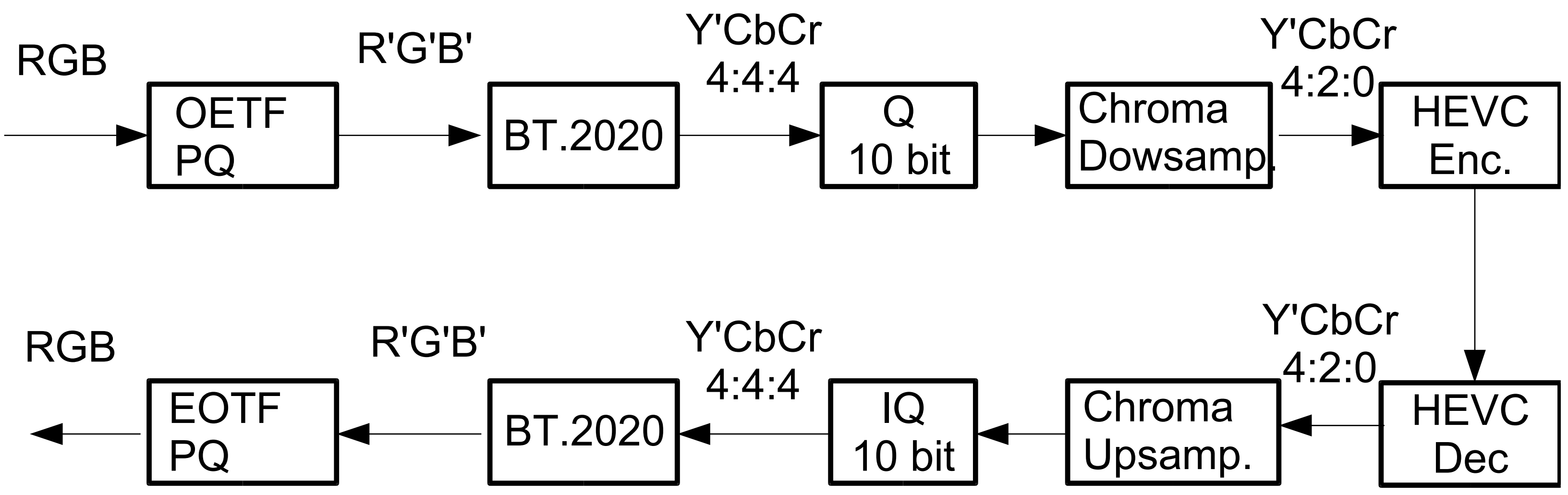
On HDR 4:2:0 chroma subsampling (AHG13 related)

