INTEGRATING THOR TOOLS INTO THE EMERGING AV1 CODEC

Steinar Midtskogen, Arild Fuldseth, Gisle Bjøntegaard

Cisco Systems Inc.,

Philip Pedersens vei 1, 1366 Lysaker, Norway

Thomas Davies

Cisco Systems, Inc.,

Bedfont Lakes, Feltham, Middlesex, TW14 8HA, United Kingdom

ICIP2017

Royalty-free video codecs

- The deployment of recent compression technologies such as HEVC/H.265 may have been delayed or restricted due to their licensing terms
- Cisco has contributed to two parallel efforts aiming to standardise a royalty-free video codec:
  - The NETVC working group within IETF (since March 2015)
    - Our Thor codec proposed as the starting point
  - The Alliance for Open Media (since September 2015)
    - Google’s VP9 codec chosen as the starting point
- Cisco originally proposed to use Thor as the starting point for the AOMedia codec (AV1), but when VP9 was chosen instead, we investigated what Thor could add to VP9

What can Thor add to VP9/AV1?

Since Thor aims for reasonable compression at only moderate complexity, we considered features of Thor that could increase the compression efficiency of VP9 and/or reduce the computational complexity:

- VP9 lacks an in-loop deringing filter (like SAO in H.265), where Thor has a low complexity constrained low-pass filter (CLPF)
- Thor's filters for sub-pixel motion compensation have lower complexity
- Thor's support for quantisation weighting matrices adds the flexibility whether to favour metrics like PSNR or FAST-SSIM and PSNR-HVS

We also considered:

- Thor's chroma from luma prediction, but it was not directly transferable because VP9/AV1 allows different prediction modes for luma and chroma
- The transforms used in Thor (identical to those in H.265), but we hadn't time to investigate properly

Constrained low pass filter (CLPF)

- In-loop and applied after the deblocking filter as a separate pass
- The purpose is to reduce coding artefacts and improve overall image quality
- Often reclaims some of the losses introduced by compute constrained encoders making CLPF an attractive tool for low complexity or real-time encoders
- Simplicity guided the design: Finding the sweet spot of the most compression gain and the least computational complexity

Constrained low pass filter (CLPF)

- Modifies a pixel *x* at position *m, n* with strength *s*:

$$x'(m,n) = x(m,n) + \sum_{i,j \in R} a(i,j) f(x(m+i, n+j) - x(m,n), s)$$

- *f(x, y)* restricts *x* to *[-y, y]* – CLPF is a non-linear filter
- The *R* neighbourhood and values of *a(i, j)*:

		A		
B	C	X	D	E
		F		

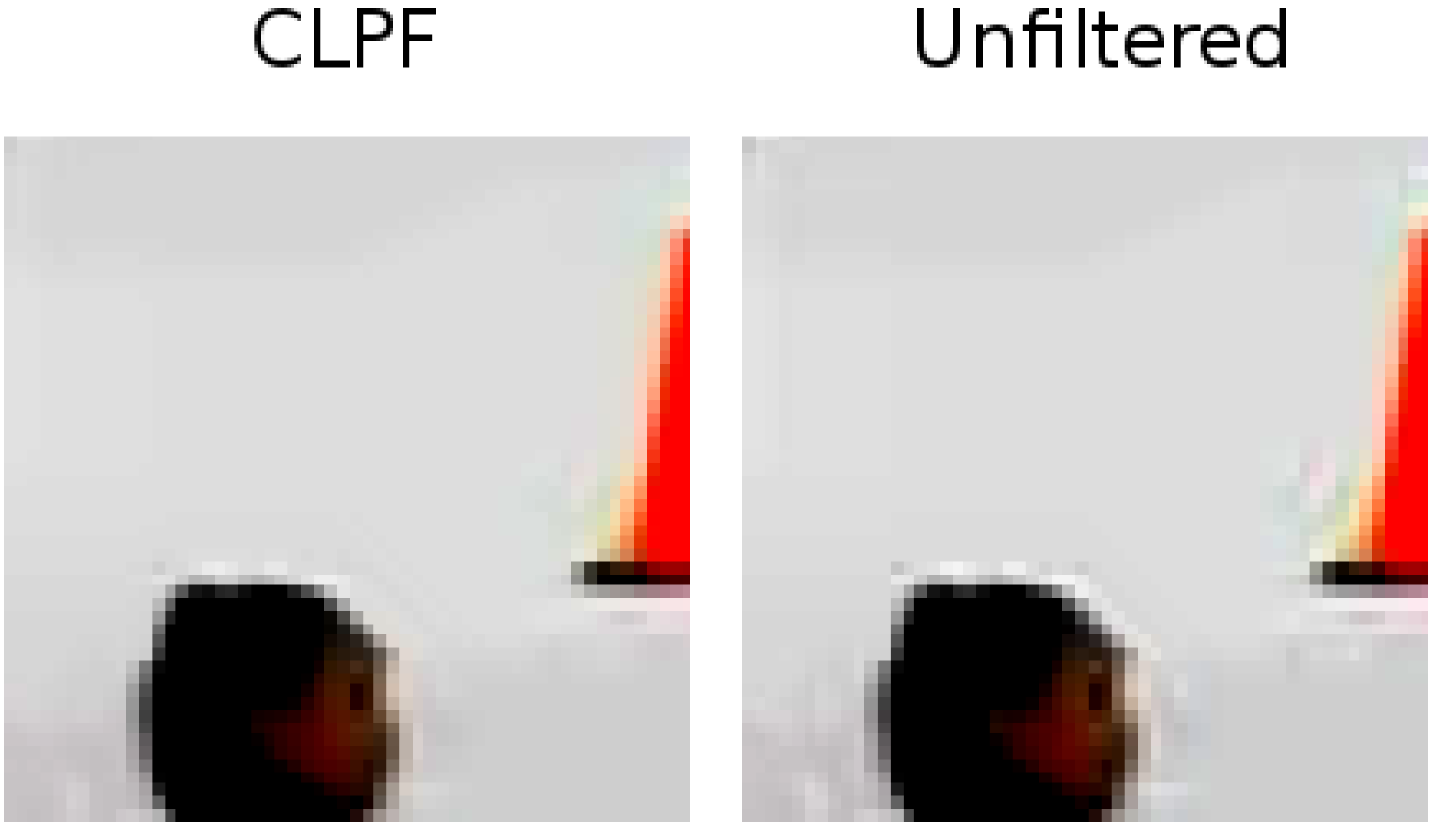
		$^4/_{16}$		
$^1/_{16}$	$^3/_{16}$		$^3/_{16}$	$^1/_{16}$
		$^4/_{16}$		

