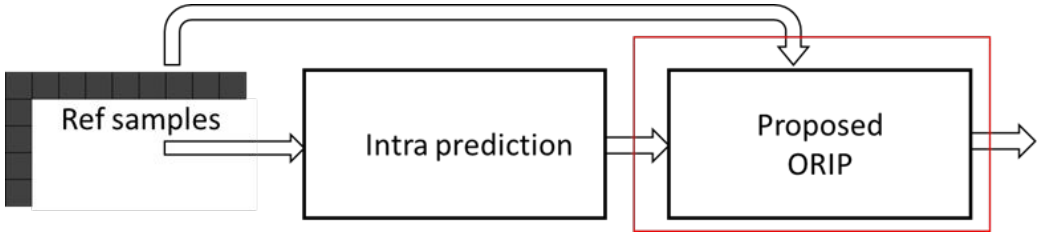




Intra: ORIP

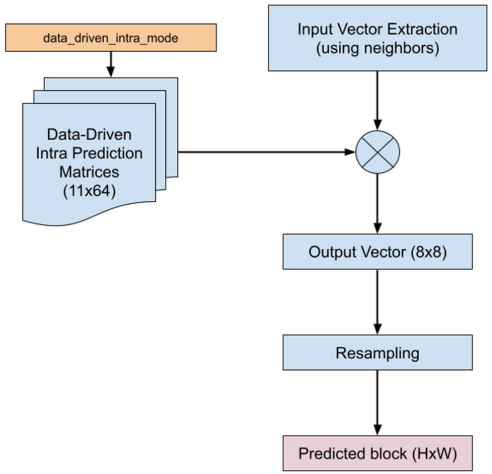
- Offset based refinement of intra prediction (ORIP)
  - The offsets are generated from top and left neighboring reconstructed and prediction samples.
  - The refinement is applied to top and left 4x4 sub-blocks only
  - ORIP is applied to Vertical, Horizontal, and Smooth mode.

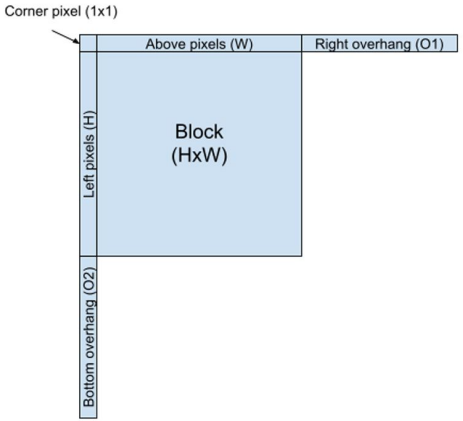


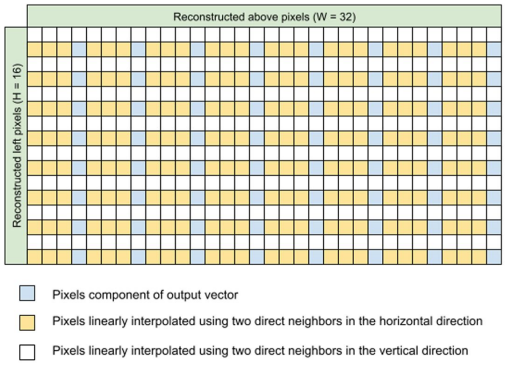


Data-driven Intra prediction

- Prediction process in DIP
  - In DIP mode, down-sampled neighboring samples are multiplied by the pre-trained matices to generate 8x8 predictions
  - Bi-linear interpolation is used to generate the final prediction samples

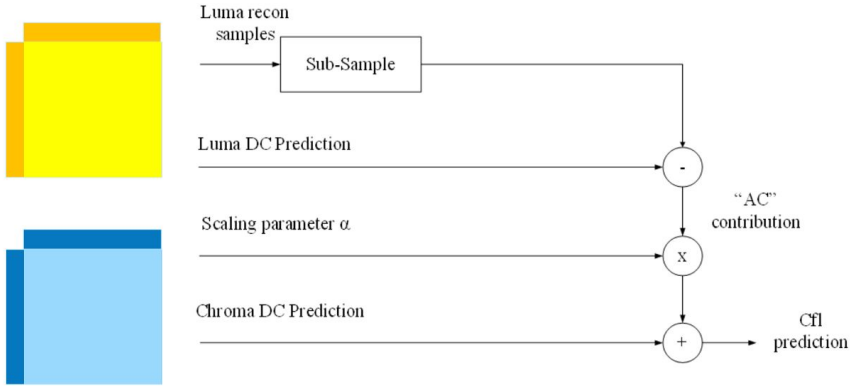






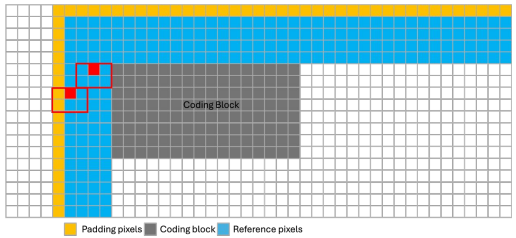
Chroma from Luma (CfL) and MHCCP

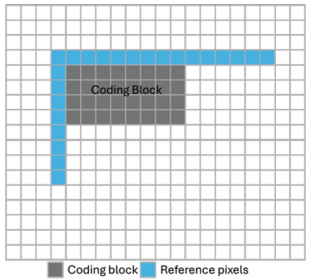
- Improved CfL
  - Luma downsampling filter
    - Three filter types (4-, 5- or 6-tap)
    - Filter selection on sequence level
  - Extra CfL mode with implicitly derived scaling parameter α
    - α = argmin sum(Rec_c – α * Rec_y)^2 over the L-shape region



- Intra: Multi Hypothesis Cross Component Prediction (MHCCP)
  - Three modes are supported in MHCCP, and 3 parameters are derived in each mode.
    - E.g. Top mode:
      chroma = $w_0T + w_1E + w_2F$

E = $(C^2 + F) >> bit\_depth$
F = $1 << (bit\_depth - 1)$

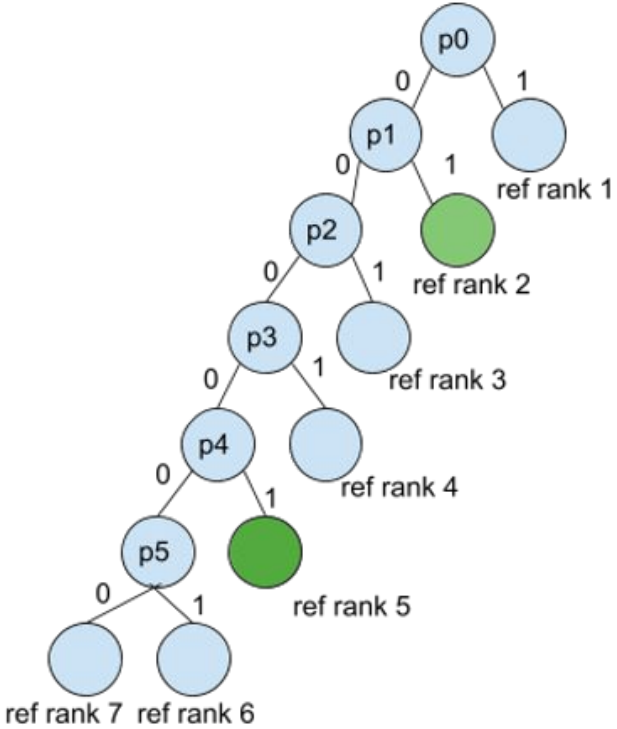




Inter prediction

Reference pictures framework

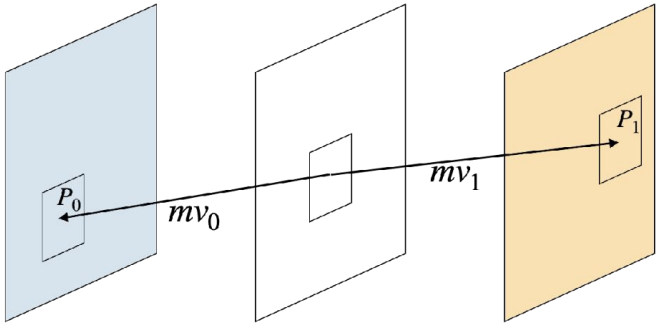
- Ranked Reference Scheme
  - Ranks all 7 references based on:
    - Temporal distance from current frame
    - Quantizer level of each reference
    - Frame resolution (resize mode)
    - Layer ID (multilayer coding)
  - Benefits:
    - Eliminates reference frame mapping signaling cost
    - Orders references by importance achieving more efficient signaling
  - Uses **unary codeword** for reference frame index signaling



