A Technical Overview of **AV1** Video Codec

Jim Bankoski, **Google**

Outline

AOMedia and AV1

Coding Techniques

Coding Performance

What’s Next

Q & A

ALLIANCE FOR OPEN MEDIA

About | Get Started | Membership | News | Contact

Coming Soon to a Screen Near You

amazon
Adobe
Chips&Media™
AMD
BBC
ARM
mozilla
ARGON DESIGN
微帧 Visionular
BITMOVIN
cisco
vimeo
Amlogic
facebook
hulu
IBM
Google
SIGMA DESIGNS
REALTEK
NGCODEC
Microsoft
ALLEGRO
ateme
XILINX®
Ittiam
CableLabs®
BROADCOM®
VeriSilicon
Videolan
NVIDIA.
socionext™
Vidyo®
iQIYI 爱奇艺
NETFLIX
Polycom®
Alibaba Group
intel®

Outline

AOMedia and AV1

Coding Techniques

Coding Performance

What's Next

Q & A

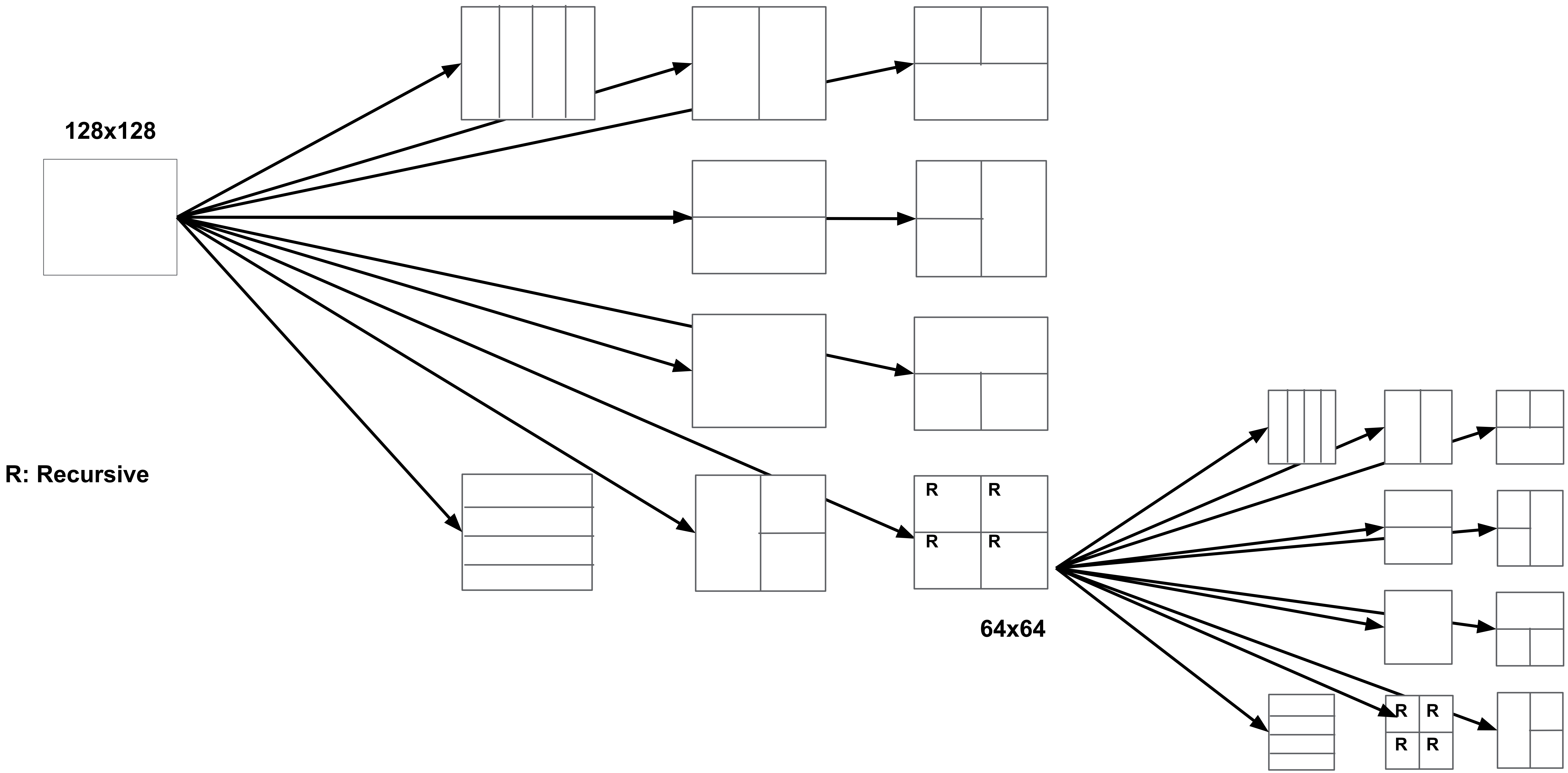
Video coding at a glance

Partition *Predict* *Transform* *Quantize* *Reconstruct* *Encode*

Video coding at a glance

Partition *Predict* *Transform* *Quantize* *Reconstruct* *Encode*

