Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 4, 2017

A. Fuldseth
G. Bjontegaard
S. Midtskogen
T. Davies
M. Zanaty
Cisco

October 31, 2016

Thor Video Codec
draft-fuldseth-netvc-thor-03

Abstract

This document provides a high-level description of the Thor video codec. Thor is designed to achieve high compression efficiency with moderate complexity, using the well-known hybrid video coding approach of motion-compensated prediction and transform coding.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 4, 2017.

Copyright Notice

Table of Contents

1. Introduction

This document provides a high-level description of the Thor video codec. Thor is designed to achieve high compression efficiency with moderate complexity, using the well-known hybrid video coding approach of motion-compensated prediction and transform coding.

The Thor video codec is a block-based hybrid video codec similar in structure to widespread standards. The high level encoder and decoder structures are illustrated in Figure 1 and Figure 2 respectively.

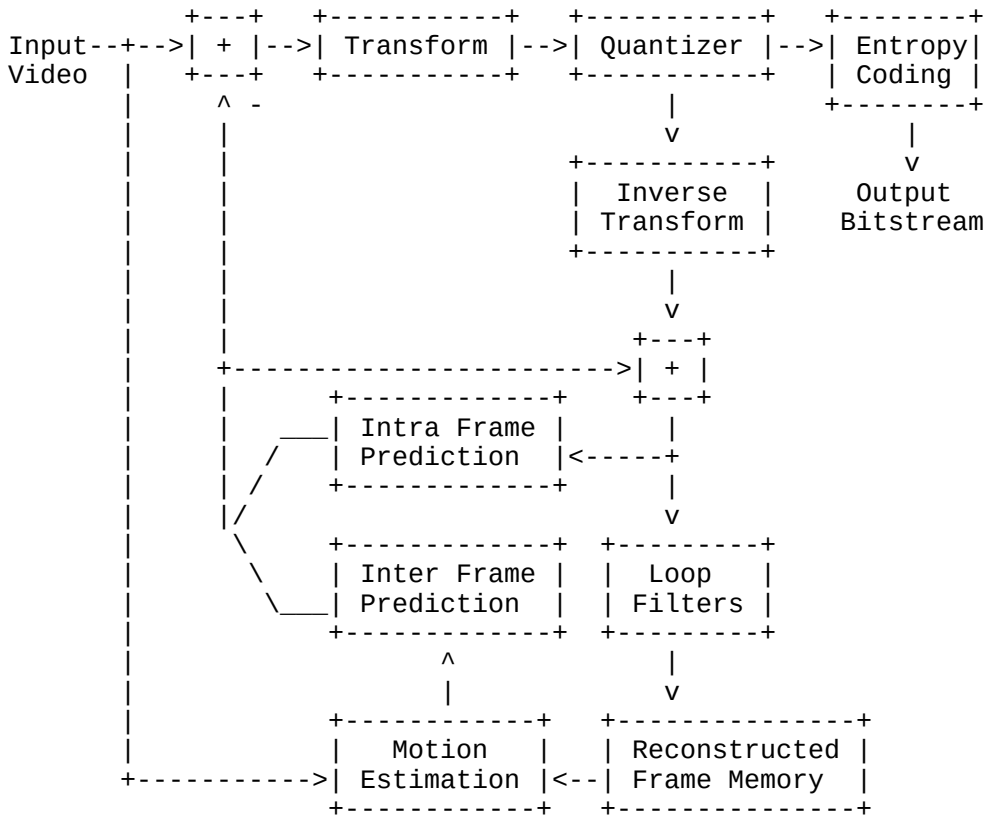

Figure 1: Encoder Structure

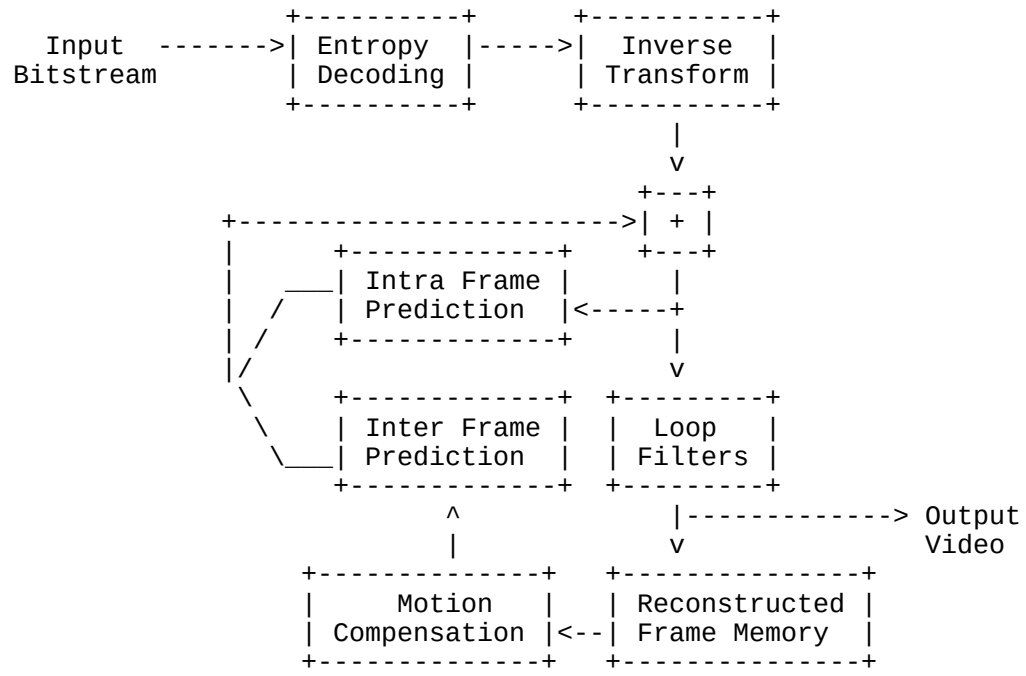

Figure 2: Decoder Structure

The remainder of this document is organized as follows. First, some requirements language and terms are defined. Block structures are described in detail, followed by intra-frame prediction techniques, inter-frame prediction techniques, transforms, quantization, loop filters, entropy coding, and finally high level syntax.

An open source reference implementation is available at github.com/cisco/thor.

2. Definitions

2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2.2. Terminology

This document frequently uses the following terms.

SB: Super Block - 64x64 or 128x128 block (luma pixels) which can be divided into CBs.

CB: Coding Block - Subdivision of a SB, down to 8x8 (luma pixels).

PB: Prediction Block - Subdivision of a CB, into 1, 2 or 4 equal blocks.

TB: Transform Block - Subdivision of a CB, into 1 or 4 equal blocks.