E.g.: Compound with refs ranked 2 & 5

- 010001, if num_refs>5
- 01000, if num_refs=5

Compound modes



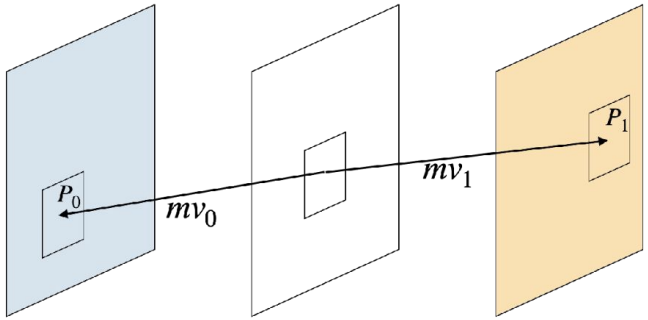
$$P = \frac{w \times P_0 + (16 - w) \times P_1}{16}$$

Equal weights $w = 8$

Compound weighted prediction (CWP): Explicit signal weight

	Weights (w)
Two ref frames are from opposite sides	8, 10, 6, 12, 4
Two ref frames are from same side	8, 12, 4, 20, -4

Compound modes



$$P(x,y) = \frac{m(x,y) \times P_0(x,y) + (64 - m(x,y)) \times P_1(x,y)}{64}$$

Difference Based

- Per pixel difference between the two reference pixels

$$m(x,y) = \begin{cases} 38 + \frac{|p_0(x,y) - p_1(x,y)|}{16}, & sign = 0 \\ 64 - (38 + \frac{|p_0(x,y) - p_1(x,y)|}{16}), & sign = 1 \end{cases}$$

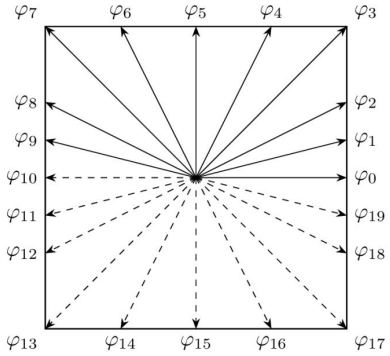
- Sign need to be signaled for the block

Implicit wedge mask

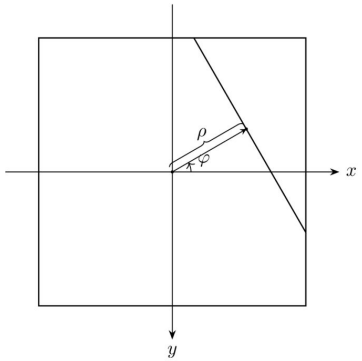
- Implicitly assign weights when combining pixel outside of boundary
- Both pixel inside/outside: m=32
- If one pixel outside:
  - m = 64 for inside pixel
  - m = 0 for outside pixel

Wedge compound mode

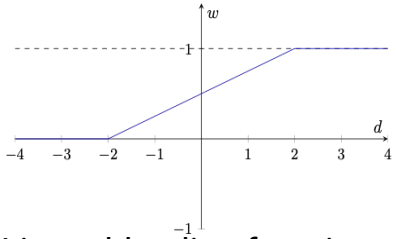
- Explicit wedge mask signaling
  - Wedge mode is extended from 32x32 to 64x64 block
  - Wedge mode is extended from 16 to 68 choices
  - Blending function based on sigmoid function



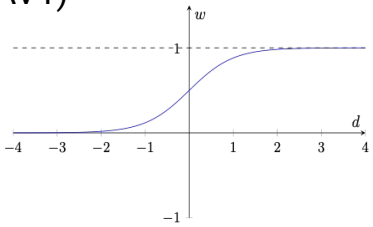
Angles



Distance



Linear blending function (AV1)

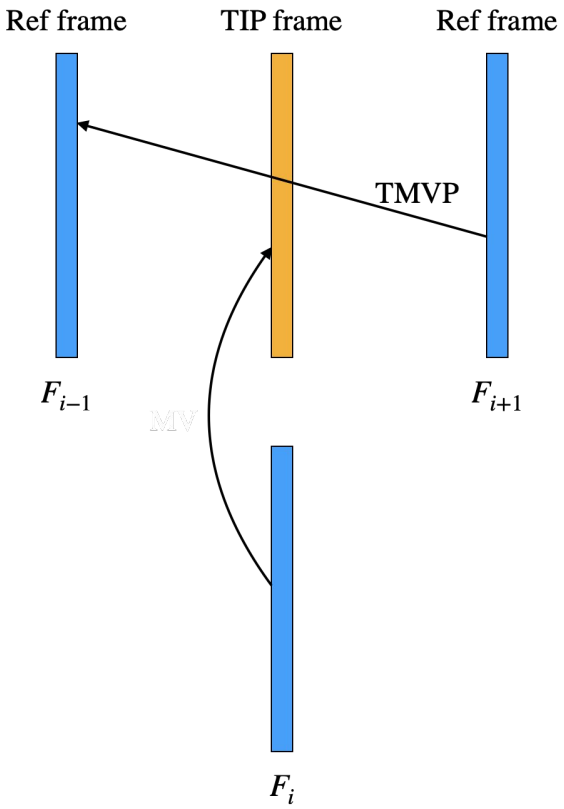


Sigmoid blending function (AV2)

- Different width of blending boundary

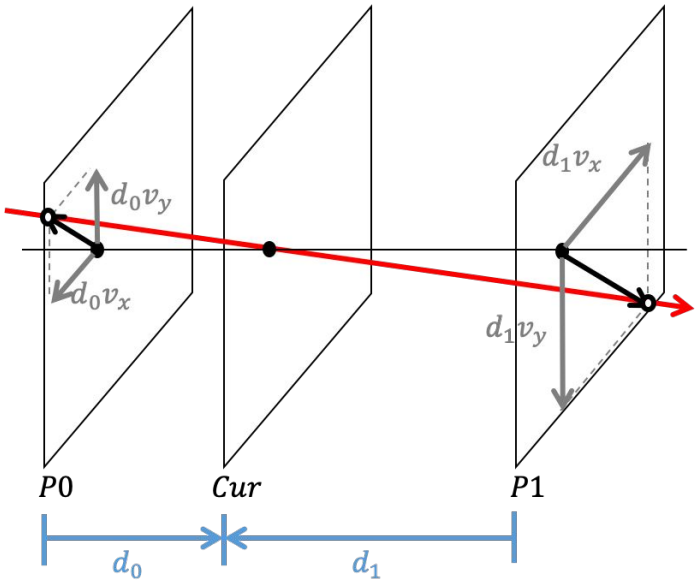
Inter: Temporal Interpolated Prediction (TIP)

- A powerful tool providing substantial gains
- Concept:
  - Interpolates a virtual frame between two reference frames.
  - Reuses existing TMVP motion fields.
- Usage:
  - **Frame level**: Directly output a picture without extra overhead → for low bitrate
  - **Block level**: Used as a single prediction mode → reduces signaling overhead



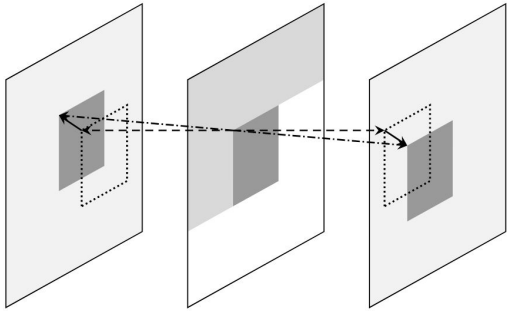
Inter: Optical flow MV refinement (OPFL)

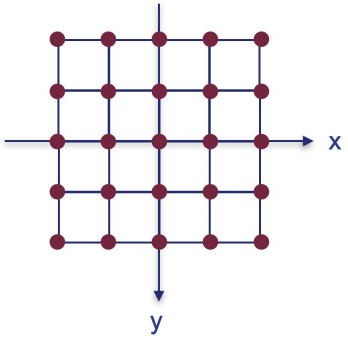
- MVs are refined for each (8x8 or 4x4) subblock in bi-directional prediction block:
  (MVi_x, MVi_y) ← (MVi_x + di * vx, MVi_y + di * vy)
  - P0, P1: Reference blocks after MC.
  - d0, d1: temporal distance
  - Cur: Current source block.
  - vx/vy derived based on optical flow equation:
    - No signaling overhead needed
    - MC applied with refined MVs
  - Applicable to TIP frames



Inter: Sub-block based MV refinement (SMVR)