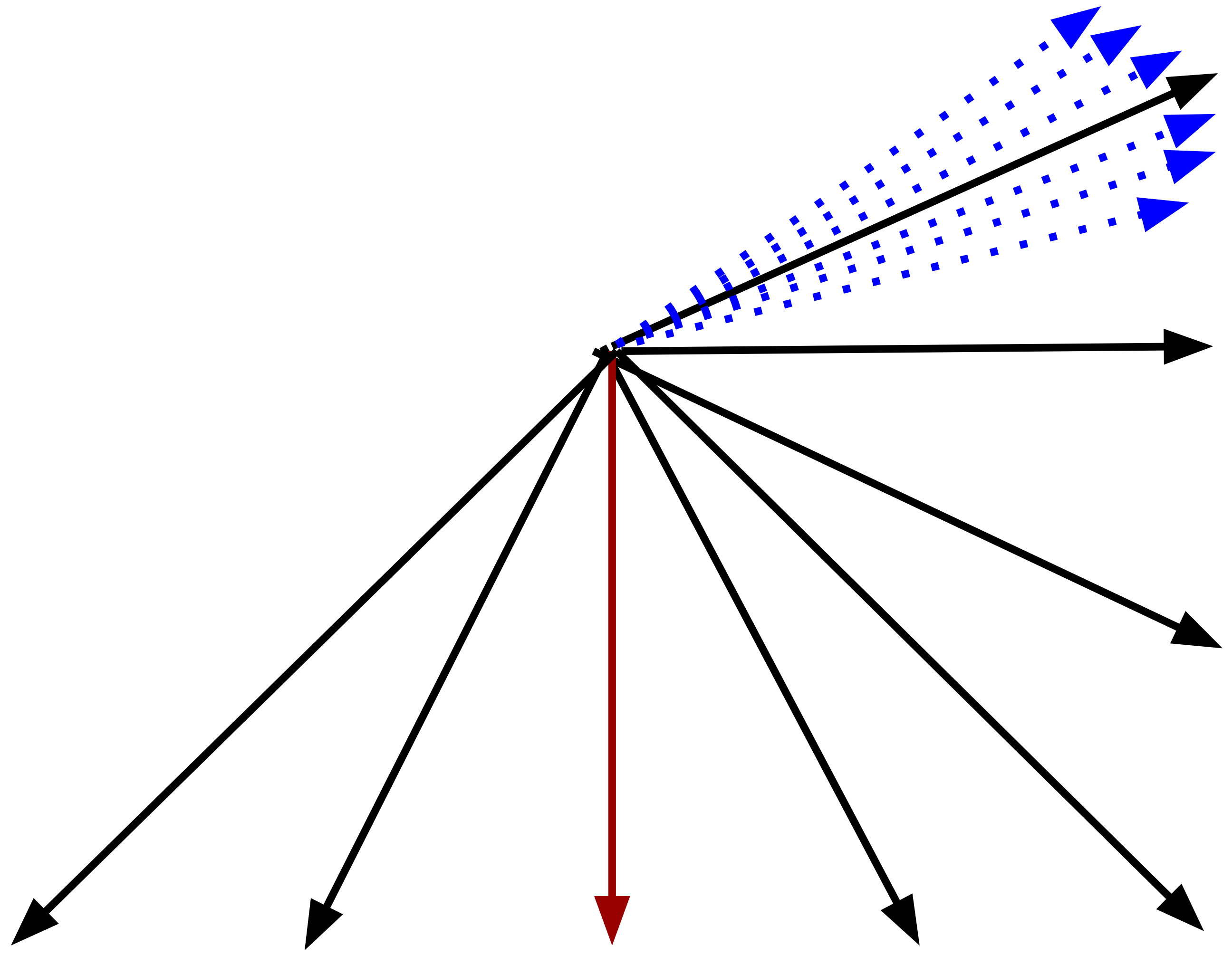
Coding Block Partition



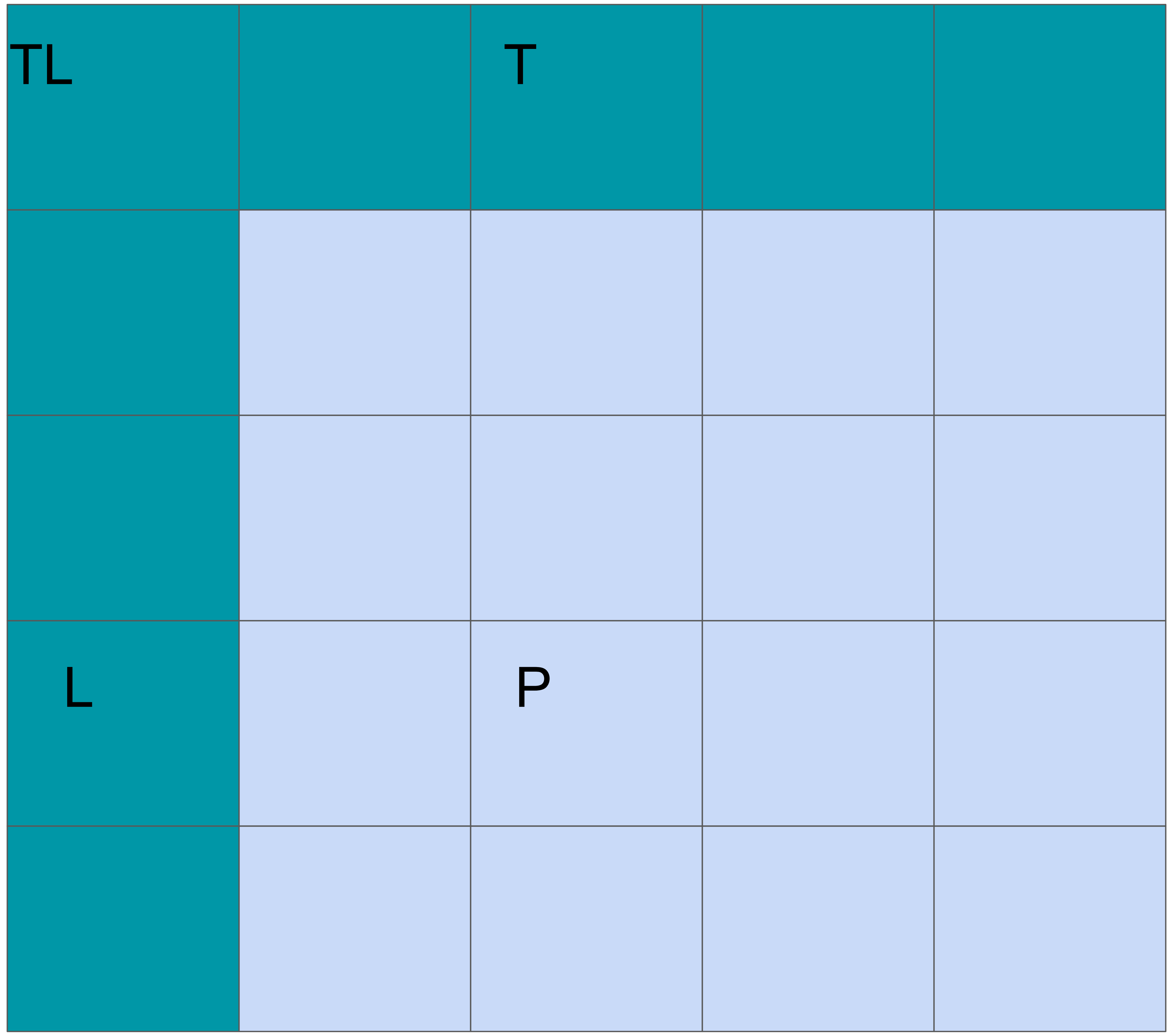
Video coding at a glance

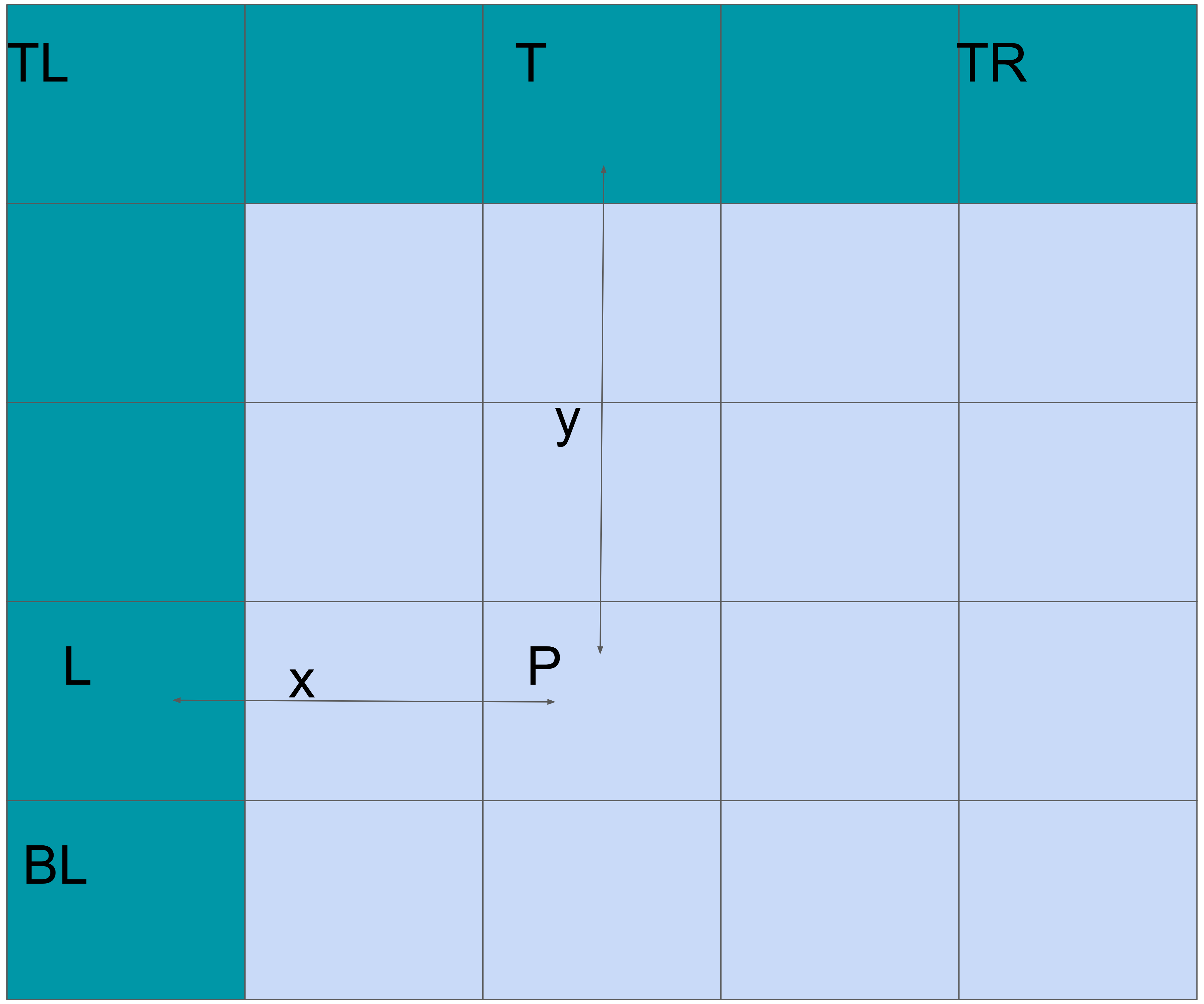
Partition → ***Predict*** → *Transform* → *Quantize* → *Reconstruct* → *Encode*

Extended Directional Intra Modes



	A	B	C	D
E	A	B	C	D
F	A	B	C	D
G	A	B	C	D
H	A	B	C	D





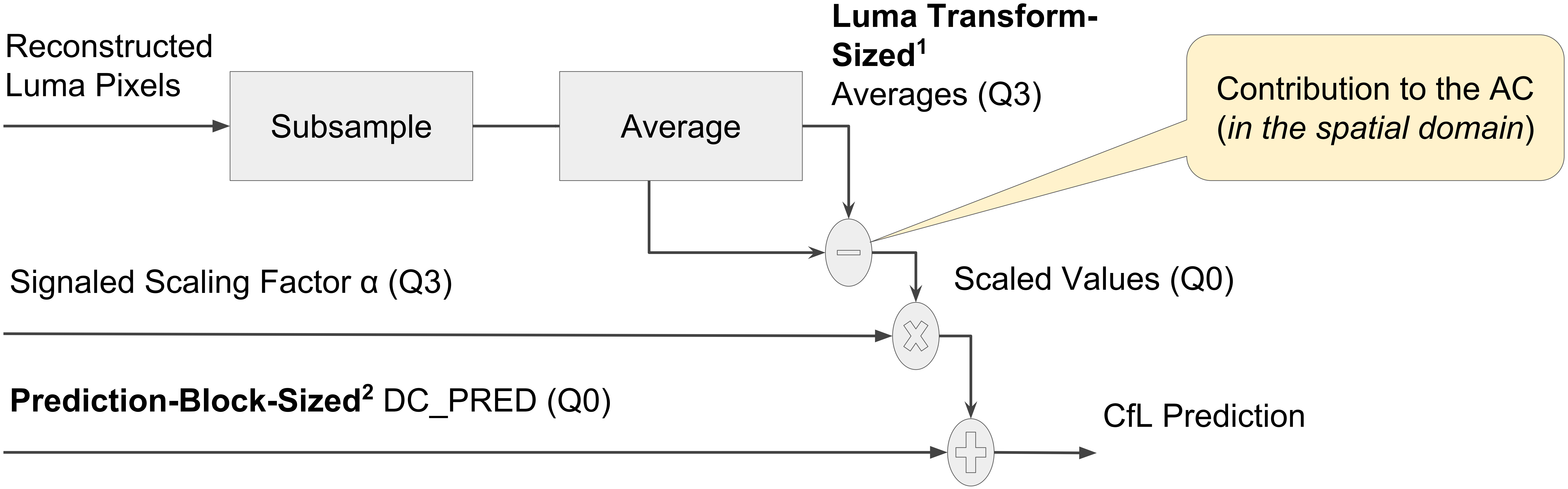
Paeth Mode:
P_{Paeth} = argmin |x - T+L-TL|, over x ∈ {L, T, TL}

SMOOTH_H: $P_{SMOOTH\_H} = w(x)\ L + (1-w(x))\ TR$

SMOOTH_V: $P_{SMOOTH\_V} = w(y)\ T + (1-w(y))\ BL$

SMOOTH: $P_{SMOOTH} = \frac{1}{2}\ (P_{SMOOTH\_H} + P_{SMOOTH\_V})$

Chroma from Luma Prediction

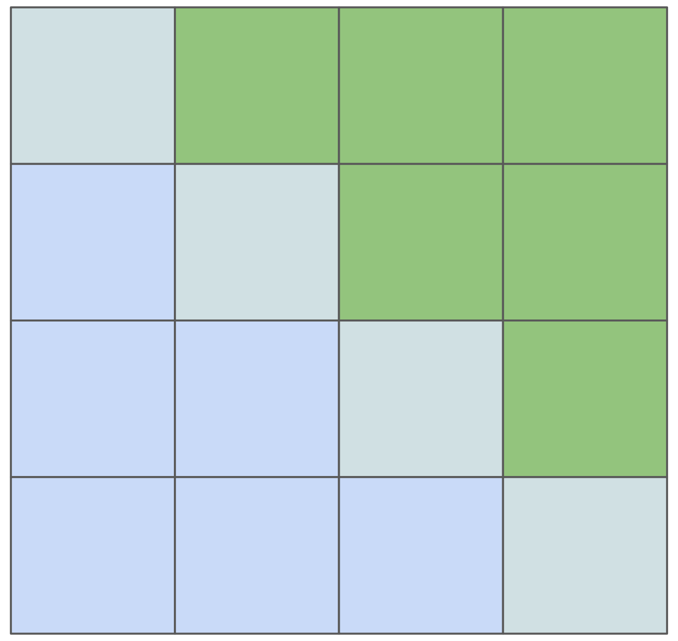


α_{Cb}, α_{Cr} signaled in bit-stream

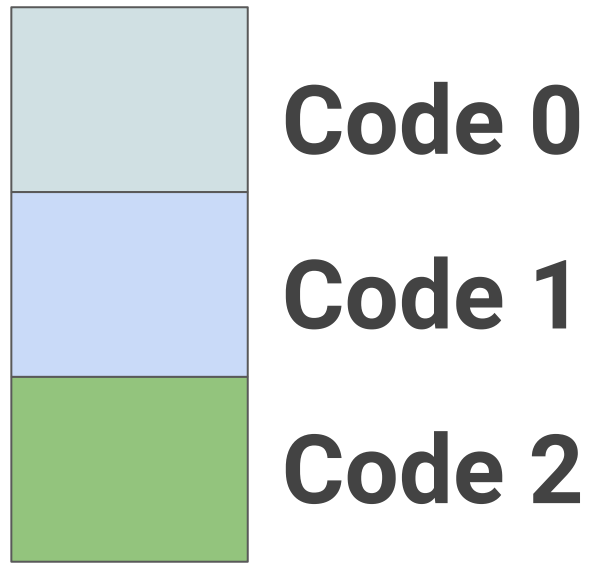
1. Luma average computed over the luma transform block
2. Chroma DC_PRED computed over prediction block