3. Block Structure

3.1. Super Blocks and Coding Blocks

Input frames with bitdepths of 8, 10 or 12 are supported. The internal bitdepth can be 8, 10 or 12 regardless if input bitdepth. The bitdepth of the output frames always follows the input frames. Chroma can be subsampled in both directions (4:2:0) or have full resolution (4:4:4).

Each frame is divided into 64x64 or 128x128 Super Blocks (SB) which are processed in raster-scan order. The SB size is signaled in the sequence header. Each SB can be divided into Coding Blocks (CB) using a quad-tree structure. The smallest allowed CB size is 8x8 luma pixels. The four CBs of a larger block are coded/signaled in the following order; upleft, downleft, upright, and downright.

The following modes are signaled at the CB level:

- Intra
- Inter0 (skip): MV index, no residual information
- Inter1 (merge): MV index, residual information
- Inter2 (uni-pred): explicit motion information, residual information
- Inter3 (ni-pred): explicit motion information, residual information

3.2. Special Processing at Frame Boundaries

At frame boundaries some square blocks might not be complete. For example, for 1920x1080 resolutions, the bottom row would consist of rectangular blocks of size 64x56. Rectangular blocks at frame boundaries are handled as follows. For each rectangular block, send one bit to choose between:

- A rectangular inter0 block and
- Further split.

For the bottom part of a 1920x1080 frame, this implies the following:

- For each 64x56 block, transmit one bit to signal a 64x56 inter0 block or a split into two 32x32 blocks and two 32x24 blocks.
- For each 32x24 block, transmit one bit to signal a 32x24 inter0 block or a split into two 16x16 blocks and two 16x8 blocks.
- For each 16x8 block, transmit one bit to signal a 16x8 inter0 block or a split into two 8x8 blocks.

Two examples of handling 64x56 blocks at the bottom row of a 1920x1080 frame are shown in Figure 3 and Figure 4 respectively.