- Compound mode
- Divide a block into 16x16 sub-blocks
- For each sub-block:
  - Search 5x5 (25 points, one step) region with MV0, MV1 (initial MVs) as center
  - For each offset (ΔMV)
    - Generate first predictor P0 using (MV0 + ΔMV) .
    - Generate second predictor P1 using (MV1 - ΔMV)
    - Compute SAD between P0 and P1
  - Find best offset (ΔMVbest) with minimum SAD



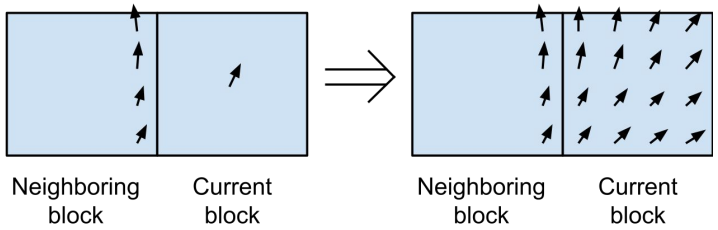


Warp prediction

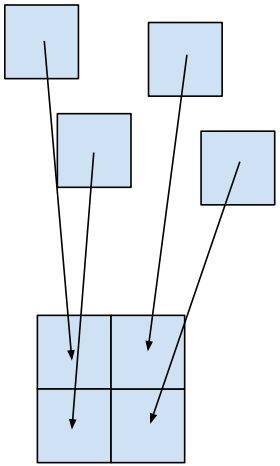
- Non-translation prediction
  - Affine model, up to 6 parameters

$$mv_x = (\text{mat}[2] - 1) * x + \text{mat}[3] * y + \text{mat}[0]$$

$$mv_y = \text{mat}[4] * x + (\text{mat}[5] - 1) * y + \text{mat}[1]$$



- Several warp modes (WARP_CAUSAL, WARP_EXTEND, WARP_MV, WARP_DELTA)
  - Differ in derive vs signal information
- AV1 warp: breaks an affine transformation up into two shears
  - Efficient, but can only handle small amounts of warping
- AV2 warp filter also works for stronger warps
  - Split block into 4x4 units
  - Translate each 4x4 unit independently



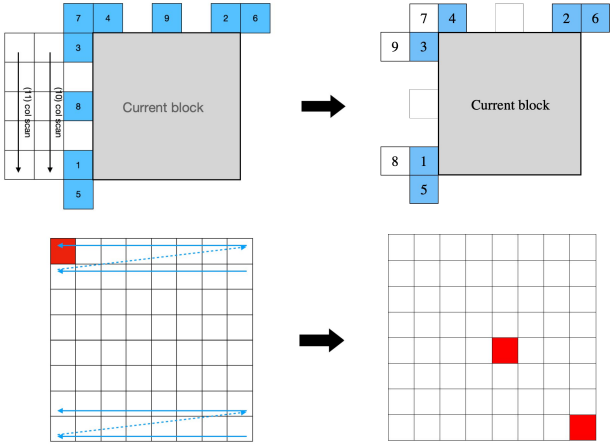
Inter: Block adaptive weighted prediction (BAWP)

- Illuminance compensation in the form
  - **ax + b**
    Parameters ***a*** can be implicitly derived or explicitly signaled, parameter ***b*** is implicitly derived
  - Implicit derivation: minimal SSE between reconstructed and predicted L-shape causal neighborhood
  - Explicit signaling: selection from a set of candidate values of ***a***

$$(a^*, b^*) = \underset{(a,b)}{\operatorname{argmin}} \; SSE\Big(\text{Reconstructed: Current block}, \text{a * Pred() + b: Current block}\Big)$$

Inter: MV prediction

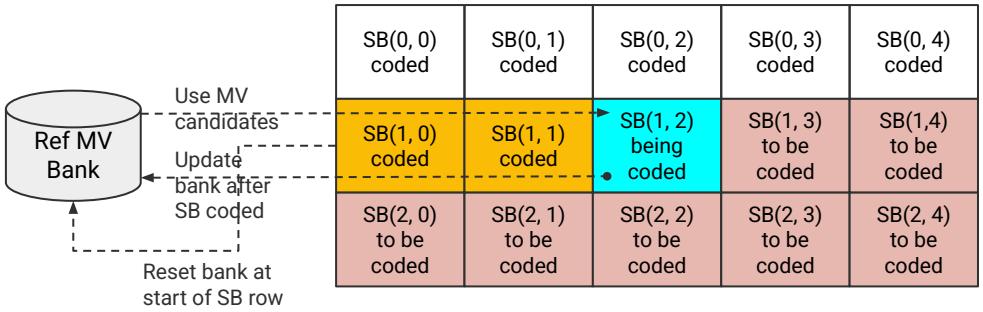
- Dynamic reference list (DRL)
  - List of MV predictiors from spatial (SMVP) and temporal (TMVP) candidates
    - Separate DRL for compound mode
    - Support of skip mode
    - Better optimized search process
      - No sorting
- Warp reference list (WRL)
  - Introduced for WARP_DELTA mode
    - Similar to DRL
  - List of warp parameters obtained from neighbor blocks

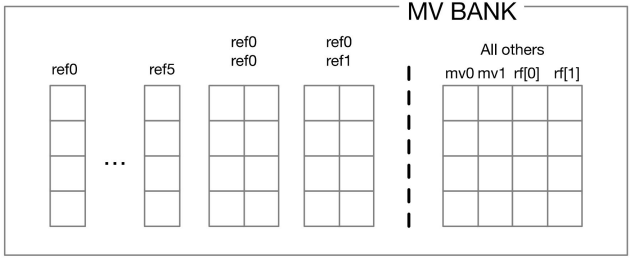


Sampling simplifications for DRL

Inter: Reference MV Bank and warp parameters bank

- Reference MV Bank & Warp Parameter Bank
  - For DRL and WRL
  - MVs / WPs from previously decoded SBs
  - A maximum of 64 MV updates for each super-block
    - Evenly distributed within the SB

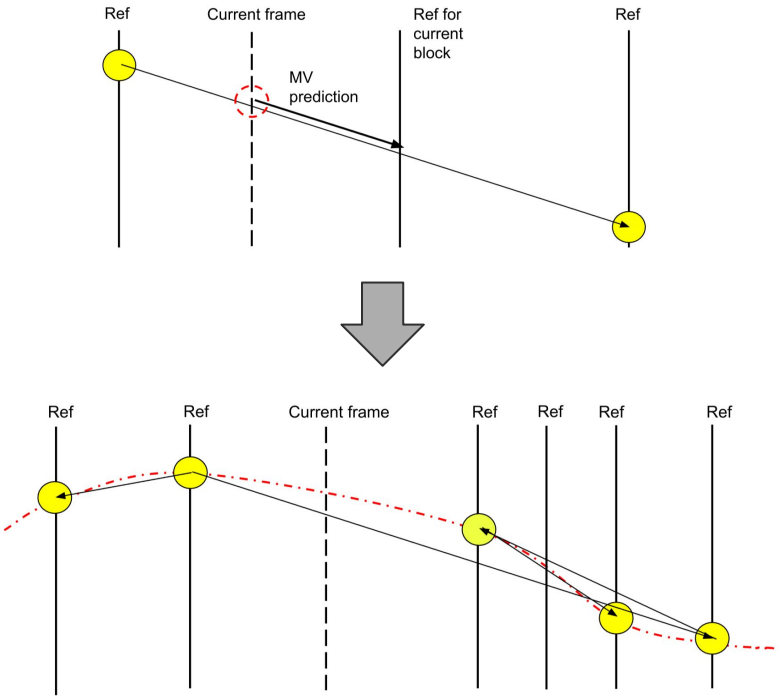




Inter: Motion trajectory tracking

- Trajectory of blocks is tracked by concatenating the decoded MVs in the reference frames.
  - Stored as a two-way mapping table
    - Block ⇔ trajectory ID
  - Improves TMVP and SMVP

- Support non-linear motion
  - e.g. camera shake, objects moving along a curved path



Inter: MV resolution

- Adaptive MV Difference (AMVD)
  - Implicit adaptive MVD resolution
  - Only applies to some inter modes (NEAR_NEW, NEW_NEAR, JMVD_AMVD, AMVD_NEWMV)
  - A look-up table based approach where a set of predefined absolute MVD values are allowed.
    - For MVD magnitude > 1 pel, allow MV magnitudes of 2, 4, 8, ..., 2048 only, leading to reduction in the signaling cost of motion vector difference (down to 8 bits)

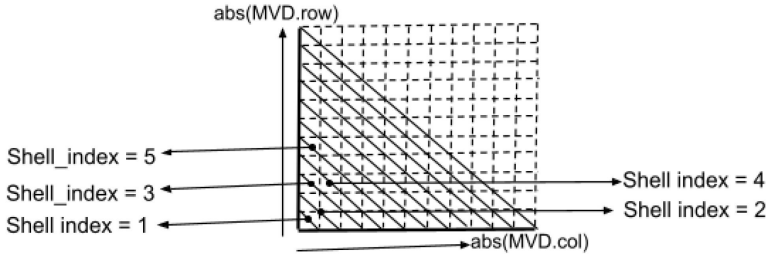
amvd set	index	0	1	2	3	4	5	6	7	8
	row/col in unit of ⅛ pel	0	2	4	6	8	16	32	64	128

- Adaptive MV Resolution (AMVR)
  - Signaled adaptive MVD resolution
  - AMVD precision from {8, 4, 2, 1, ½, ¼, ⅛} pixel selected and signaled.