- Pseudo-code:

X' = X + (4*clip(A-X,-s,s) + clip(B-X,-s,s) + 3*clip(C-X,-s,s) +
3*clip(D-X,-s,s) + clip(E-X,-s,s) + 4*clip(F-X,-s,s))/16

Constrained low pass filter (CLPF)

- The frame is divided into 32x32, 64x64 or 128x128 filter blocks
- One filter strength is chosen for the entire frame: 0 (off), 1, 2 or 4
- Only non-skip coding blocks within a filter block may be filtered
- One bit is optionally signalled per luma filter block to disable the filter for that block if it contains at least one non-skip coding block
- All non-skip chroma coding blocks are filtered (no signalling)
- Limited search space for the encoder (13 options for a frame):
  - off
  - filter all qualified blocks with s=1, 2 or 4 (no block level signalling)
  - optionally filter blocks with s=1, 2 or 4 for block sizes 32x32, 64x64 or 128x128
- Offers a good compression/complexity trade-off compared to other proposed filters (“dering” and “loop restoration”)


CLPF
Unfiltered

Constrained low pass filter (CLPF)

- Compression results (BD-Rate) and complexity change. **Low delay** configuration.

(medium complexity configuration in parentheses)

	PSNR	PSNR HVS	SSIM	CIEDE 2000	APSNR	MS SSIM	encoder complexity	decoder complexity
CLPF	-2.79% (-5.93%)	-1.65% (-3.92%)	-2.21% (-5.86%)	-3.01% (-5.87%)	-2.76% (-5.85%)	-1.80% (-4.40%)	(-0.60%)	(8.64%)
dering	-2.59% (-5.25%)	-1.86% (-4.20%)	-2.50% (-5.89%)	-2.13% (-4.56%)	-2.54% (-5.15%)	-1.98% (-4.63%)	(1.63%)	(22.1%)
dering+CLPF	-3.61% (-7.25%)	-2.31% (-5.11%)	-3.06% (-7.35%)	-3.39% (-6.91%)	-3.56% (-7.14%)	-2.45% (-5.64%)	(1.84%)	(26.8%)
loop restoration	-4.19% (-7.36%)	-1.77% (-3.98%)	-3.12% (-6.87%)	-3.32% (-6.26%)	-4.14% (-7.27%)	-1.79% (-4.29%)	(385%)	(264%)

Constrained low pass filter (CLPF)

- Compression results (BD-Rate) and complexity change.
  High delay configuration.