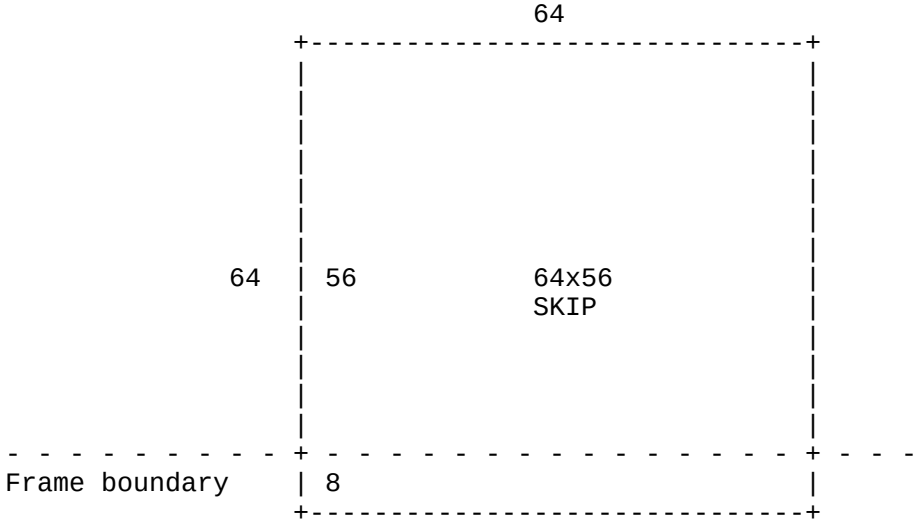

Figure 3: Super block at frame boundary

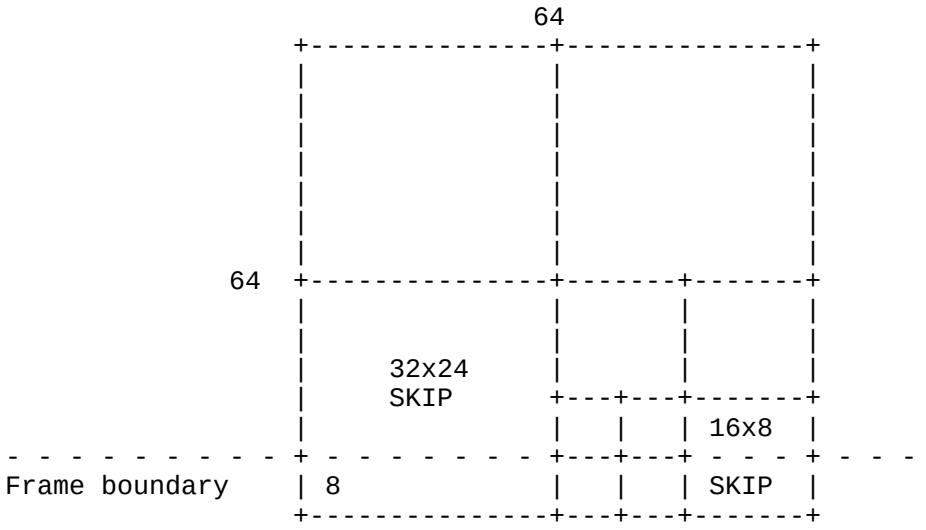

Figure 4: Coding block at frame boundary

3.3. Transform Blocks

A coding block (CB) can be divided into four smaller transform blocks (TBs).

3.4. Prediction Blocks

A coding block (CB) can also be divided into smaller prediction blocks (PBs) for the purpose of motion-compensated prediction. Horizontal, vertical and quad split are used.

4. Intra Prediction

8 intra prediction modes are used:

1. DC
2. Vertical (V)
3. Horizontal (H)
4. Upupright (north-northeast)
5. Upupleft (north-northwest)
6. Upleft (northwest)

7. Upleftleft (west-northwest)

8. Downleftleft (west-southwest)

The definition of DC, vertical, and horizontal modes are straightforward.

The upleft direction is exactly 45 degrees.

The upupright, upupleft, and upleftleft directions are equal to arctan(1/2) from the horizontal or vertical direction, since they are defined by going one pixel horizontally and two pixels vertically (or vice versa).

For the 5 angular intra modes (i.e. angle different from 90 degrees), the pixels of the neighbor blocks are filtered before they are used for prediction:

$y(n) = (x(n-1) + 2*x(n) + x(n+1) + 2)/4$

For the angular intra modes that are not 45 degrees, the prediction sometimes requires sample values at a half-pixel position. These sample values are determined by an additional filter:

$z(n + 1/2) = (y(n) + y(n+1))/2$

5. Inter Prediction

5.1. Multiple Reference Frames

Multiple reference frames are currently implemented as follows.

- Use a sliding-window process to keep the N most recent reconstructed frames in memory. The value of N is signaled in the sequence header.
- In the frame header, signal which of these frames shall be active for the current frame.
- For each CB, signal which of the active frames to be used for MC.