Inter: MV coding and range

- Joint coding of absolute row and column
  - Decode shell_index
  - Decode absolute value of col

$$shell\_index = |MVD.row| + |MVD.col|$$

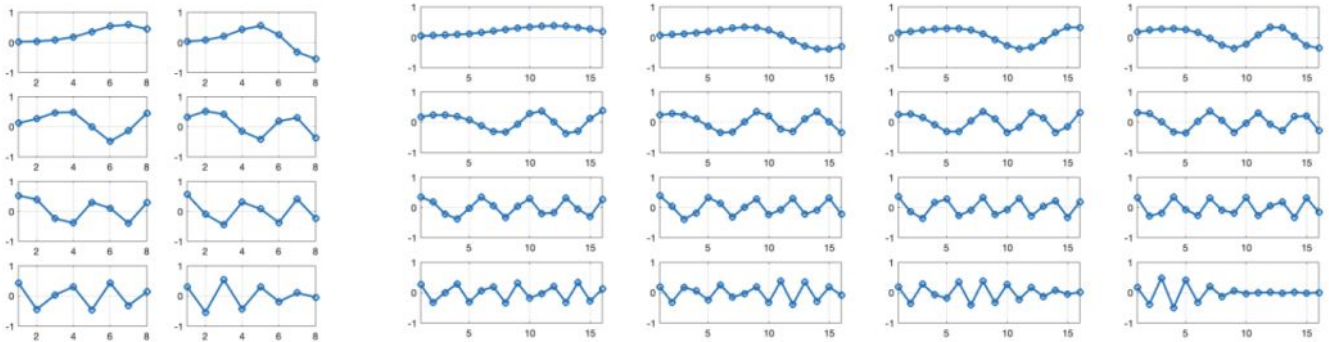


- AV1: 15 bits MV range
  - 12 bits full pixel range from -2^11 to 2^11
- Extend the range to 17 bits in AV2
  - 4x MV range coverage
- Providing benefits for:
  - Videos of larger resolution (4K/8K) and fast motion

Transforms

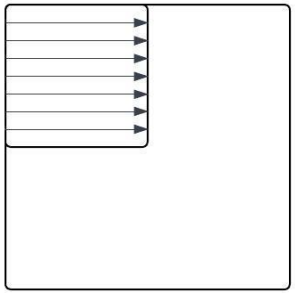
Core primary transforms

- 16 primary transform types, with 4 types (DCT/ADST/flipped-ADST/identity) as 1D transforms
- AV2 improvements
  - Supports a unified matrix framework for DCT-II that can replace AV1's butterflies for lengths 8 and above
  - Kernel improvements on ADST
    - Intra: DST-IV (4-point), learned unitary transform (8-point), or DST-VII (16-point)
    - Inter: DST-IV (4-point), data-driven transform (DDT) for 8-point and 16-point

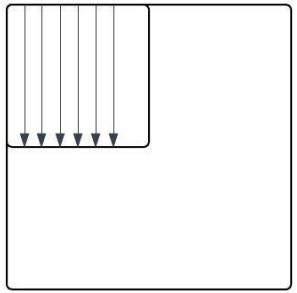


Large transforms

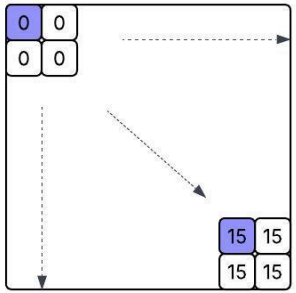
- Large transforms (64x64 or larger):
  - Extends maximum chroma transform size to 64x64
  - 32-point IDCT-II with upsampling



Inverse DCT-II 32p
(Horizontal)



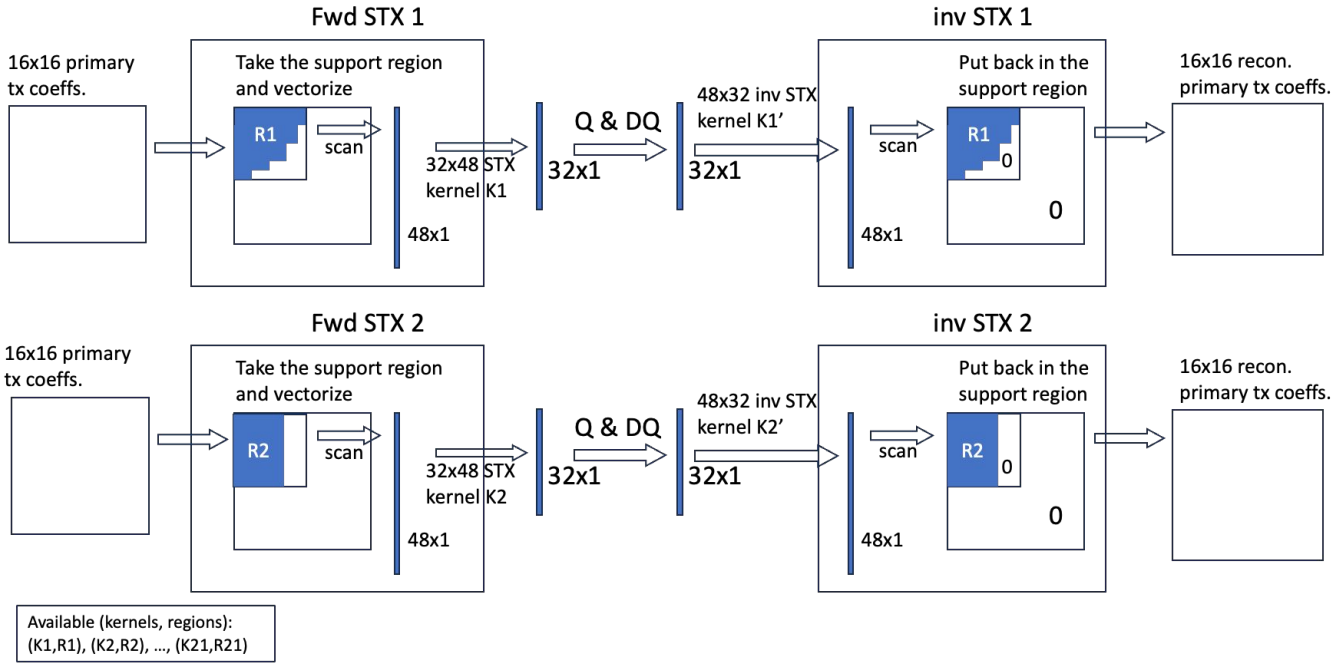
Inverse DCT-II 32p
(Vertical)



Upsampling

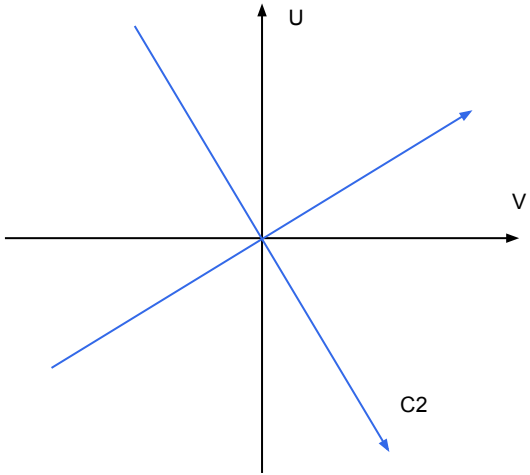
Secondary transforms

- Each secondary transform is applied to “low frequency region” of primary transform coefficients
  - For sub-8x8, top-left 4x4 region is always used (input: 16 coeffs., output: 8 coeffs.)
  - For 8x8 and larger, a kernel-specific shape of 48 coefficients is used (input: 48 coeffs., output: 32 coeffs.)