Palette Mode

Pixels



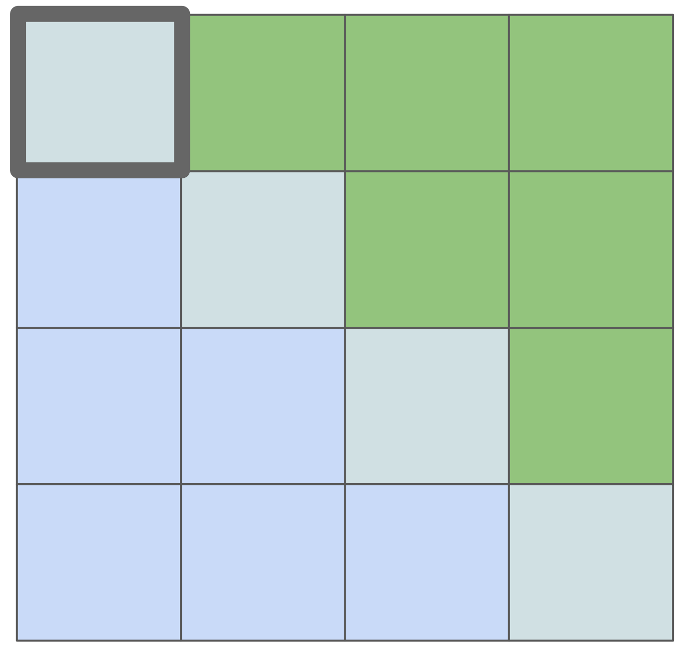
Palette



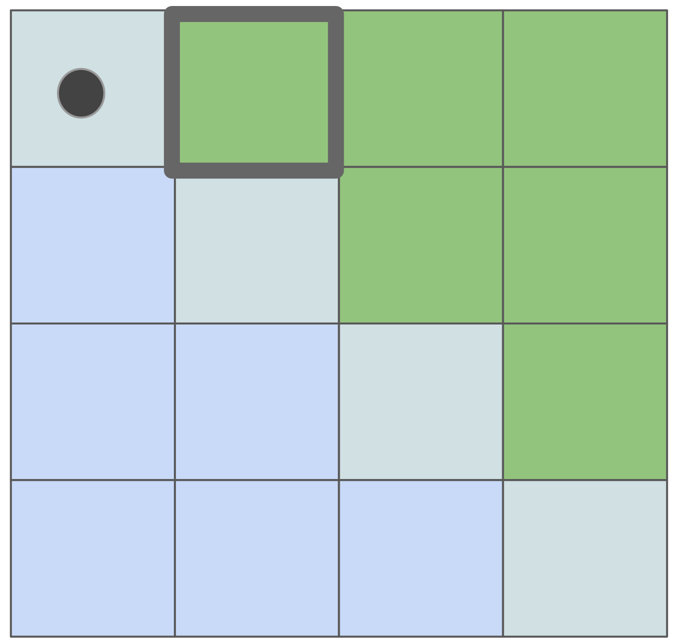
Wavefront Order

0	1	3	6
2	4	7	10
5	8	11	13
9	12	14	15

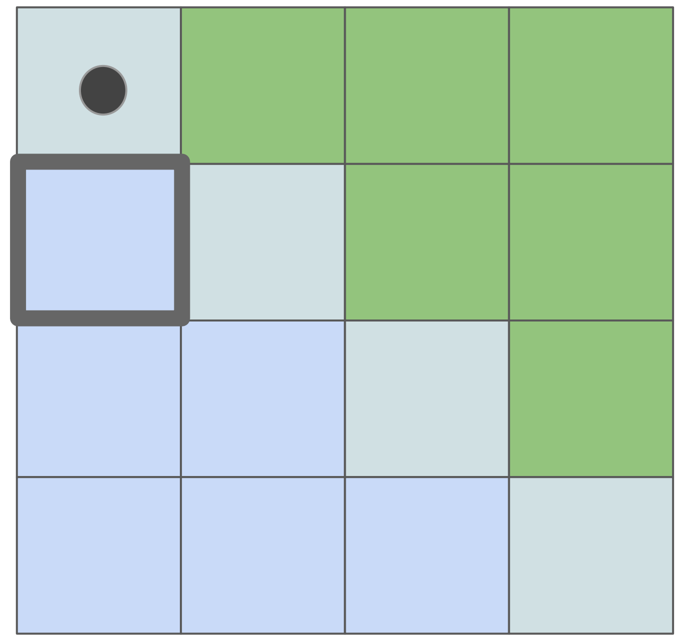
Encoding process proceeds in wavefront order



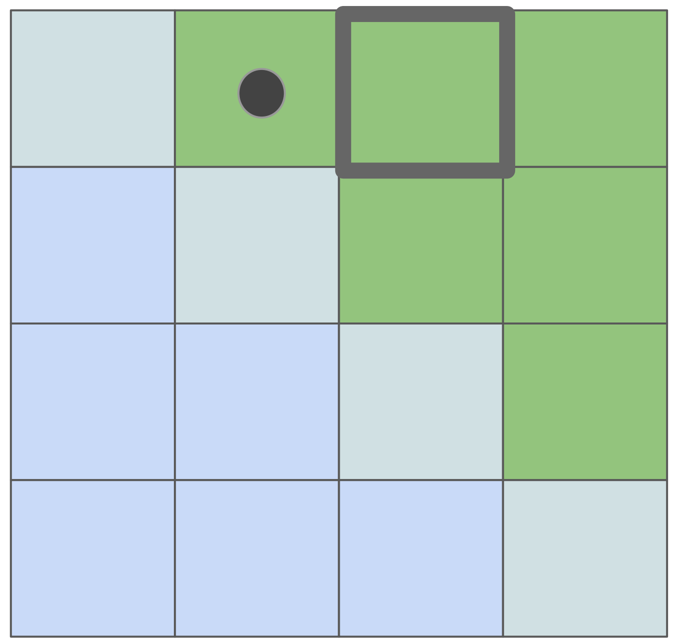
Code 0



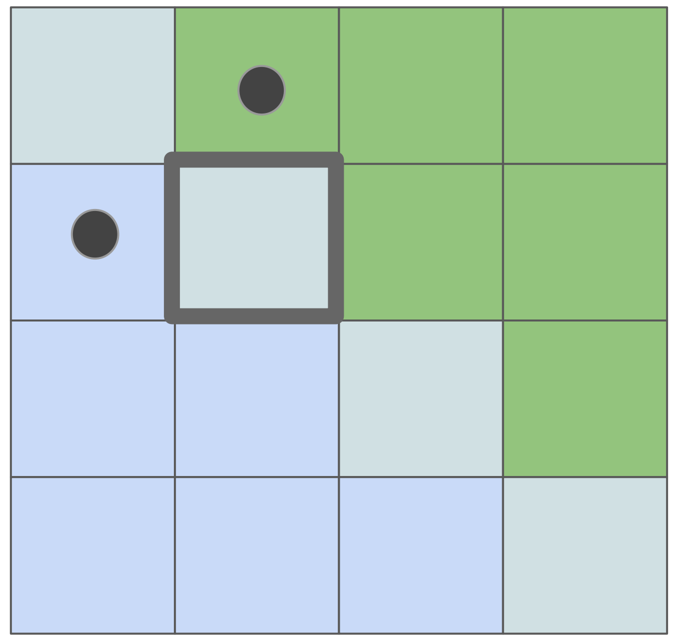
Code 2 using left value as context



Code 1 using above value as context

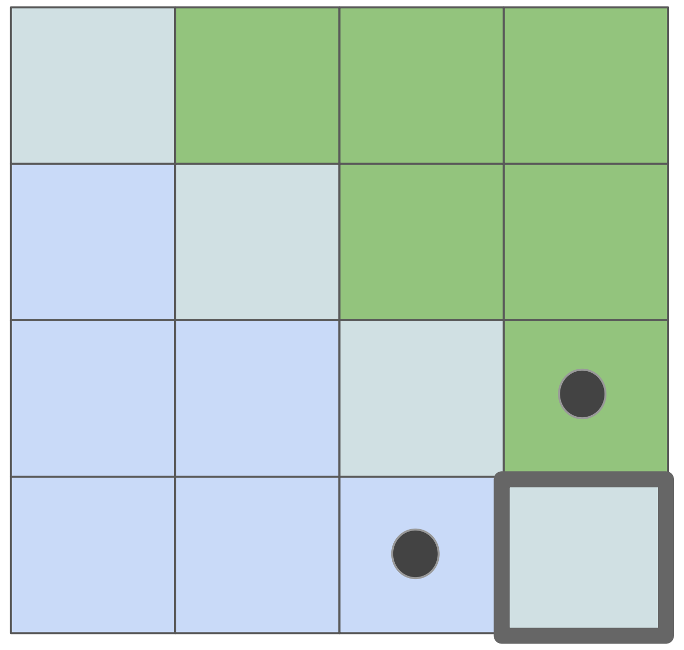


Code 2 using left value as context



Code 0 using left and above as context

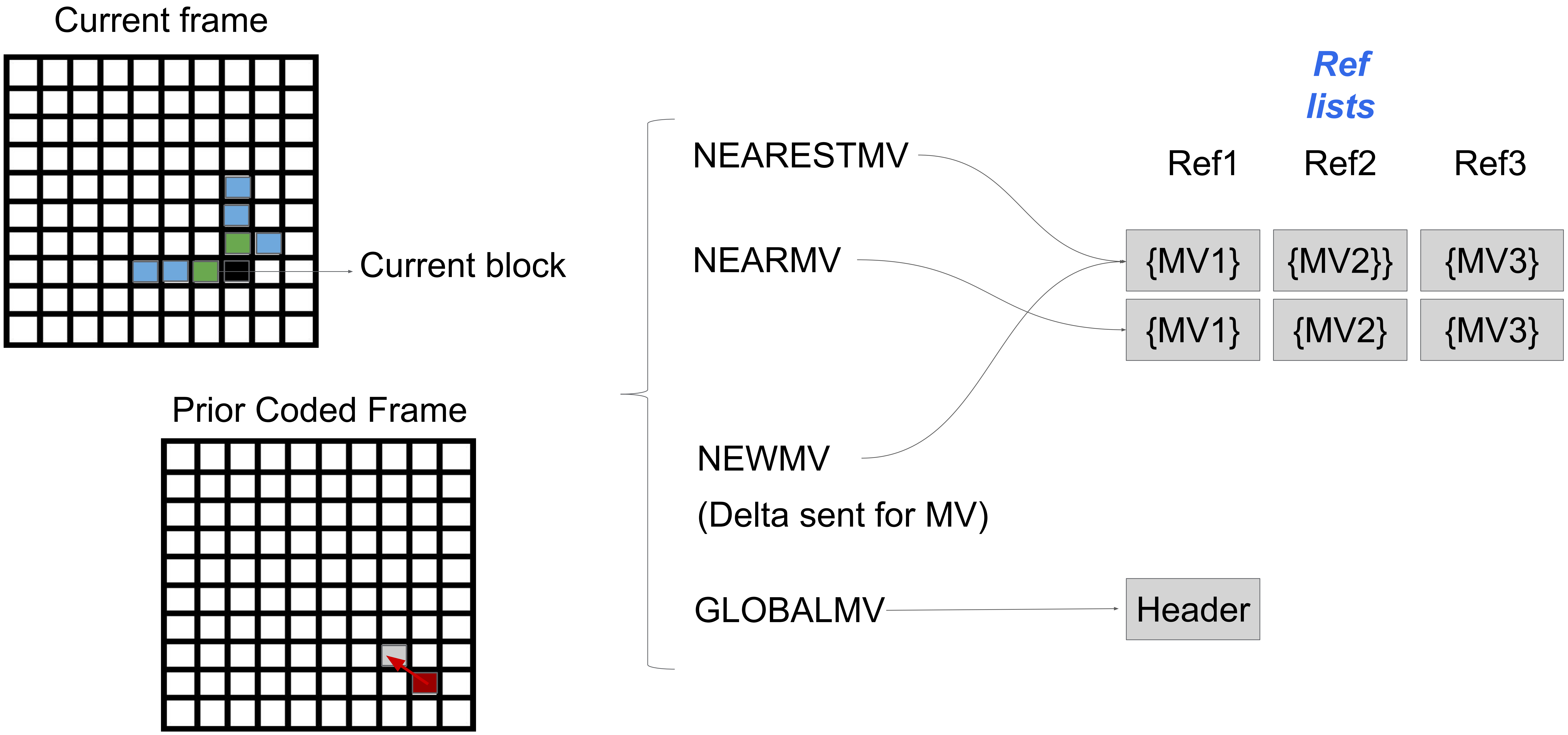
...