Combined with re-ordering, this allows for MPEG-1 style B frames.

A desirable future extension is to allow long-term reference frames in addition to the short-term reference frames defined by the sliding-window process.

5.2. Bi-Prediction

In case of bi-prediction, two reference indices and two motion vectors are signaled per CB. In the current version, PB-split is not allowed in bi-prediction mode. Sub-pixel interpolation is performed for each motion vector/reference index separately before doing an average between the two predicted blocks:

$$p(x,y) = (p0(x,y) + p1(x,y))/2$$

5.3. Improved chroma prediction

If specified in the sequence header, the chroma prediction, both intra and inter, or either, is improved by using the luma reconstruction if certain criteria are met. The process is described in the separate CLPF draft [I-D.midtskogen-netvc-chromapred].

5.4. Reordered Frames

Frames may be transmitted out of order. Reference frames are selected from the sliding window buffer as normal.

5.5. Interpolated Reference Frames

A flag is sent in the sequence header indicating that interpolated reference frames may be used.

If a frame is using an interpolated reference frame, it will be the first reference in the reference list, and will be interpolated from the second and third reference in the list. It is indicated by a reference index of -1 and has a frame number equal to that of the current frame.

The interpolated reference is created by a deterministic process common to the encoder and decoder, and described in the separate IRFVC draft [I-D.davies-netvc-irfvc].

5.6. Sub-Pixel Interpolation

5.6.1. Luma Poly-phase Filter

Inter prediction uses traditional block-based motion compensated prediction with quarter pixel resolution. A separable 6-tap poly-phase filter is the basis method for doing MC with sub-pixel accuracy. The luma filter coefficients are as follows:

When bi-prediction is enabled in the sequence header:

1/4 phase: [2,-10,59,17,-5,1]/64

2/4 phase: [1,-8,39,39,-8,1]/64

3/4 phase: [1,-5,17,59,-10,2]/64

When bi-prediction is disabled in the sequence header:

1/4 phase: [1,-7,55,19,-5,1]/64

2/4 phase: [1,-7,38,38,-7,1]/64

3/4 phase: [1,-5,19,55,-7,1]/64

With reference to Figure 5, a fractional sample value, e.g. i0,0 which has a phase of 1/4 in the horizontal dimension and a phase of 1/2 in the vertical dimension is calculated as follows:

a0,j = 2*A-2,i - 10*A-1,i + 59*A0,i + 17*A1,i - 5*A2,i + 1*A3,i

where j = -2,...,3

i0,0 = (1*a0,-2 - 8*a0,-1 + 39*a0,0 + 39*a0,1 - 8*a0,2 + 1*a0,3 + 2048)/4096

The minimum sub-block size is 8x8.