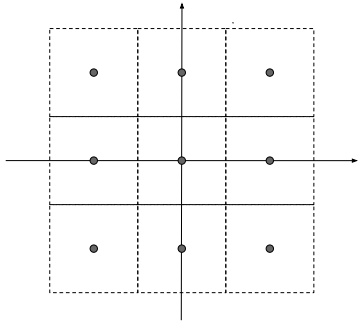
Cross chroma-component transforms

- 2D rotations applied to colocated u and v coefficients. (U,V) → (C1,C2).
  - Same coefficient coding for (C1, C2) as for (U, V)
- 7 rotation angles (0, 30, 45, 60, -30, -45, -60 degrees)
- Decorrelated chroma components



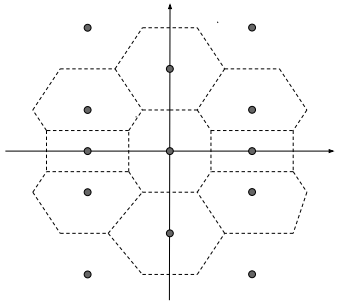
Quantization and Entropy coding

Coefficient quantization



Scalar Quantization

Original method inherited from AV1.



Trellis Coded Quantization (TCQ)

Partition N-dimensional signal space more efficiently.

Parity Hiding (PH)

Hide one bit of information per coding block.

Quantization in AV2

- AV1 Quantization Design
  - q_index (QP) is defined by custom LUTs
  - 256 quantization levels
  - Different tables for 8/10/12-bits and AC/DC
  - Can't reach as low bitrates as HEVC/VVC even with max q_index value.
- AV2 New Quantizer
  - q_index (QP) to quantization step-size mapping given by a unified exponential formula
  - 10-bit, 12-bit quantizers and AC/DC modifiers are shifts on indices
  - Steps double for every 24 q_index
  - 256/304/352 levels for 8/10/12-bit content
  - Range expanded to cover wider range of qualities

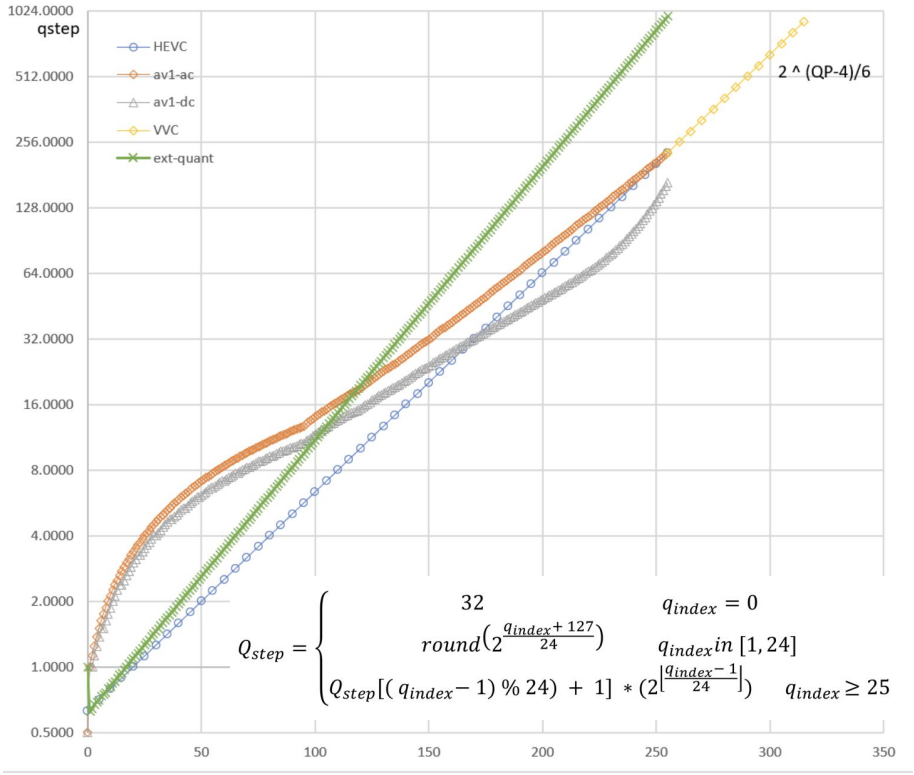


Figure 5. log2(normalized qstep) over HEVC/VVC QP*5, or AV1 q_idx mapping for 8-bit video

[3] R. Lei, D. Mukherjee, M. Krishnan, X. Zhao, S. Liu, J. Boyce, "Proposal on Extended Quantization Design for AV2", AOMedia document CWG-B007.

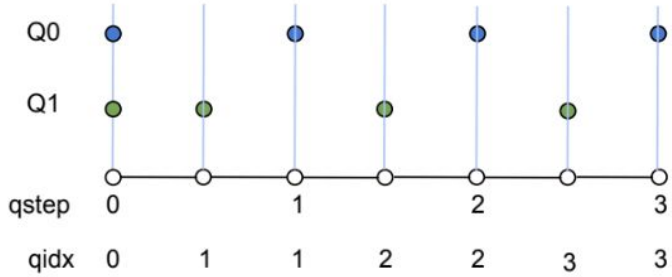
Quantization Matrices in AV2

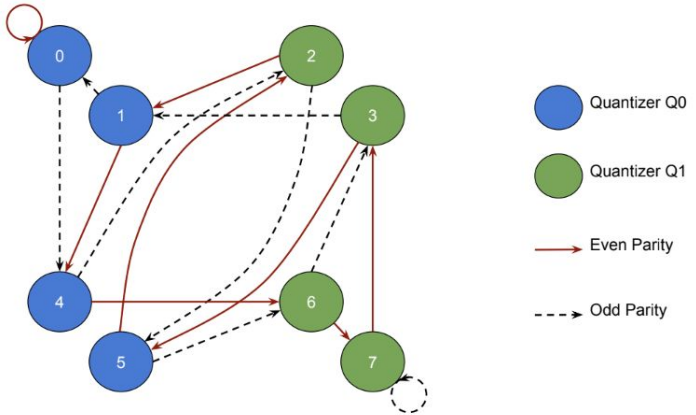
- Quantization Matrix (QM) in AV1
  - 15 set of matrices, each set contains 32x32, 32x16 and 16x32 matrix for luma and chroma. The total size is 100k Bytes.
  - Coefficients are defined as constant LUT.
  - User-defined quantization matrices are not supported.
  - All regions within a frame need to use the same matrices.

- Quantization Matrix in AV2
  - Use a 9 parameter model to derive the fundamental QM
  - Reduce fundamental matrices size to 8x8, 8x4, 4x8 and use upsampling to get the matrices for other block sizes
  - Allow user defined QMs
  - Different segments within a frame can use different QM

Trellis Coded Quantization (TCQ)

- Trellis Coded Quantization (TCQ) is introduced as an alternative to scalar quantization.
- TCQ mode defines two quantizers, labeled as Q0 and Q1.
- Quantizer is controlled by a state machine. After each coefficient is decoded, the next state is determined by the coefficient parity (least significant bit).



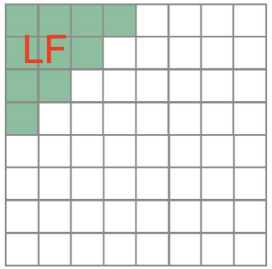


Parity hiding

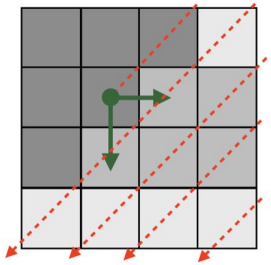
- Parity Hiding is activated when there are 4+ non-zero AC coefficients.
- Luma DC coefficient parity is not signalled and instead derived from the parity of the other coefficients.
- Parity is obtained from the sum of base range (BR) and low range (LR) AC coefficient levels.