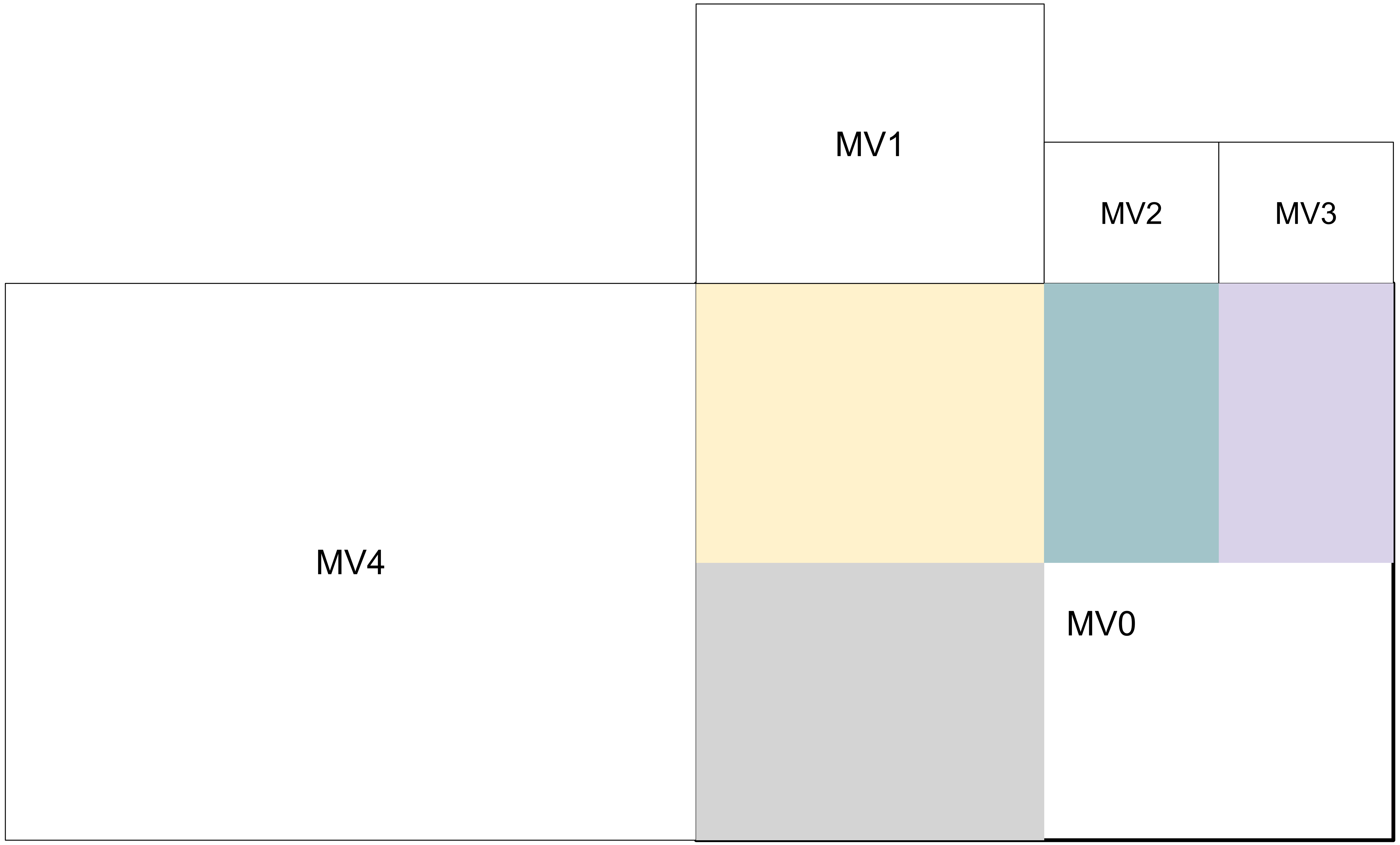
Code 0 using left and above as context

Intra Block Copy

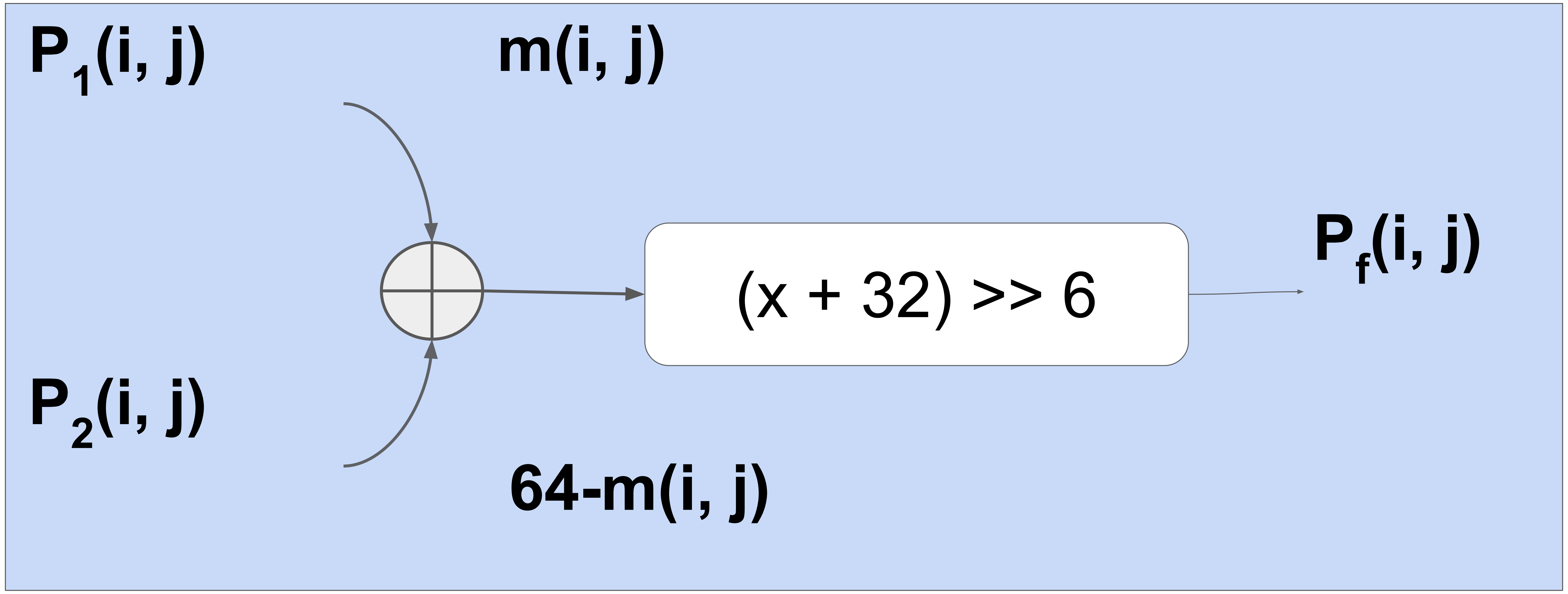
Dynamic Motion Vector Referencing



Overlapped Block Motion Compensation

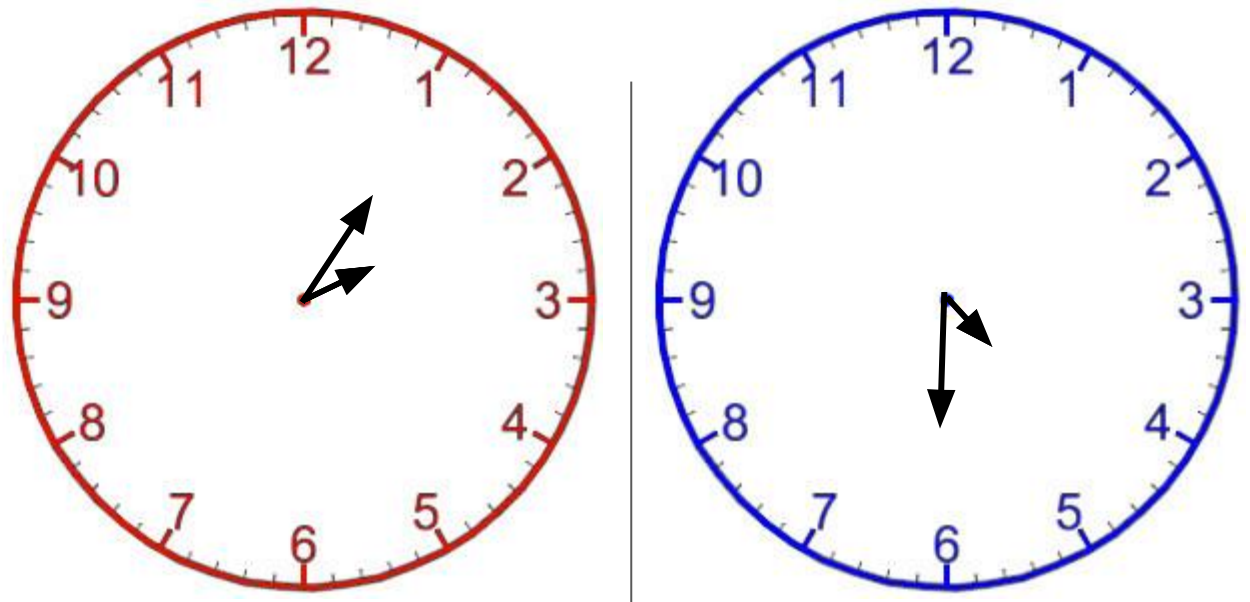


Masked Compound Prediction



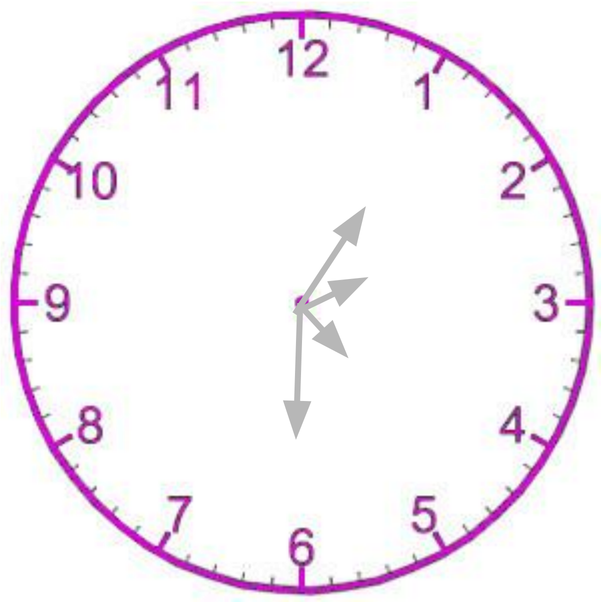
Integerized mask $m(i, j) \in [0, 64]$

Advanced Compound Predictors



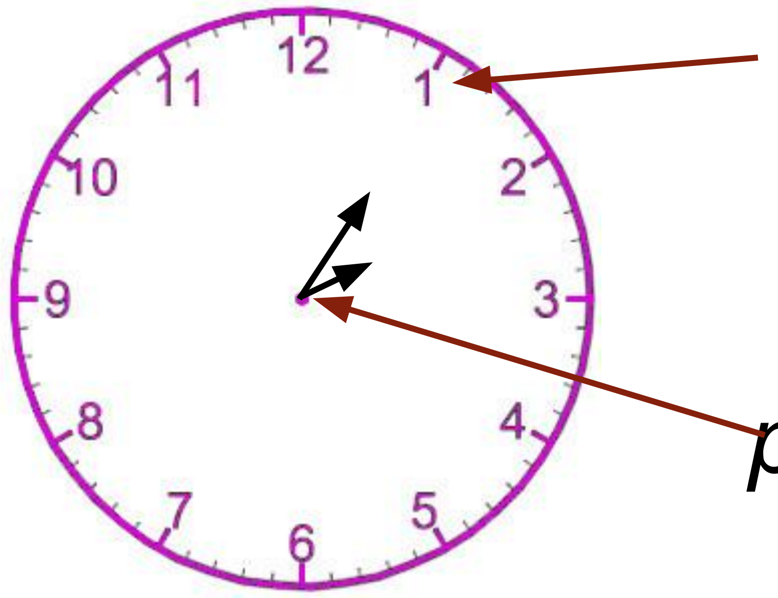
Predictor 1 | Predictor 2

Distance Weighted Predictor



distance in time determines weight for predictor

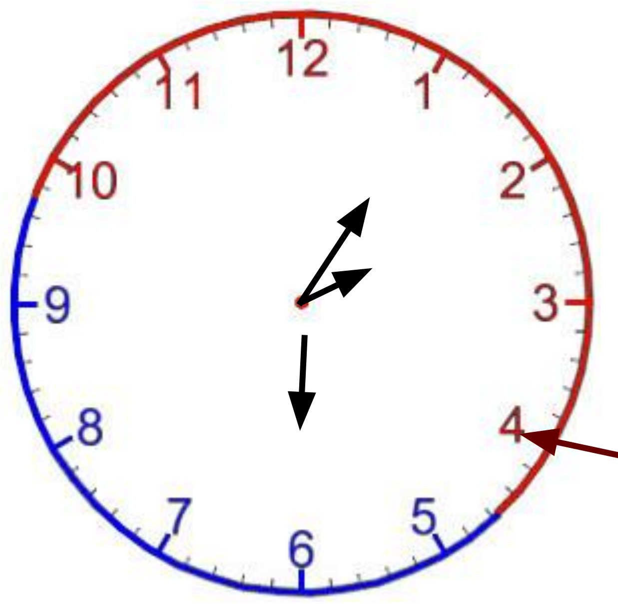
Difference Weighted Predictor



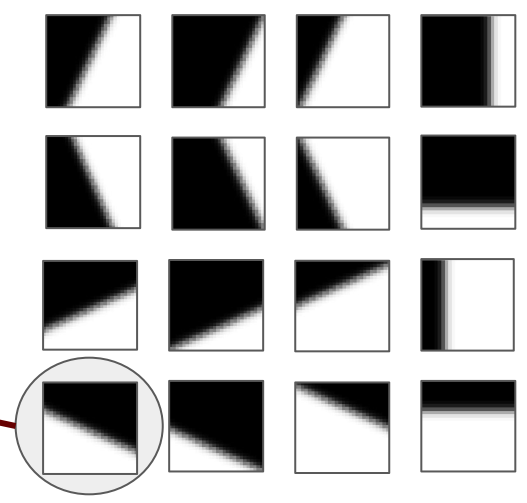
blend where similar

pick 1 where different

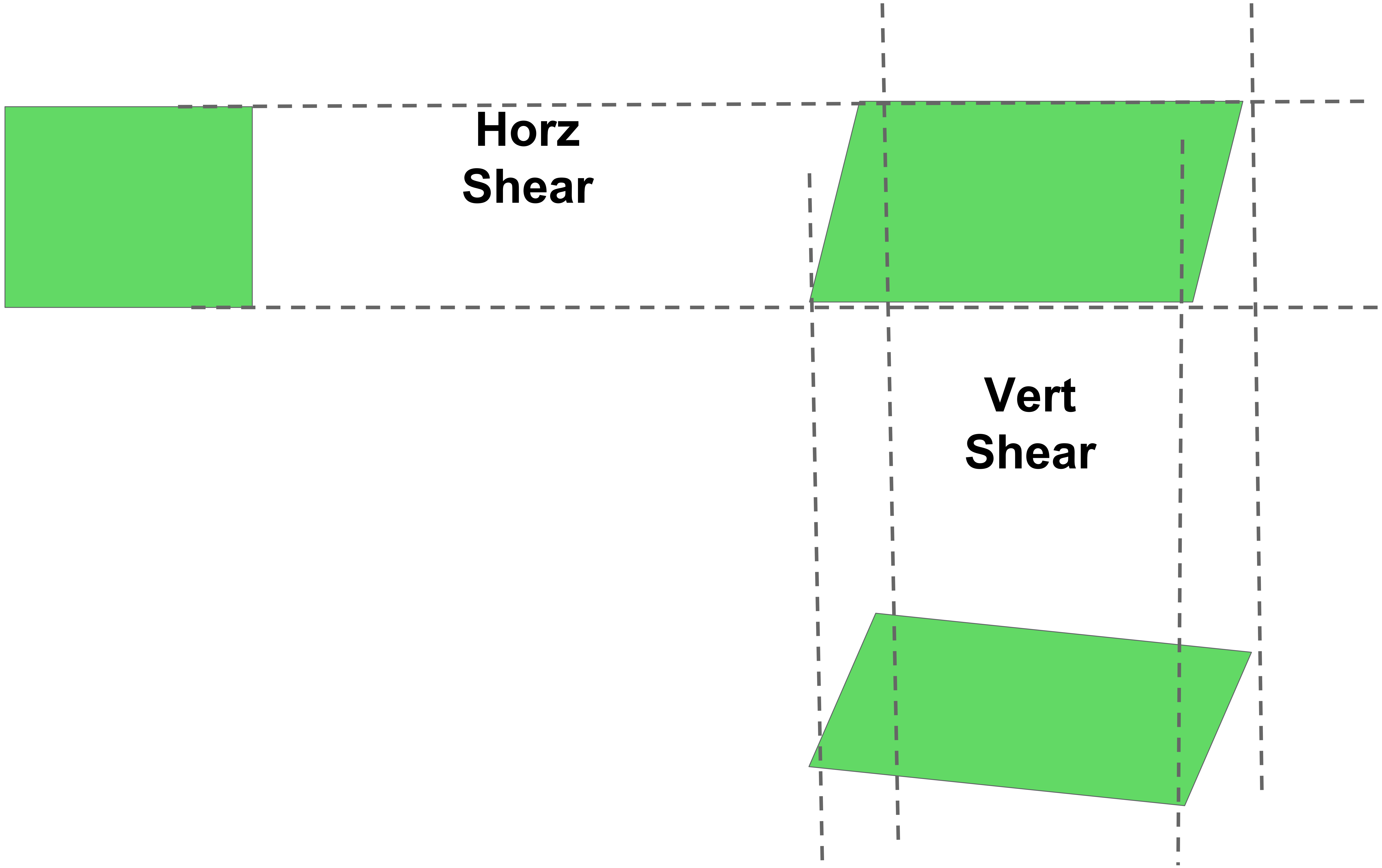
Wedge



pick mask



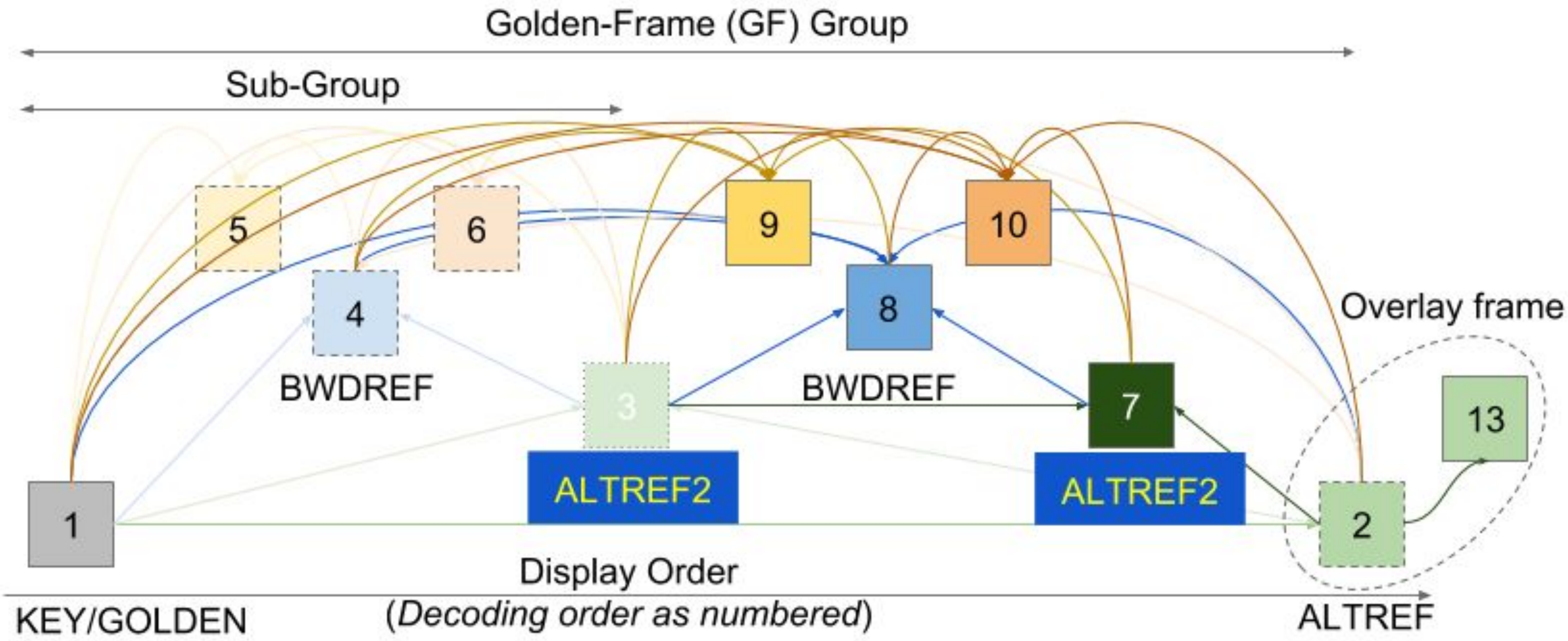
Warped Motion Compensation




W


W

Pyramid style encoding



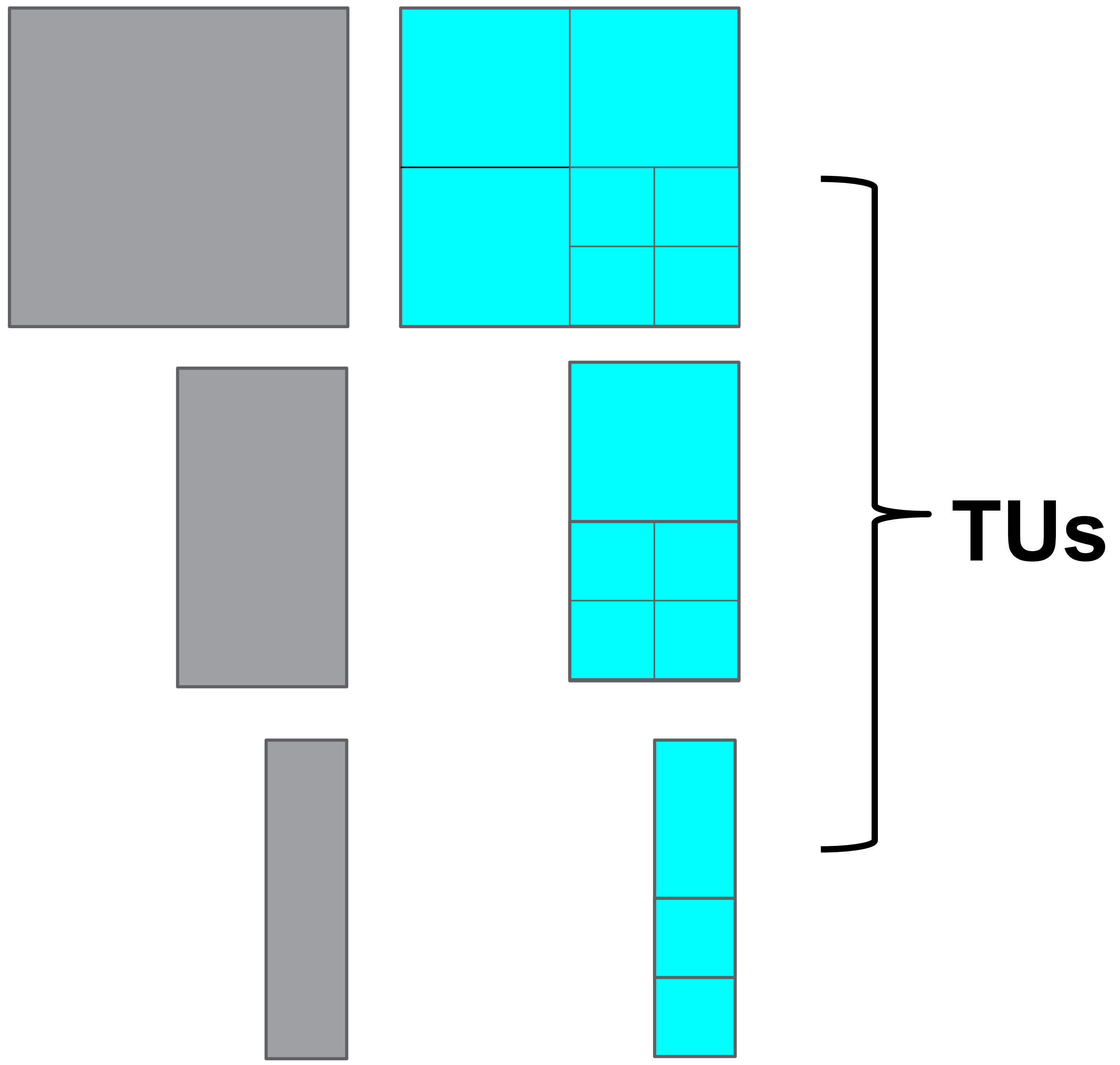
Video coding at a glance

Partition — *Predict* — ***Transform*** — *Quantize* — *Reconstruct* — *Encode*

Transform Block Partitioning