Andrey Norkin
anorkin@netflix.com

JCTVC-Y0033

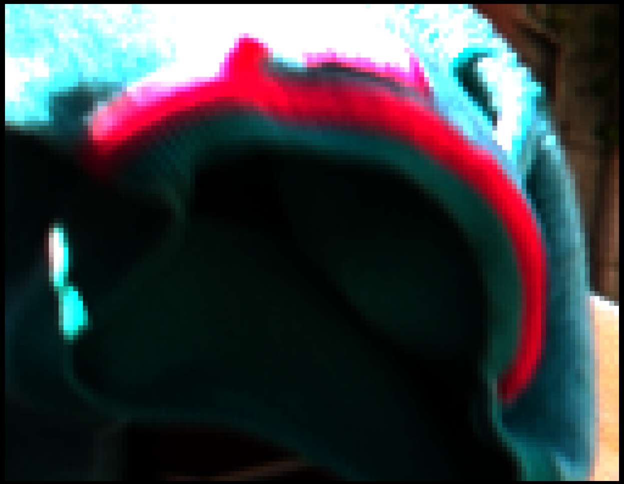
NETFLIX

HDR10 encoding and decoding pipeline

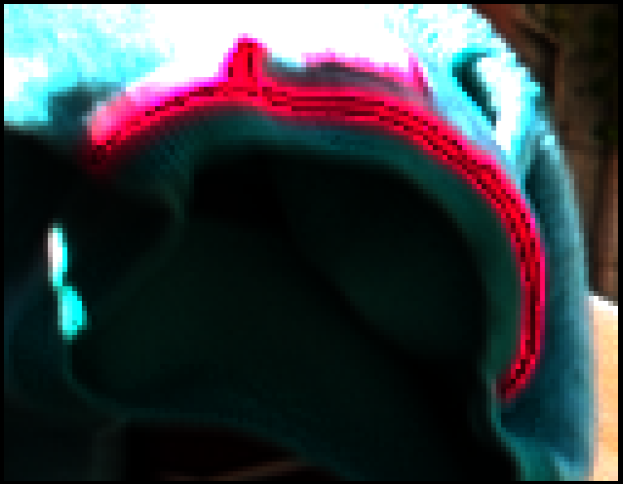


- HDR 10 is
  - BT.2020 color gamut
  - ST.2084 (PQ) transfer function
  - Y’CbCr 4:2:0 non-constant luminance
  - HEVC Main 10 profile
  - SEI messages

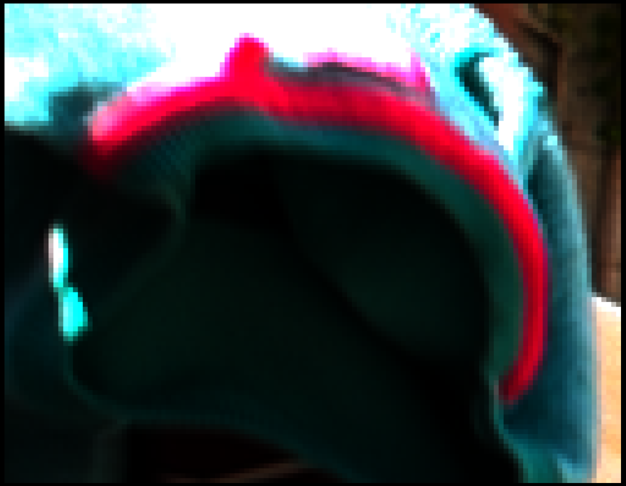
Subjective quality comparison (Fire Eater)



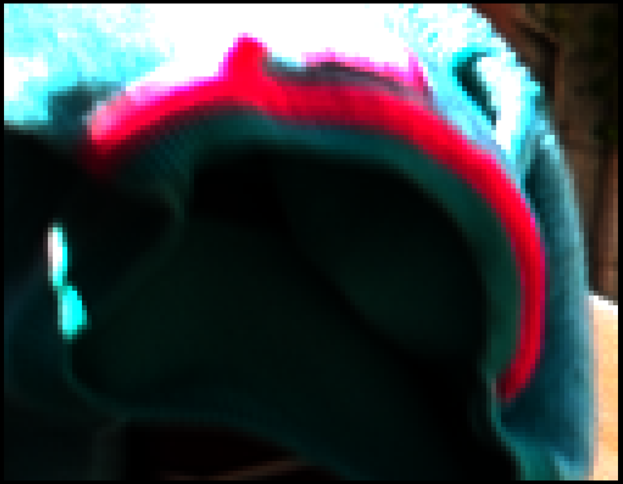
Original



Direct downsampling



Algorithm 1



Algorithm 2

Subjective quality comparison (Market)

Original

Direct downsampling

Algorithm 1

Algorithm 2

Proposed algorithms

- Motivation for this work:
  - Non-iterative approach
    - faster processing,
    - same number of operations per sample

- Proposed algorithm:
  - EOTF (RGB) -> R'G'B'
  - R'G'B' -> Y'CbCr 4:4:4
  - Downsample chroma Y'CbCr 4:4:4 -> Y'CbCr 4:2:0
  - Upsample chroma Y'CbCr 4:2:0 -> Y'CbCr 4:4:4
  - **Algorithm 1: Find luma Y' that minimizes (weighted) sum of squared errors per color component**
  - **Algorithm 2: Find luma Y' that minimizes weighted sum of per-component errors**

Algorithm 1

Minimize (weighted) sum of squared differences in linear R, G, and B

$$D = w_R\,(R_{new} - R_{org})^2 + w_G\,(G_{new} - G_{org})^2 + w_B\,(B_{new} - B_{org})^2$$

Denoting EOTF of component X' as $f(X')$,

$$D = w_R\,(f(R'_{new}) - f(R'_{org}))^2 + w_G\,(f(G'_{new}) - f(G'_{org}))^2 + w_B\,(f(B'_{new}) - f(B'_{org}))^2.$$