Coefficient coding

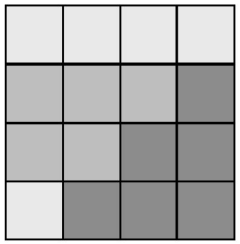
- Region-based coefficient coding
  - applied to non-IDTX transform coefficients (e.g., DCT/ADST)
    - low-frequency (LF) region with sophisticated context modeling
    - unified scanning (diagonal scans)

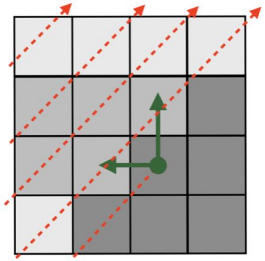


LF region in 8x8


reverse diagonal

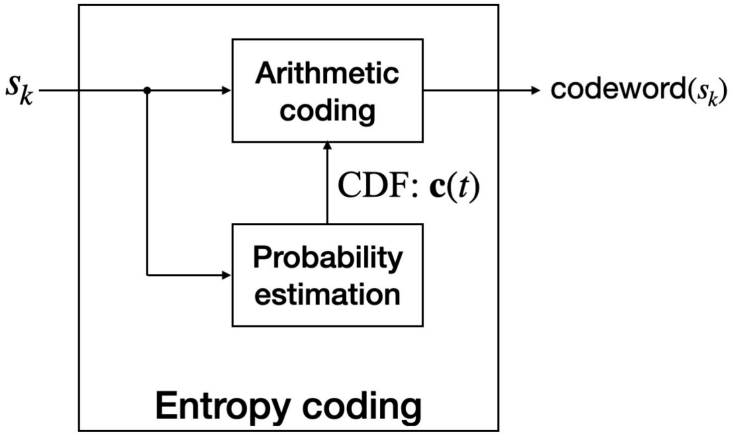
- Forward Skip Coding (FSC)
  - special mode for IDTX transform coefficients
    - intra residuals are larger in bottom-right
    - forward diagonal scan
    - simpler, refined context modeling
  - Benefit: efficient screen content coding with IDTX


a typical IDTX coded block


forward diagonal

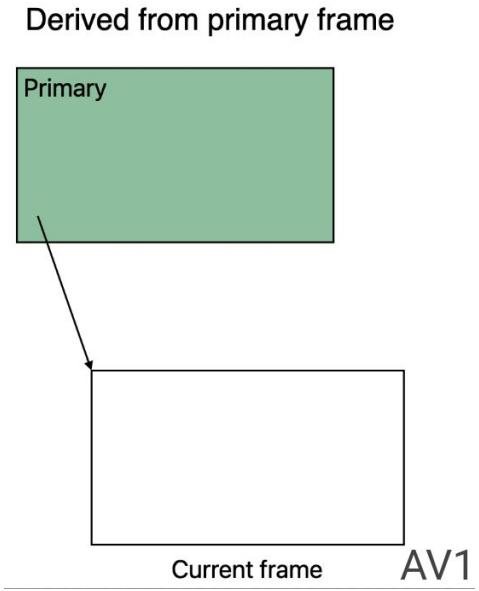
Entropy coding

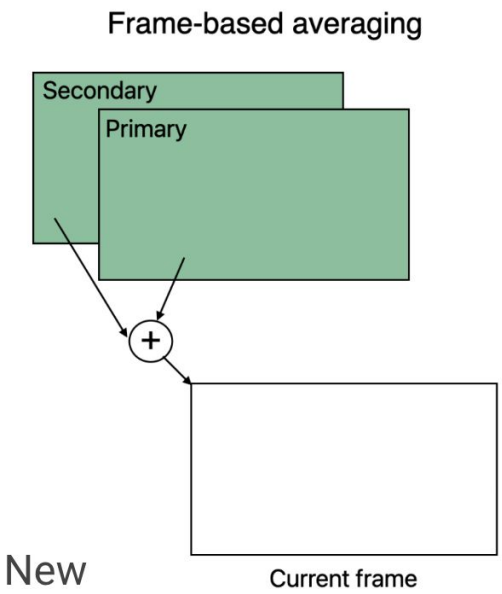
- Arithmetic coding engine from AV1
- Probability Adaptation Rate Adjustment (PARA)
  - More flexibility by changing speed of adaptation
  - Adjusts the adaptation speed
  - Data-driven approach
    - PARA parameters: trained offline*
    - Part of the context initialization tables
- Simplifications:
  - Maximum symbol value is reduced to 8 (was up to 16 in AV1).
  - Bypass coding improvement
  - Context modeling simplifications.

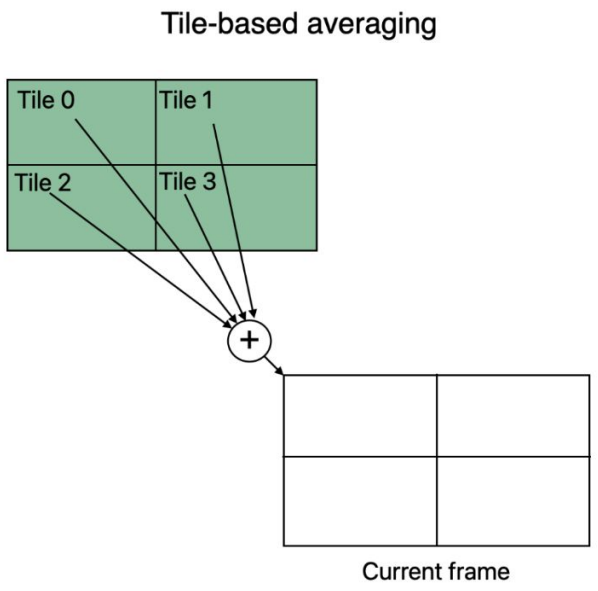


Probability initialization

- AV1 allows copying probability values from a previous frame
  - copied from primary frame (from it's largest tile)
- New: more flexible probability initialization by averaging probabilities

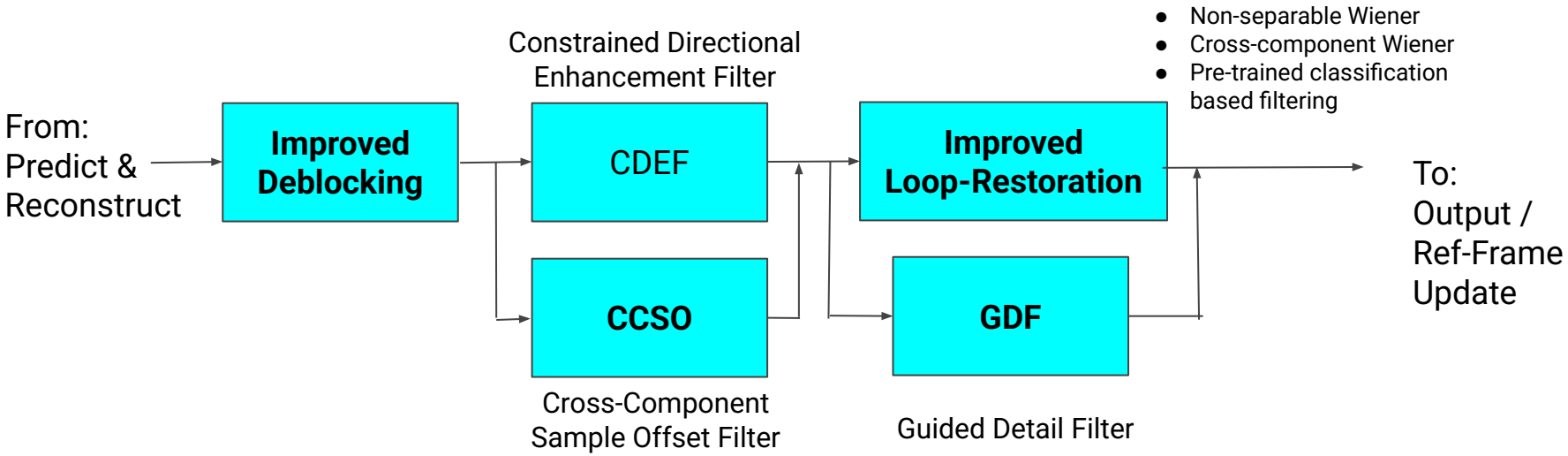






In-loop filters

AV2 loop filtering pipeline



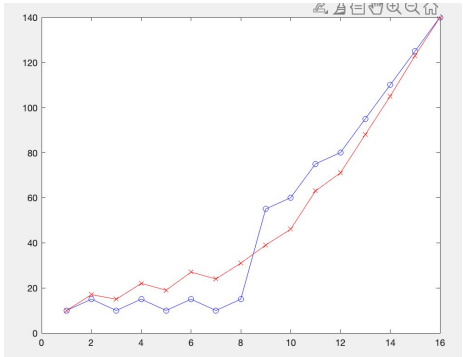
Super-resolution mode:

- Removed

Generalized deblocking filter

- Key design principles for AV2 deblocking
  - Multiple filters are replaced with one filter that can operate at different lengths
  - Filter preserving details allows to apply deblocking more aggressively without too much blurriness
  - Maximum length of filtering is increased to accommodate larger block sizes

- Number of samples modified at each side of block boundary
  - AV1: 1 to 6 for luma, 1 or 2 for chroma
  - AV2: 1 to 8 for luma, 1 to 4 for chroma



[4] A. Norkin, "Deblocking filter for AVM", AOMedia document CWG-C014.