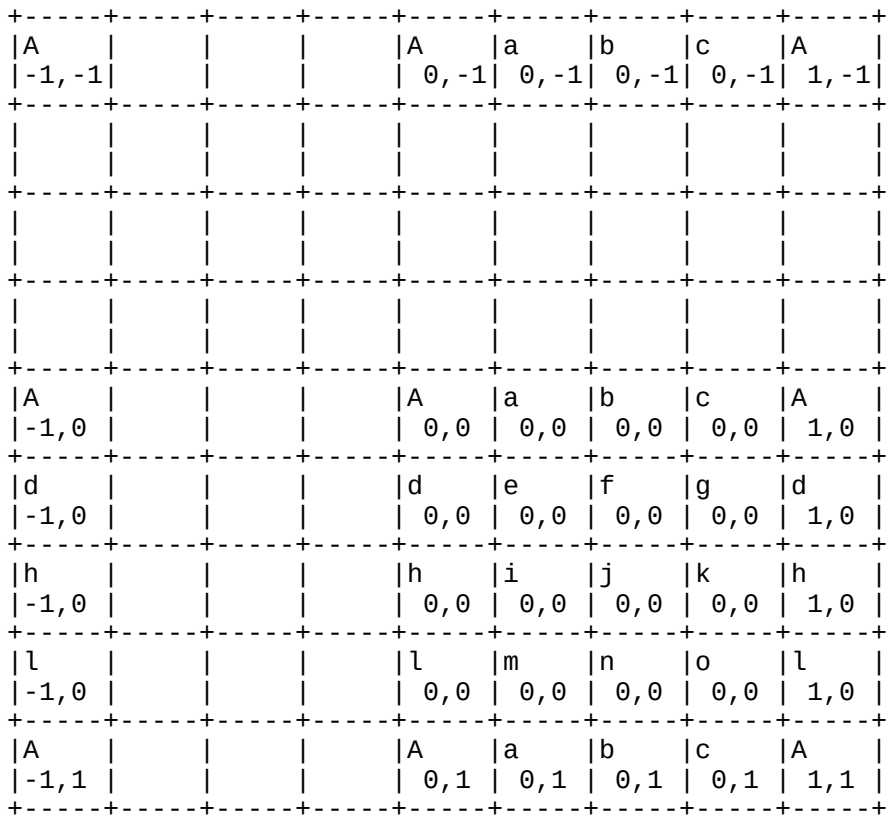

Figure 5: Sub-pixel positions

5.6.2. Luma Special Filter Position

For the fractional pixel position having exactly 2 quarter pixel offsets in each dimension, a non-separable filter is used to calculate the interpolated value. With reference to Figure 5, the center position j0,0 is calculated as follows:

j0,0 =

[0*A-1,-1 + 1*A0,-1 + 1*A1,-1 + 0*A2,-1 +

1*A-1,0 + 2*A0,0 + 2*A1,0 + 1*A2,0 +

1*A-1,1 + 2*A0,1 + 2*A1,1 + 1*A2,1 +

0*A-1,2 + 1*A0,2 + 1*A1,2 + 0*A2,2 + 8]/16

5.6.3. Chroma Poly-phase Filter

Chroma interpolation is performed with 1/8 pixel resolution using the following poly-phase filter.

1/8 phase: [-2, 58, 10, -2]/64

2/8 phase: [-4, 54, 16, -2]/64

3/8 phase: [-4, 44, 28, -4]/64

4/8 phase: [-4, 36, 36, -4]/64

5/8 phase: [-4, 28, 44, -4]/64

6/8 phase: [-2, 16, 54, -4]/64

7/8 phase: [-2, 10, 58, -2]/64

5.7. Motion Vector Coding

5.7.1. Inter0 and Inter1 Modes

Inter0 and inter1 modes imply signaling of a motion vector index to choose a motion vector from a list of candidate motion vectors with associated reference frame index. A list of motion vector candidates are derived from at most two different neighbor blocks, each having a unique motion vector/reference frame index. Signaling of the motion vector index uses 0 or 1 bit, dependent on the number of unique motion vector candidates. If the chosen neighbor block is coded in bi-prediction mode, the inter0 or inter1 block inherits both motion vectors, both reference indices and the bi-prediction property of the neighbor block.

For block sizes less than 64x64, inter0 has only one motion vector candidate, and its value is always zero.

Which neighbor blocks to use for motion vector candidates depends on the availability of the neighbor blocks (i.e. whether the neighbor blocks have already been coded, belong to the same slice and are not outside the frame boundaries). Four different availabilities, U, UR, L, and LL, are defined as illustrated in Figure 6. If the neighbor block is intra it is considered to be available but with a zero motion vector.

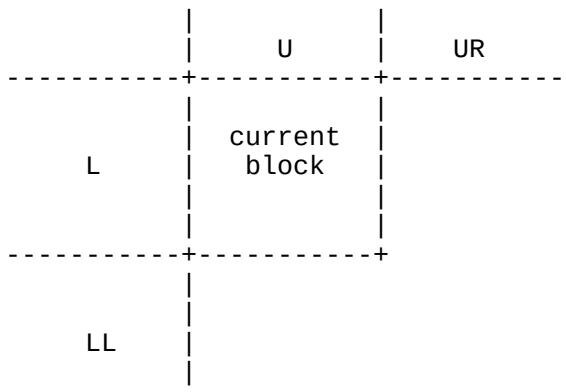

Figure 6: Availability of neighbor blocks

Based on the four availabilities defined above, each of the motion vector candidates is derived from one of the possible neighbor blocks defined in Figure 7.

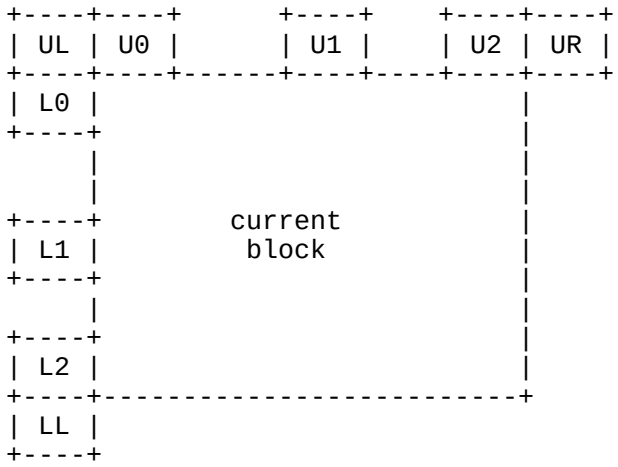

Figure 7: Motion vector candidates

The choice of motion vector candidates depends on the availability of neighbor blocks as shown in Table 1.