- **16** separable 2-D kernels: { **DCT**, **ADST**, **fADST**, **IDTX** }^{2}

64x64, 64x32, 32x64, 64x16, 16x64
32x32, 32x16, 16x32, 32x8, 8x32
16x16, 16x8, 8x16, 16x4, 4x16
8x8, 8x4, 4x8
4x4



Video coding at a glance

Partition — *Predict* — *Transform* — ***Quantize*** — *Reconstruct* — *Encode*

Quantization / Trellis

-3	0	0	1
-1	2	0	0
0	0	0	0
0	0	0	0

-3	0	0	0
-1	2	0	0
0	0	0	0
0	0	0	0

-3	0	0	1
-1	1	0	0
0	0	0	0
0	0	0	0

-3	0	0	1
0	2	0	0
0	0	0	0
0	0	0	0

-2	0	0	1
-1	2	0	0
0	0	0	0
0	0	0	0

-3	0	0	0
-1	1	0	0
0	0	0	0
0	0	0	0

-3	0	0	0
0	2	0	0
0	0	0	0
0	0	0	0

-2	0	0	0
-1	2	0	0
0	0	0	0
0	0	0	0

-3	0	0	1
0	1	0	0
0	0	0	0
0	0	0	0

-2	0	0	1
-1	1	0	0
0	0	0	0
0	0	0	0

-2	0	0	1
0	2	0	0
0	0	0	0
0	0	0	0

-3	0	0	0
0	1	0	0
0	0	0	0
0	0	0	0

-2	0	0	0
-1	1	0	0
0	0	0	0
0	0	0	0

-2	0	0	1
0	1	0	0
0	0	0	0
0	0	0	0

-2	0	0	0
0	1	0	0
0	0	0	0
0	0	0	0

-3	0	0	1
-1	2	0	0
0	0	0	0
0	0	0	0

TX Coefficient Coding

Encode EOB position

In reverse scan order starting at EOB

- encode magnitude of coefficient (up to 15) using context of up to 5 neighbors in same block that have already been coded