  (medium complexity configuration in parentheses)

	PSNR	PSNR HVS	SSIM	CIEDE 2000	APSNR	MS SSIM	encoder complexity	decoder complexity
CLPF	-1.16% (-2.95%)	-0.41% (-1.50%)	-0.84% (-3.14%)	-1.42% (-3.24%)	-1.18% (-2.99%)	-0.50% (-2.04%)	(-0.62%)	(5.73%)
dering	-1.40% (-3.01%)	-0.77% (-2.04%)	-1.33% (-3.49%)	-1.05% (-2.70%)	-1.40% (-3.02%)	-0.89% (-2.46%)	(0.72%)	(16.7%)
dering+CLPF	-1.62% (-3.69%)	-0.61% (-1.98%)	-1.31% (-3.93%)	-1.56% (-3.72%)	-1.63% (-3.71%)	-0.75% (-2.54%)	(0.76%)	(19.8%)
loop restoration	-2.44% (-4.17%)	-0.71% (-1.62%)	-1.87% (-4.20%)	-1.89% (-3.72%)	-2.44% (-4.23%)	-0.76% (-2.13%)	(346%)	(239%)

Improved filter coefficients

- For motion compensation at sub-pixel resolution Thor uses computationally less complex filters:
  - Thor: 6 taps and 7 bit coefficients, no shift between horiz/vert
  - VP9: 8 taps and 8 bit coefficients, 7 bit shift between horiz/vert
- A reduction from 8 to 7 bits ensures that the sums after applying the filter in one direction stay within 16 bit
- New coefficients for AV1:
  - Standard filter: 6 taps, 7 bit coefficents (taken from Thor's uni-prediction filter)
  - Sharp filter: 8 taps and 7 bit coefficients
  - Smooth filter: 6 taps, 7 bit coefficients
- Coefficients added to accommodate for 1/16 sub-pixel resolutions

Improved filter coefficients

pos	−2	−1	0	1	2	3
$0/16$	0	0	64	0	0	0
$1/16$	1	−3	63	4	−1	0
$2/16$	1	−5	61	9	−2	0
$3/16$	1	−6	58	14	−4	1
$4/16$	1	−7	55	19	−5	1
$5/16$	1	−7	51	24	−6	1
$6/16$	1	−8	47	29	−6	1
$7/16$	1	−7	42	33	−6	1
$8/16$	1	−7	38	38	−7	1

Table 1: Standard filter coefficents

pos	−3	−2	−1	0	1	2	3	4
$0/16$	0	0	0	64	0	0	0	0
$1/16$	−1	1	−3	63	4	−1	1	0
$2/16$	−1	3	−6	62	8	−3	2	0
$3/16$	−1	4	−9	60	13	−5	3	−1
$4/16$	−2	5	−11	58	19	−7	3	−1
$5/16$	−2	5	−11	54	24	−9	4	−1
$6/16$	−2	5	−12	50	30	−10	4	−1
$7/16$	−2	5	−12	45	35	−11	5	−1
$8/16$	−2	6	−12	40	40	−12	6	−1

Table 2: Sharp filter coefficents

pos	−2	−1	0	1	2	3
$0/16$	0	0	64	0	0	0
$1/16$	1	14	31	17	1	0
$2/16$	0	13	31	18	2	0
$3/16$	0	11	31	20	2	0
$4/16$	0	10	30	21	3	0
$5/16$	0	9	29	22	4	0
$6/16$	0	8	28	23	5	0
$7/16$	−1	8	27	24	6	0
$8/16$	−1	7	26	26	7	−1

Table 3: Smooth filter coefficents

- Results:

metric / conf.	PSNR	PSNR HVS	SSIM	CIEDE 2000	APSNR	MS SSIM
low delay	−0.40%	−0.56%	−0.62%	−0.31%	−0.40%	−0.59%
high delay	−0.02%	−0.04%	−0.10%	0.13%	−0.03%	−0.07%

Table 9: Results with precision loss between the filter passes.

metric / conf.	PSNR	PSNR HVS	SSIM	CIEDE 2000	APSNR	MS SSIM
low delay	−0.98%	−1.02%	−1.41%	−1.58%	−0.98%	−1.24%
high delay	−0.45%	−0.43%	−0.62%	−0.89%	−0.46%	−0.43%