R', G', and B' are obtained from Y', Cb, Cr as follows

$$\begin{pmatrix} R' \\ G' \\ B' \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \begin{pmatrix} Y' \\ Cb \\ Cr \end{pmatrix}$$

Because of a complicated form of ST.2084 (PQ), finding a closed form solution might be difficult

Solution (linear approximation of EOTF)

First we find

$$e_R = -Y'_{org}\, \cancel{a}_{1,1} + (Cb_{new} - Cb_{org})\, a_{1,2} + (Cr_{new} - Cr_{org})\, a_{1,3}\,,$$
$$e_G = -Y'_{org}\, \cancel{a}_{2,1} + (Cb_{new} - Cb_{org})\, a_{2,2} + (Cr_{new} - Cr_{org})\, a_{2,3}\,,$$
$$e_B = -Y'_{org}\, \cancel{a}_{3,1} + (Cb_{new} - Cb_{org})\, a_{3,2} + (Cr_{new} - Cr_{org})\, a_{3,3}\,.$$

The value of Y' is equal to:

$$Y'_{new} = -\frac{w_R f'(R'_{org})^2 e_R a_{1,1} + w_G f'(G'_{org})^2 e_G a_{2,1} + w_B f'(B'_{org})^2 e_B a_{3,1}}{w_R f'(R'_{org})^2 a_{1,1}^2 + w_G f'(G'_{org})^2 a_{2,1}^2 + w_B f'(B'_{org})^2 a_{3,1}^2}$$

Provided $a_{1,1} = a_{2,1} = a_{3,1} = 1$, as in BT.709 and BT.2020, the expression simplifies to

$$Y'_{new} = -\frac{w_R f'(R'_{org})^2 e_R + w_G f'(G'_{org})^2 e_G + w_B f'(B'_{org})^2 e_B}{w_R f'(R'_{org})^2 + w_G f'(G'_{org})^2 + w_B f'(B'_{org})^2}$$

and , if all the weights are also set equal to 1, the Y' can be found as follows:

$$Y'_{new} = -\frac{f'(R'_{org})^2 e_R + f'(G'_{org})^2 e_G + f'(B'_{org})^2 e_B}{f'(R'_{org})^2 + f'(G'_{org})^2 + f'(B'_{org})^2}$$

Algorithm 2

Minimize a squared sum of weighted differences in linear R, G, and B

$$D = (w_R (R_{new} - R_{org}) + w_G (G_{new} - G_{org}) + w_B (B_{new} - B_{org}))^2$$

Denoting EOTF of component X' as $f(X')$,

$$D = (w_R (f(R'_{new}) - f(R'_{org})) + w_G (f(G'_{new}) - f(G'_{org})) + w_B (f(B'_{new}) - f(B'_{org})))^2.$$

R', G', and B' are obtained from Y', Cb, Cr as follows

$$\begin{pmatrix} R' \\ G' \\ B' \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \begin{pmatrix} Y' \\ Cb \\ Cr \end{pmatrix}$$

Solution (linear approximation of EOTF)

First we find

$$
\begin{aligned}
e_{\mathrm{R}} &= -\mathrm{Y'}_{\mathrm{org}}\, \cancel{a}_{1,1} + (\mathrm{Cb}_{\mathrm{new}} - \mathrm{Cb}_{\mathrm{org}})\, a_{1,2} + (\mathrm{Cr}_{\mathrm{new}} - \mathrm{Cr}_{\mathrm{org}})\, a_{1,3}\,, \\
e_{\mathrm{G}} &= -\mathrm{Y'}_{\mathrm{org}}\, \cancel{a}_{2,1} + (\mathrm{Cb}_{\mathrm{new}} - \mathrm{Cb}_{\mathrm{org}})\, a_{2,2} + (\mathrm{Cr}_{\mathrm{new}} - \mathrm{Cr}_{\mathrm{org}})\, a_{2,3}\,, \\
e_{\mathrm{B}} &= -\mathrm{Y'}_{\mathrm{org}}\, \cancel{a}_{3,1} + (\mathrm{Cb}_{\mathrm{new}} - \mathrm{Cb}_{\mathrm{org}})\, a_{3,2} + (\mathrm{Cr}_{\mathrm{new}} - \mathrm{Cr}_{\mathrm{org}})\, a_{3,3}\,.
\end{aligned}
$$