In scan order

- If coeff is not 0
    - if DC code the sign with context of above and left DC signs
    - else code sign
    - if coeff >= 15 golomb code coeff - 15

Example TX Coefficient Coding

zig-zag scan

0	1	5	6
2	4	7	12
3	8	11	13
9	10	14	15

TX coeffs

-17	0	4	1
-1	2	0	0
0	0	-1	0
1	0	0	0

Encoding process

-17	0	4	1
-1	2	0	0
0	0	-1	0
1	0	0	0

code
EOB = 11

-17	0	4	1
-1	2	0	0
0	0	-1	0
1	0	0	0

code 1 using context from values in yellow

-17	0	4	1
-1	2	0	0
0	0	-1	0
1	0	0	0

code 0 using context from values in yellow

-17	0	4	1
-1	2	0	0
0	0	-1	0
1	0	0	0

code 1 using context from values in yellow

...

-17	0	4	1
-1	2	0	0
0	0	-1	0
1	0	0	0

code 15+ using context from values in yellow

-17	0	4	1
-1	2	0	0
0	0	-1	0
1	0	0	0

golomb code 2 (17-15) & code (-) using context left and above dc signs

-17	0	4	1
-1	2	0	0
0	0	-1	0
1	0	0	0

skip because its a 0

-17	0	4	1
-1	2	0	0
0	0	-1	0
1	0	0	0

code (+)

-17	0	4	1
-1	2	0	0
0	0	-1	0
1	0	0	0

skip because its a 0

...

-17	0	4	1
-1	2	0	0
0	0	-1	0
1	0	0	0

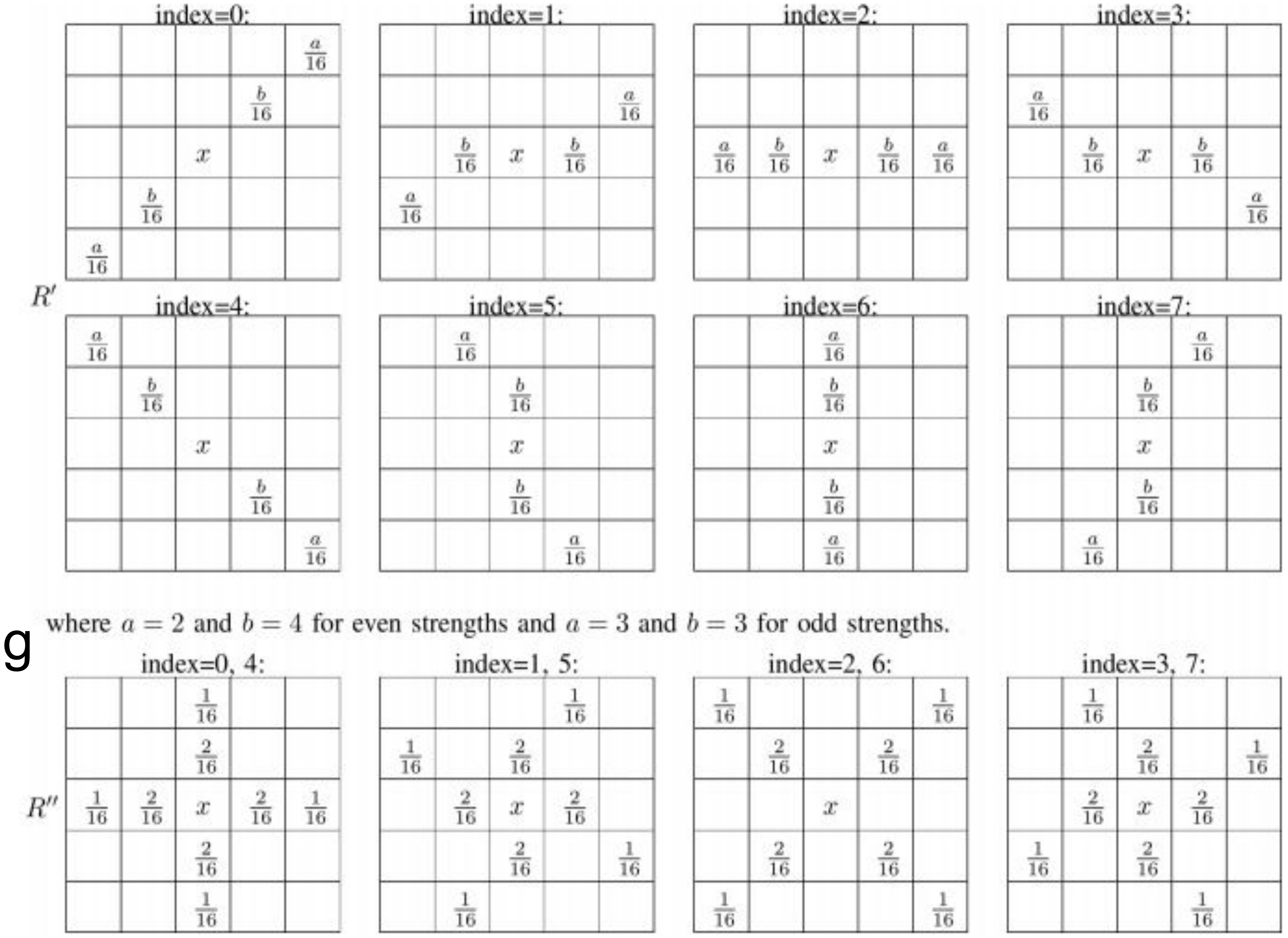
code (-)