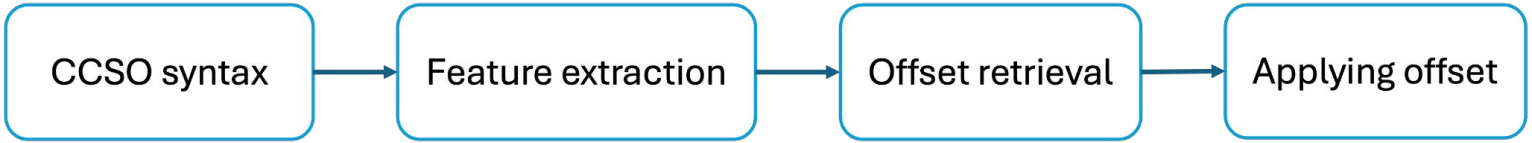
Generalized deblocking filter (results)

- Objective results (original proposal)
  - There were further improvements
- Subjective examples

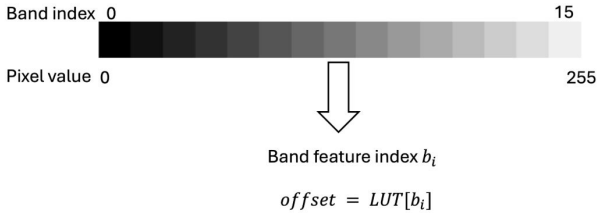
Config	PSNR-Y	PSNR-U	PSNR-V	PSNR-YUV
AI	-0.15%	-0.60%	-0.63%	-0.19%
RA	-0.86%	-1.41%	-1.31%	-0.90%
LD	-1.05%	-1.16%	-1.59%	-1.07%

Cross-component sample offset (CCSO)

- CCSO
  - uses ***luma*** pixels to derive ***offsets*** and refine both ***luma*** and ***chroma*** reconstruction pixels



- Two modes
  - CCSO: band feature mode



  - CCSO: Edge + band feature mode
    - Gradient computation

Luma Channel

5	1	0	3	6
4	2	rl	2	4
6	3	0	1	5

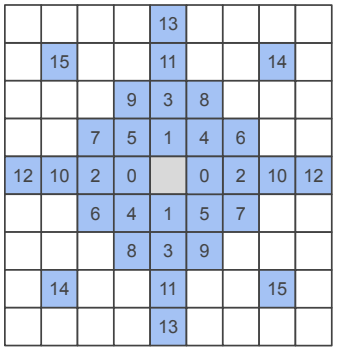
Luma/Chroma Channel

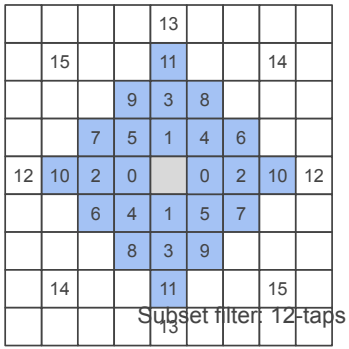
		rc		

Loop restoration (LR)

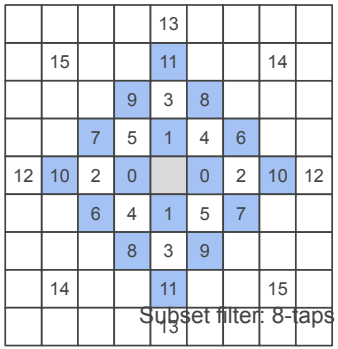
- Frame divided into Restoration Units (RU)
  - RU sizes can be 512x512, 256x256, 128x128 or 64x64
- LR filters in AV2:
  - Pixel-classified non-separable Wiener filter, explicitly signaled at frame level
  - Non-separable Wiener - explicitly signaled, unclassified
    - Cross-component for chroma
  - Pixel-classified non-separable Wiener filter - pretrained
    - Luma only

Explicitly-signaled Wiener filter shapes

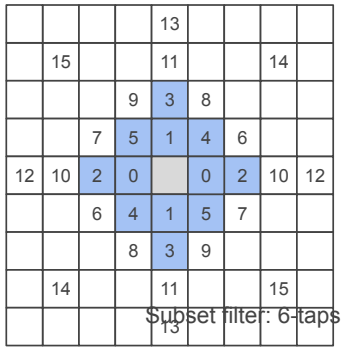




Subset filter: 12-taps



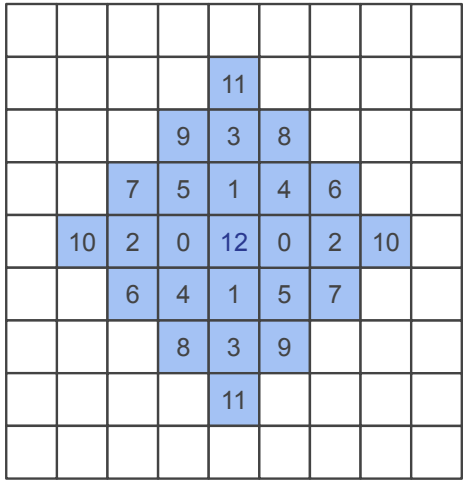
Subset filter: 8-taps



Subset filter: 6-taps

Loop restoration (LR)

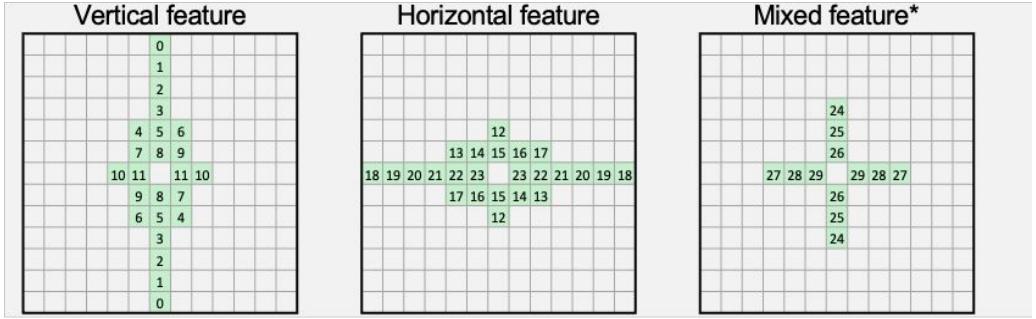
- Pre-trained non-separable pixel-classified Wiener filters (luma)
- 4 sets of filters are pre-trained
- For every 4x4 block:
  - Classify into one of 64 classes
    - Based on edge gradients, quantization step, and whether transform is skipped
  - Apply a 13-tap pre-trained filter based on classification index
- No signaling of coefficients needed
  - Low overhead



PC Wiener filter:
13-taps

Pre-trained Pixel-Classified Wiener - Luma

Guided detail filter (GDF)



- Multichannel filter:
  - Extract features representing local neighborhood of to-be-filtered sample
- Quantizer
  - Convert features to table index
- Sample rectifier:
  - Lookup expected error in table
  - Scale the expected error
  - Compensate scaled error to the sample

AV2 features for special use cases

Screen content coding

- Palette mode (AV1/AV2)
- Intra block copy (Intra-BC):
  - Global Intra-BC (AV1/AV2)
  - Local Intra-BC (AV2)
- Transform skip/FSC (AV2)

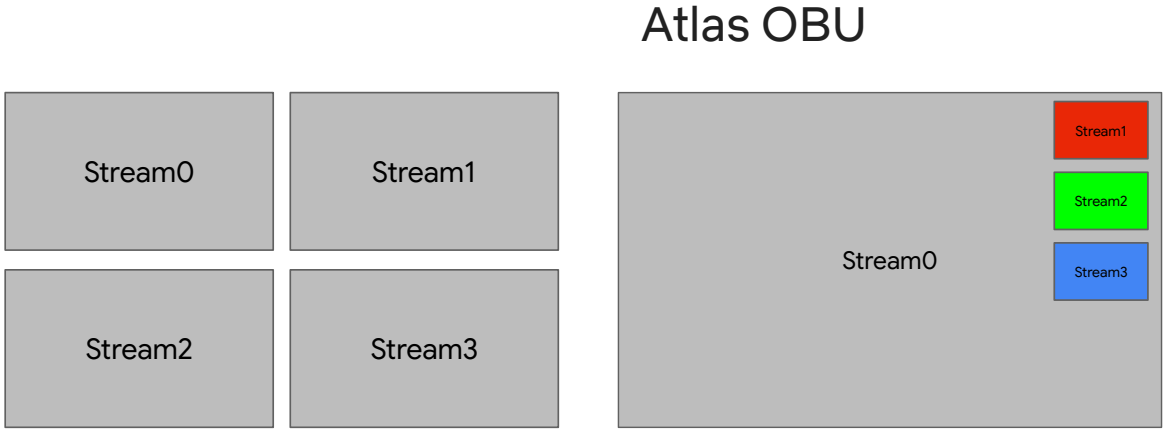
Intra block copy (IBC)