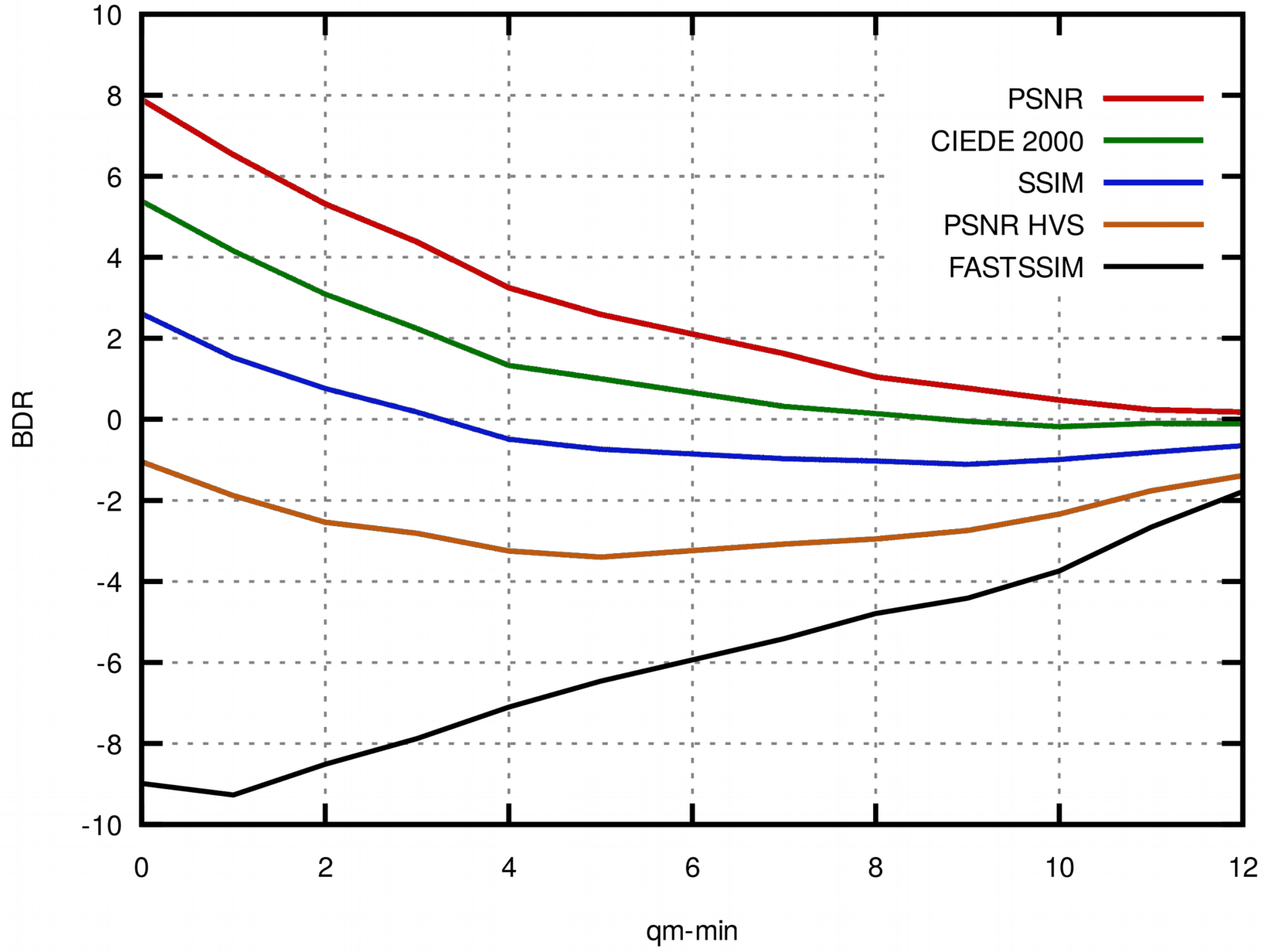
Table 10: Results without precision loss between the filter passes.

Quantisation matrices

- Allow variable quantiser step sizes for different frequencies to better match the human contrast sensitivity
- Widely used since at least JPEG, but lacking in VP9
- Hurts PSNR, but improves FAST-SSIM and PSNR-HVS
- A fixed matrix set is shared between the encoder and decoder containing matrices for each combination of:
  - transform block size (4x4, 8x8, 16x16, 32x32)
  - component type (luma, chroma)
  - block type (intra, inter)
  - quantisation matrix index indicating flatness (16 values)
- The flatness is signalled at the frame level

Quantisation matrices

- Results (BD-Rate for different strengths/levels of flatness)



Current status in AV1

The AV1 specification is work in progess. Much has changed since the paper was written and some details still remain. But:

- The proposed quantisation matrices were adopted with common intra/inter matrices & smaller matrices derived from larger to save memory
- The proposed interpolation filters were adopted using 2 bits increased resolution for intermediate sums
- CLPF was merged with another deringing proposal from Mozilla to form a joint proposal from Mozilla and Cisco: the Constrained Directional Enhancement filter (CDEF)
  - Presented separately in a grand challenge session at ICIP 2017
  - Adds compression gains over CLPF at the cost of added complexity

Questions?