Video coding at a glance

Partition — *Predict* — *Transform* — *Quantize* — ***Reconstruct*** — *Encode*

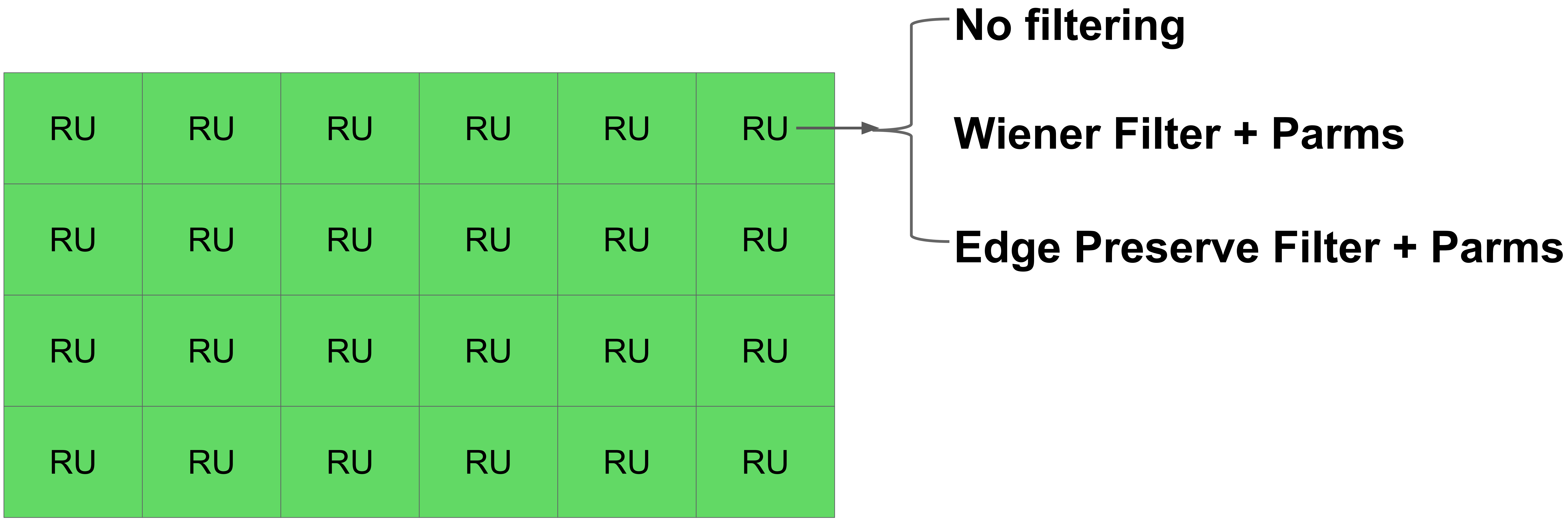
Constrained Dire. Enhancement Filtering

- Applied after deblocking
- Edge directions are estimated at 8x8 block level
- 5x5 pre-designed detail-preserving deringing filters are applied



where $a = 2$ and $b = 4$ for even strengths and $a = 3$ and $b = 3$ for odd strengths.

In-loop restoration Filters

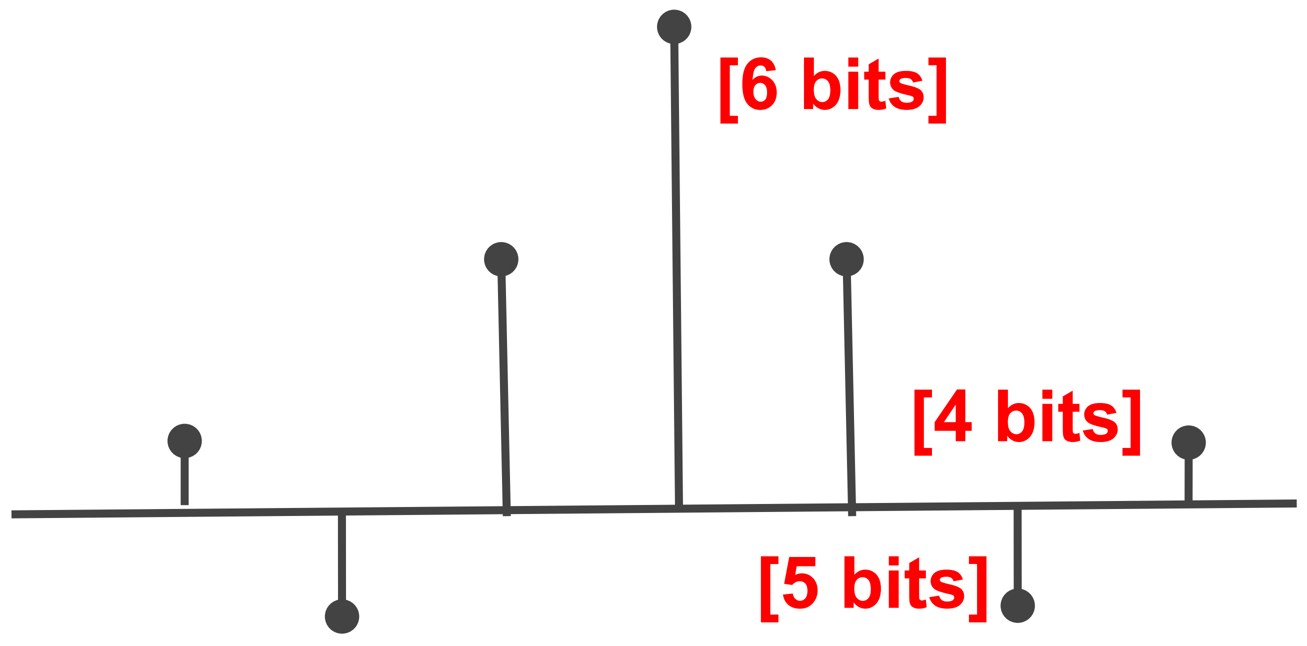


Frame

In-loop restoration Filters

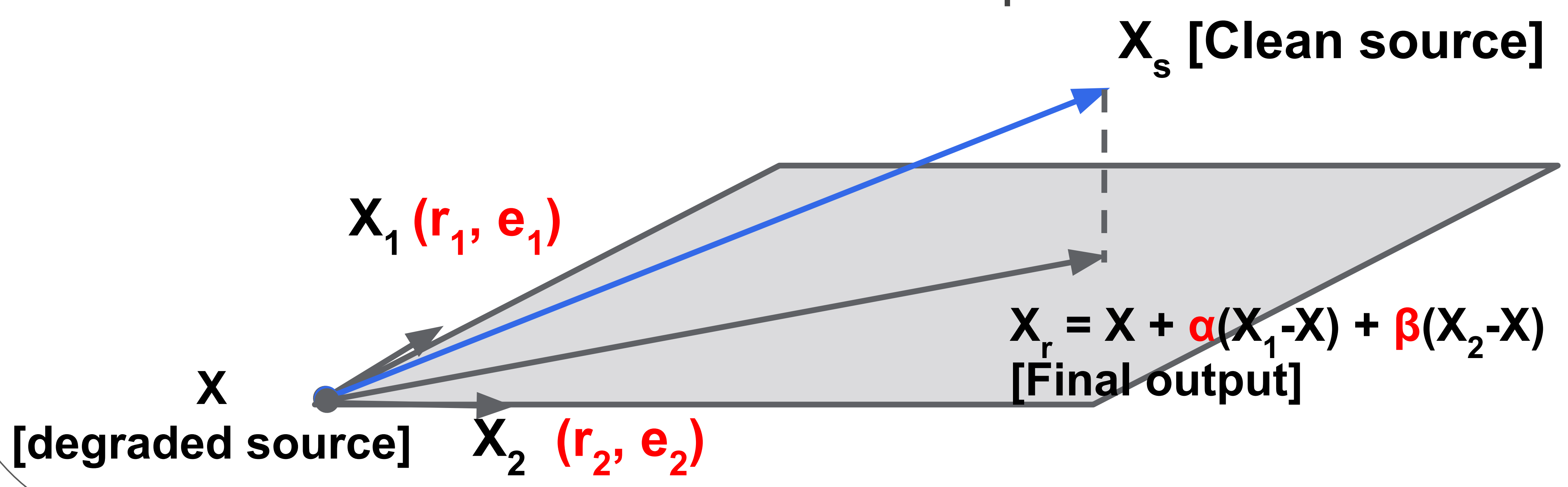
Type A: Wiener filter

Separable (horz + vert filter)
7-tap, symmetric, normalized

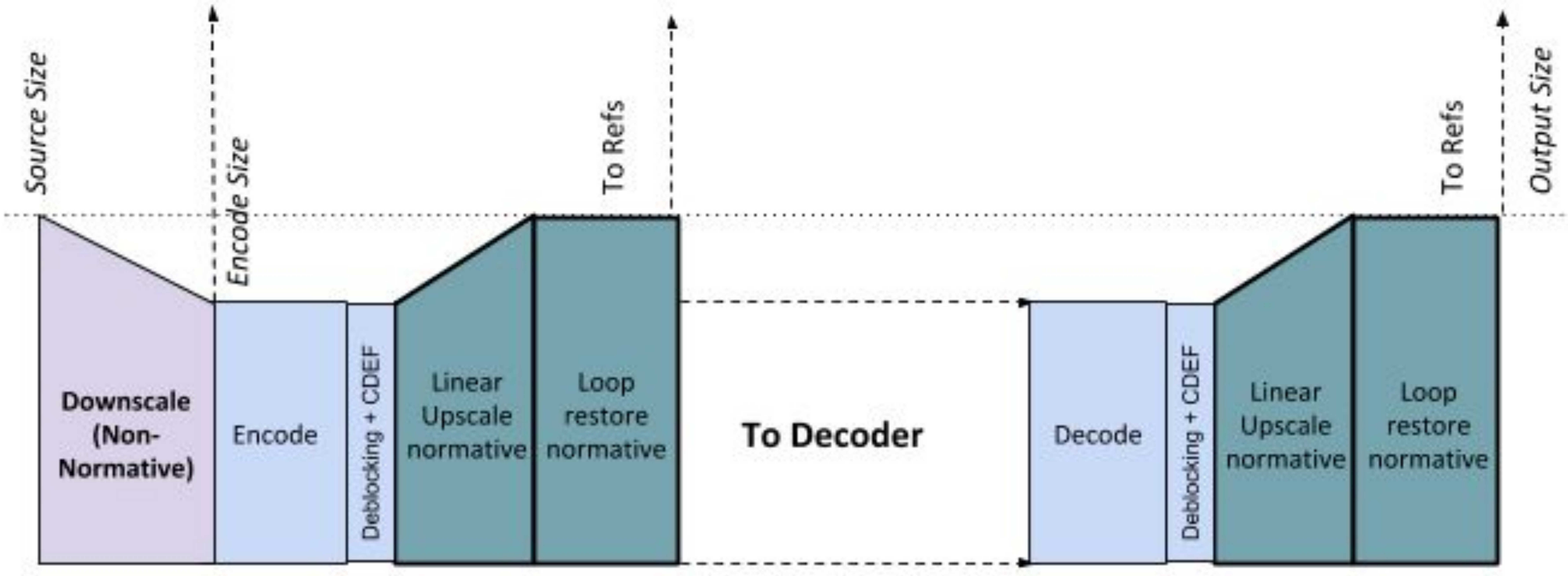


Type B: Self-guided projected filters

X_1 and X_2 are cheap restored versions, Subspace projection can yield a much better final restoration X_r.