- Motion compensation using the reconstructed samples in the same frame as the reference

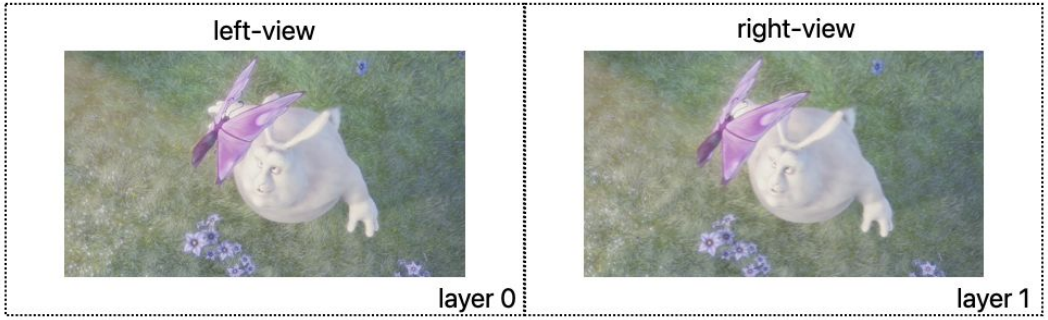
Feature	AV1	AV2
Global search	Y	Y
Local search	N	**Y**
Luma block vector precision	Integer	**Quarter pel, bilinear interpolation**
Chroma block vector precision	Fractional	Fractional
In-loop filters when global Intra-BC enabled	**N**	**Y**
Block Adaptive Weighted Prediction	N	**Y**

Multi-layer and multi-stream video

- Mosaic video
  - N videos shown in 1 screen

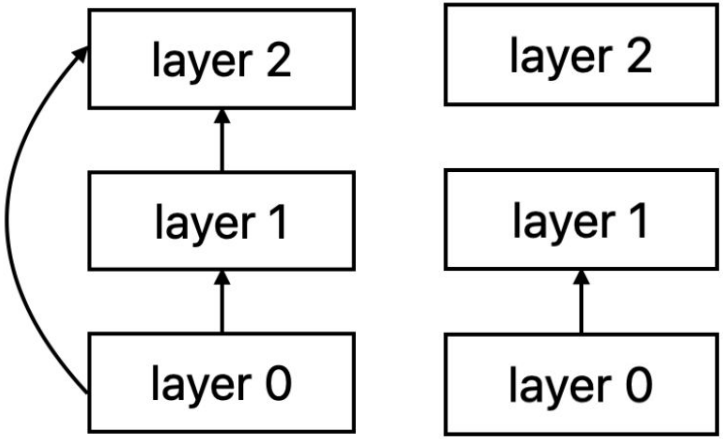


- Immersive video support
  - Stereo video support

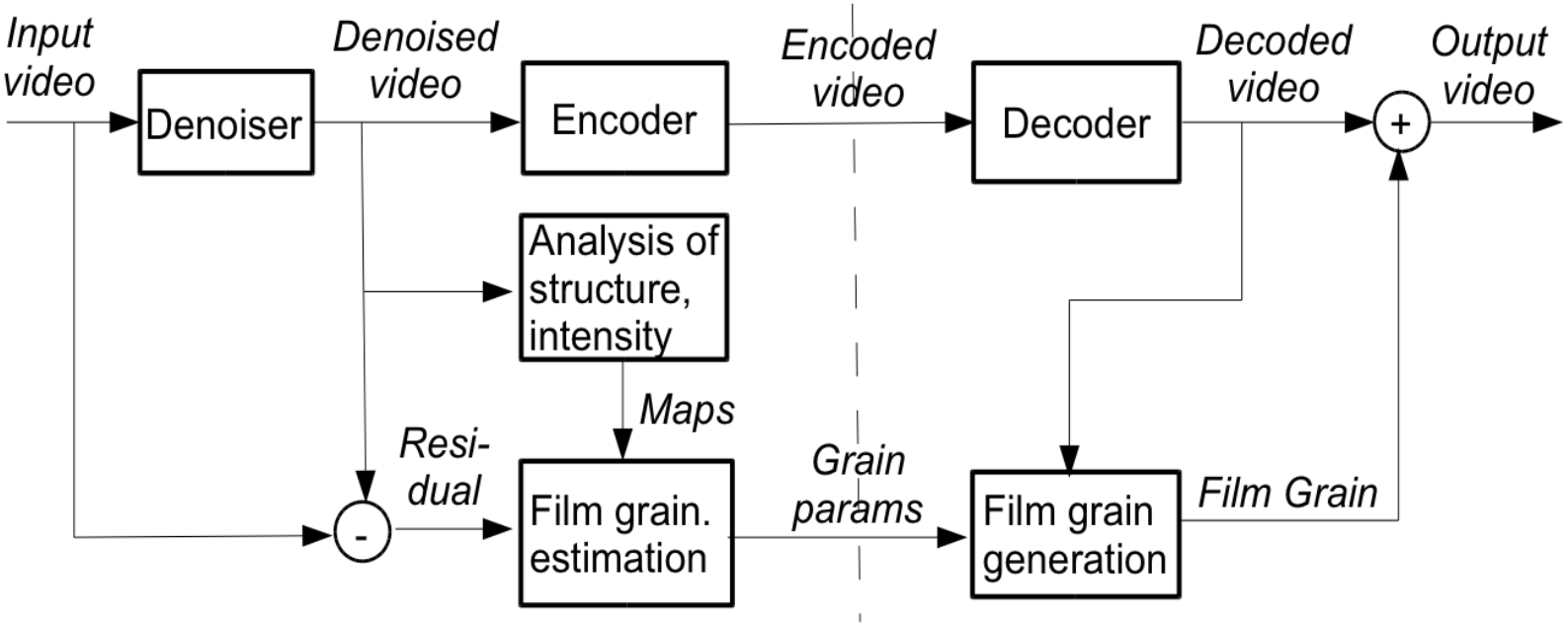


Multi-layer and multi-stream video

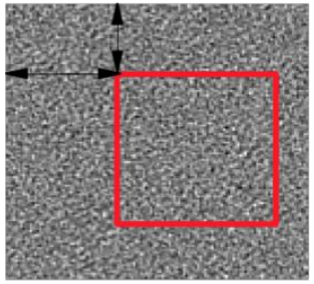
- Enhanced multi-layered coding design
  - Allows more layers (8 embedded and 31 extended)
    - Embedded allow prediction between them
  - Configurable DPB with up to 16 reference frames
  - Scalability support
  - Layer-dependency descriptors (allows sparser dependencies)



Film grain synthesis

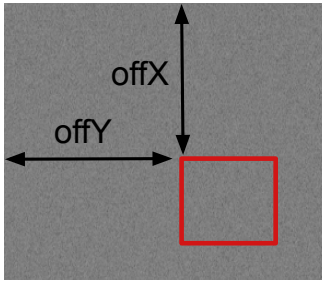


Film grain synthesis



64x64

32x32 block



64x64
template
16x16 block

- AV1 film grain synthesis uses 64x64 template with 32x32 block size
- AV2 film grain synthesis has that option too. In addition, it allows to apply film grain in 16x16 blocks, with the same 64x64 template
- Using 16x16 or 32x32 blocks can be selected on a frame basis
- The option for the block size is signaled with the FGS parameter set
- Some changes to the offset derivation to make them more random

Conslusions

Conclusions

- Development of coding tools for AV2 is almost complete, after ~5 years of development
  - Low level tools finalized
  - A few pending items related to High Level Syntax (HLS) design remain
  - Specification writing is ongoing
- Final AV2 Release: ~End of 2025
- Status
  - AVM-11.0: BD-rate gain over AV1
    - -28.63% (YUV-PSNR 14:1:1) and -32.59% (VMAF)
  - All tools in AVM-11.0 rigorously vetted for decoder hardware complexity
  - Work on AV2 verification test is ongoing
- Future work
  - Optimizations for decoder and encoder SW complexity reduction
  - Encoder side improvements and encoder tools for visual quality optimization
  - Further AV2 extensions, including high bit depth (12 and above) profiles and perhaps an AI/ML extension

Thank you!

anorkin@netflix.com