The value of Y' is equal to:

$$
Y'_{new} = -\frac{w_R f'(R'_{org})e_R + w_G f'(G'_{org})e_G + w_B f'(B'_{org})e_B}{w_R f'(R'_{org}) + w_G f'(G'_{org}) + w_B f'(B'_{org})}
$$

Experimental results

- Both algorithms have been implemented in HDRTools v.012 software package.
- LUT table have been implemented in in HDRTools v.012 for approximating both TF and its derivative.
- To study effect of chroma subsampling on colors at the color gamut boundary, simulations on BT.709 content in BT.709 container have been performed, i.e.
  RGB BT.709 -> YCbCr BT.709 -> YCbCr 4:2:0
- The experiment have also been performed on BT.2020 container
  RGB BT.709 -> YCbCr BT.709 -> YCbCr 4:2:0

Objective performance

Table 1. BT.709 in BT.709 container (averages). Total conversion time over all sequences in the set is reported.

Algorithm	**tPSNR-X**	**tPSNR-Y**	**tPSNR-Z**	**tPSNR-XYZ**	**tOSNR-XYZ**	**DE0100**	**MD0100**	**L0100**	**ConvTime(s)**
Direct	51.01	55.11	48.02	50.35	50.70	39.47	22.41	45.62	1403
Micro-grading	**56.57**	**69.77**	**47.68**	**51.87**	**51.76**	**39.96**	**22.46**	**49.63**	**6151**
Algorithm 1	59.26	54.05	46.84	50.59	51.08	40.09	22.26	44.30	2143
Algorithm 2	**56.54**	**66.67**	**47.65**	**51.82**	**51.74**	**39.96**	**22.40**	**49.27**	**2176**

Table 2. BT.2020 container (averages). Total conversion time over all sequences in the set is reported.

Algorithm	**tPSNR-X**	**tPSNR-Y**	**tPSNR-Z**	**tPSNR-XYZ**	**tOSNR-XYZ**	**DE0100**	**MD0100**	**L0100**	**ConvTime(s)**
Direct	52.40	62.69	44.43	48.42	47.34	37.96	22.78	48.72	1770
Micro-grading	**54.06**	**69.71**	**44.16**	**48.45**	**47.45**	**38.05**	**22.79**	**50.68**	**8972**
Algorithm 1	55.92	53.75	45.14	48.88	48.19	38.10	22.75	44.89	3350
Algorithm 2	**54.11**	**68.22**	**44.16**	**48.45**	**47.45**	**38.05**	**22.77**	**50.43**	**3461**

Computational complexity

- Number of operations per luma sample
  - Worst case in luma micro-grading (max. 10 iterations)
  - Linear approximation algorithms

- Squared EOTF derivative (or EOTF values) can be replaced by LUT

Table 3. Comparison of number of operations per luma sample in iterative micro-grading worst case (for maximum of 10 iterations) and proposed Algorithm 1.

Algorithm	Adds	Mults	Divs	Table look-ups	Compari-sons	Shifts (div by 2)
Iterative algorithm	55	69	0	30	10 (70)*	10
Algorithm 1 (no weights)	10	9	1	3	0 (2)*	0
Algorithm 2 (with weights)	10	12	1	3	0 (2)*	0

* When clipping of R', G', and B' in Iterative algorithm to the range (0, 1) or Y' in closed form solution is implemented with comparison operations and included in operations count.

Computational complexity

- Running times

Table 4. Runtimes of the algorithms, BT.709 test set

Algorithm	**Total running time (s)**	**Running time ratio over direct subsampling**
Direct	1403	100.0%
Micro-grading	6151	438.4%
Algorithm 1	2143	152.7%
Algorithm 2	2176	155.1%

Table 5. Runtimes of the algorithms, BT.2020 test set

Algorithm	**Total running time (s)**	**Running time ratio over direct subsampling**
Direct	1770	100.0%
Micro-grading	8972	506.9%
Algorithm 1	3350	189.3%
Algorithm 2	3461	195.5%

Conclusions and discussion

- Algorithms 1 and 2 remove color artifacts present in HDR10 videos
- Algorithm 2 performs very similar to luma micro-grading subjectively and objectively
- Both Algorithm 1 and 2 have much lower worst-case complexity than and lower average complexity than luma micro-grading

Thank you!