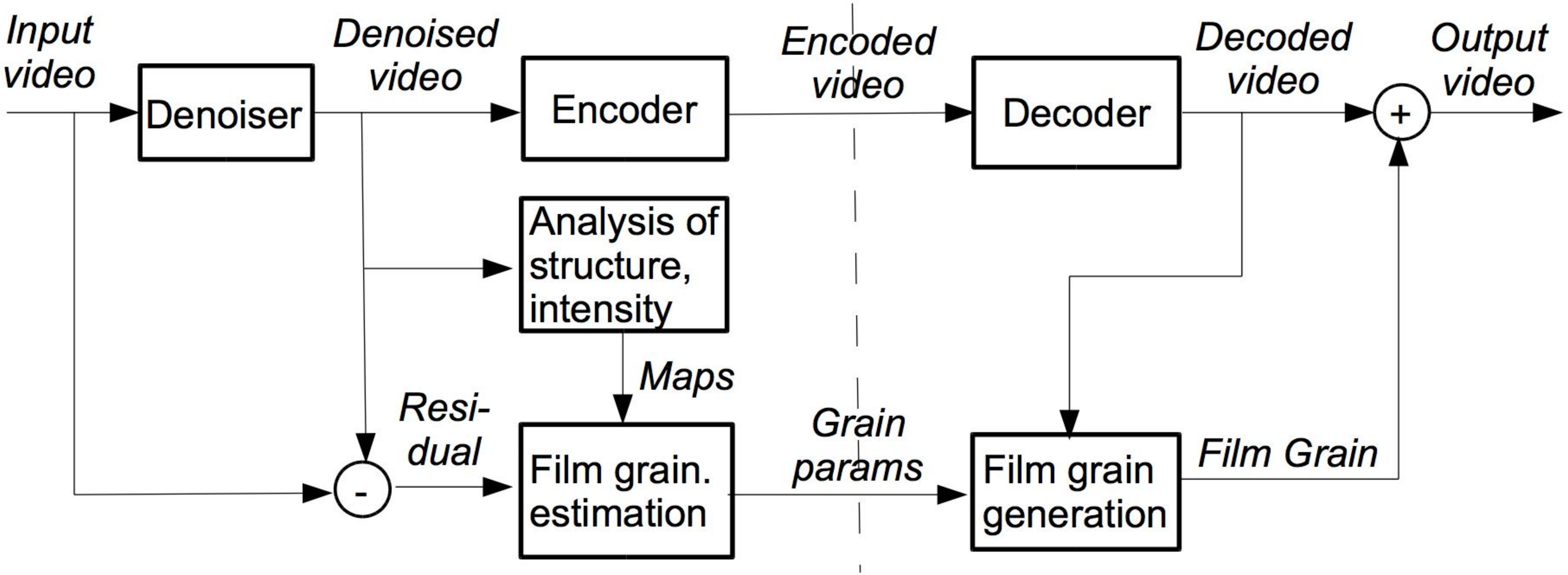
In-loop Frame Super Resolution



Film Grain Synthesis

- Film grain is present in much of the commercial content
- It is difficult to compress but needs to be preserved as part of creative intent
- AV1 supports film grain synthesis via a normative post-processing applied outside of the encoding/decoding loop

Film Grain Synthesis



Video coding at a glance

Partition · *Predict* · *Transform* · *Quantize* · *Reconstruct* · ***Encode***

AV1 Symbol Coding

- Most syntax elements have non-binary long alphabets
- AV1 multi-symbol arithmetic coder facilitates **high throughput** symbol coding and straightforward probability **model adaptation**
    - AV1 arithmetic coding is based on **15-bit CDF** tables
    - CDFs are **tracked** and **updated symbol-to-symbol**

Outline

AOMedia and AV1

Coding Techniques

Coding Performance

What's Next

Q & A

Compression Efficiency

- Test condition: AWCY[1] objective1-fast[2], 30 x 1080p~360p clips, 60 frames
- AV1 CQ mode, libvpx-VP9 CQ mode, x265 CRF mode
- BDRate (%)

Codecs \ Metric	PSNR-Y	PSNR-Cb	PSNR-Cr	CIEDE-2000
AV1 speed 0 vs. **libvpx speed 0**	-29.06	-32.41	-34.29	-31.12
AV1 speed 1 vs. **libvpx speed 0**	-27.15	-31.70	-33.35	-29.76
AV1 speed 0 vs. **x265 placebo**	-24.82	-41.69	-42.69	-35.60
AV1 speed 1 vs. **x265 placebo**	-22.81	-41.16	-42.07	-34.34

[1] arewecompressedyet.com **[2]** https://people.xiph.org/~tdaede/sets/objective-1-fast/

Compression Efficiency

- Results from Facebook Tests[1]

[1] https://code.facebook.com/posts/253852078523394/av1-beats-x264-and-libvpx-vp9-in-practical-use-case/


5.8
2.9
VP9
AV1

Coding Complexity

AV1 VBR mode at **speed 0~3**, compared against **libvpx-vp9 speed 0**

Resolution, encoder speed mode	ENC s/frame	ENC time vs libvpx	DEC frame/s	DEC time vs libvpx
720p-8 bit, speed 0	394	***175x***	68	***4.0x***
720p-8 bit, speed 1	99	***44x***	78	***3.5x***
720p-8 bit, speed 2	57	***25x***	66	***3.8x***
720p-8 bit, speed 3	34	***15x***	73	***3.7x***
1080p-10 bit, speed 0	2284	***141x***	18	***3.1x***
1080p-10 bit, speed 1	440	***27x***	19	***2.9x***
1080p-10 bit, speed 2	265	***16x***	18	***3.2x***
1080p-10 bit, speed 3	156	***10x***	19	***2.9x***

Outline

Prediction Type Choices

- **56 Single Reference Choices**
  - *7 frames * 4 Modes * 2 for OBMC*
- **12768 Compound Reference Choices**
  - *7 frames * 4 modes * 6 frames * 4 modes * (16 wedges + 1 weighted + 1 difference)*
- **71 Intra Modes**
  - *8 directions * 7 deltas + 12 DC modes + PAETH + INTRABLOCK_COPY + PALETTE*
- ***36708* Inter Intra Choices**
  - *(7 frames * 4 modes) * (8 directions * 7 deltas + 12 DC modes + PAETH) * (3 gradual + 16 wedges)*
- **49603 Total Prediction Choices**

Prediction Size Choices

Any single 8x8 block can be in any of the following partitionings

128x128, 32x128, 128x32, 64x128, 128x64,
64x64, 16x64, 64x16, 32x64, 64x32,
32x32, 8x32, 32x8, 16x32, 32x16,
16x16, 8x16, 16x8, 8x8

That’s 19 different prediction block sizes

Transform Choices

- **16** separable 2-D kernels:
  - (1 **DCT** + 1 **ADST** + 1 **fADST** + 1 **IDTX**) *
    (1 **DCT** + 1 **ADST** + 1 **fADST** + 1 **IDTX**)

Transform Sizes

- 3 choices for every coding blocksize
  - Full resolution
  - ½ width and ½ height
  - ¼ width and ½ height

Huge number of choices

Think **45,237,936** (ish) choices

Try everything encoder takes

9000 times as long as VP9

Machine Learning

- Figure out simple features to prune our search tree
  - split or no split partitioning
  - continue looking or quit
  - which modes to try
  - machine learned upscaling
  - Size to make frames

What's next?

- **Speed up** the codec
  - More SIMD coverage, ML based fast mode determination, ...
  - Set up and tune lower complexity speed modes (speed 2 - 8)
- Continue improving **compression performance**
  - Rate control, adaptive quantization, frame super resolution, …
  - Different eng usage modes will be explored, e.g. perceptual quality mode

On the table for Next Time

- Optical flow tests provided up to 50% gains (avg 15-20%)
- Render 3d to 2d + Video
- Learned Transforms
- Machine learned image / texture generation
- Hopefully some of **YOUR GREAT INVENTIONS**!

Outline

AOMedia and AV1

Coding Techniques

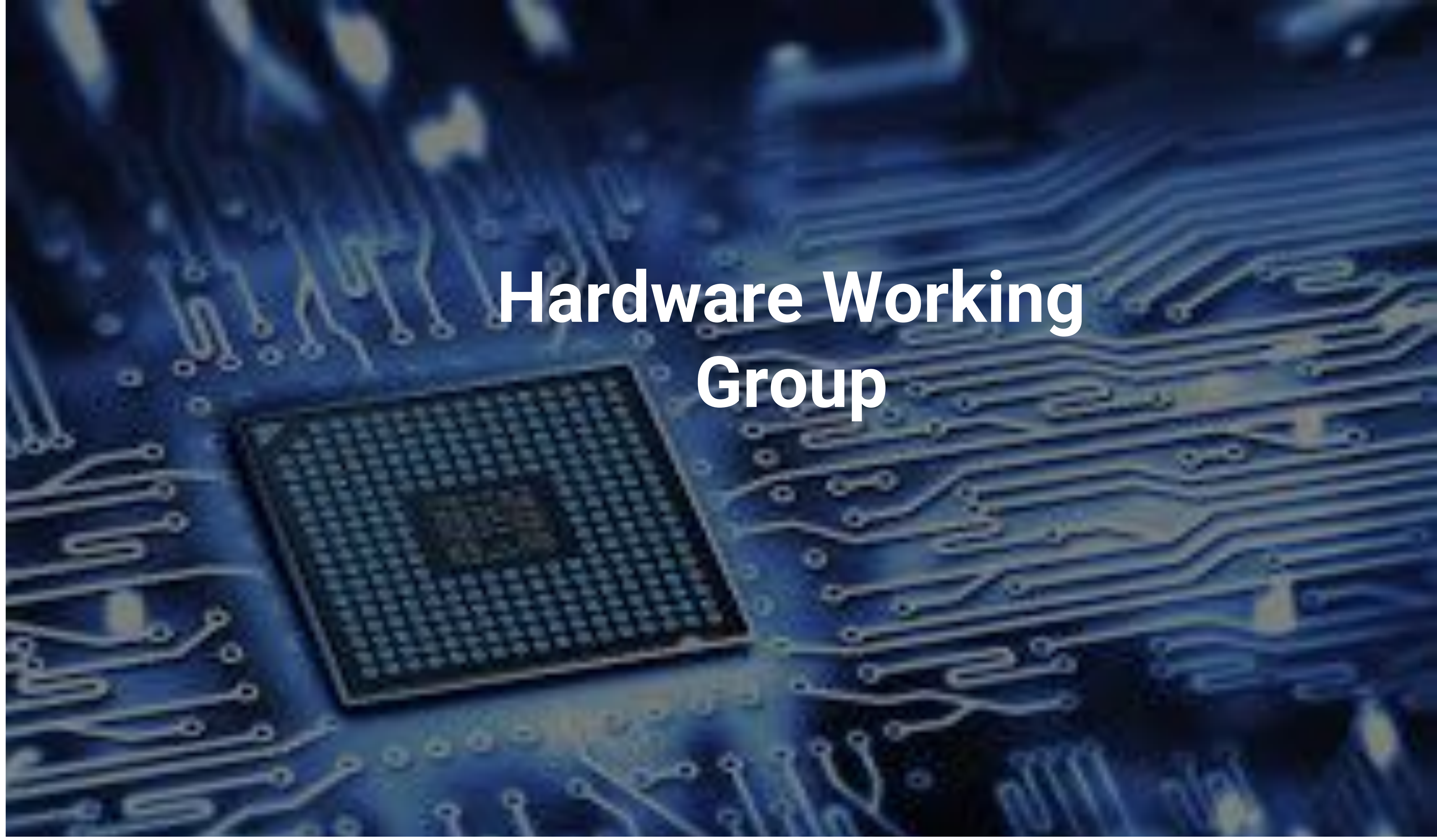
Coding Performance

What's Next

Q & A

Q + A


Codec Working Group


Hardware Working Group


Tapas Group


QA and Testing Group
BUG