U	UR	L	LL	Motion vector candidates
0	0	0	0	zero vector
1	0	0	0	U2, zero vector
0	1	0	0	NA
1	1	0	0	U2,zero vector
0	0	1	0	L2, zero vector
1	0	1	0	U2,L2
0	1	1	0	NA
1	1	1	0	U2,L2
0	0	0	1	NA
1	0	0	1	NA
0	1	0	1	NA
1	1	0	1	NA
0	0	1	1	L2, zero vector
1	0	1	1	U2,L2
0	1	1	1	NA
1	1	1	1	U2,L2

Table 1: Motion vector candidates for different availability of neighbor blocks

5.7.2. Inter2 and Bi-Prediction Modes

Motion vectors are coded using motion vector prediction. The motion vector predictor is defined as the median of the motion vectors from three neighbor blocks. Definition of the motion vector predictor uses the same definition of availability and neighbors as in Figure 6 and Figure 7 respectively. The three vectors used for median filtering depends on the availability of neighbor blocks as shown in Table 2. If the neighbor block is coded in bi-prediction mode, only the first motion vector (in transmission order), MV0, is used as input to the median operator.

U	UR	L	LL	Motion vectors for median filtering
0	0	0	0	3 x zero vector
1	0	0	0	U0,U1,U2
0	1	0	0	NA
1	1	0	0	U0,U2,UR
0	0	1	0	L0,L1,L2
1	0	1	0	UL,U2,L2
0	1	1	0	NA
1	1	1	0	U0,UR,L2,L0
0	0	0	1	NA
1	0	0	1	NA
0	1	0	1	NA
1	1	0	1	NA
0	0	1	1	L0,L2,LL
1	0	1	1	U2,L0,LL
0	1	1	1	NA
1	1	1	1	U0,UR,L0

Table 2: Neighbor blocks used to define motion vector predictor through median filtering

5.7.3. Motion Vector Direction

Motion vectors referring to reference frames later in time than the current frame are stored with their sign reversed, and these reversed values are used for coding and motion vector prediction.

6. Transforms

Transforms are applied at the TB or CB level, implying that transform sizes range from 4x4 to 128x128. The transforms form an embedded structure meaning the transform matrix elements of the smaller transforms can be extracted from the larger transforms.

7. Quantization

For the 32x32, 64x64 and 128x128 transform sizes, only the 16x16 low frequency coefficients are quantized and transmitted.

The 64x64 inverse transform is defined as a 32x32 transform followed by duplicating each output sample into a 2x2 block. The 128x128 inverse transform is defined as a 32x32 transform followed by duplicating each output sample into a 4x4 block.

7.1. Quantization matrices

A flag is transmitted in the sequence header to indicate whether quantization matrices are used. If this flag is true, a 6 bit value qmtx_offset is transmitted in the sequence header to indicate matrix strength.

If used, then in dequantization a separate scaling factor is applied to each coefficient, so that the dequantized value of a coefficient ci at position i is:

```
(ci * d(q) * IW(i,c,s,t,q) + 2^(k + 5)) >> (k + 6)
```

Figure 8: Equation 1

where IW is the scale factor for coefficient position i with size s, frame type (inter/inter) t, component (Y, Cb or Cr) c and quantizer q; and k=k(s,q) is the dequantization shift. IW has scale 64, that is, a weight value of 64 is no different to unweighted dequantization.

7.1.1. Quantization matrix selection

The current luma qp value qpY and the offset value qmtx_offset determine a quantisation matrix set by the formula:

```
qmlevel = max(0,min(11,((qpY + qmtx_offset) * 12) / 44))
```

Figure 9: Equation 2

This selects one of the 12 different sets of default quantization matrix, with increasing qmlevel indicating increasing flatness.

For a given value of qmlevel, different weighting matrices are provided for all combinations of transform block size, type (intra/inter), and component (Y, Cb, Cr). Matrices at low qmlevel are flat (constant value 64). Matrices for inter frames have unity DC gain (i.e. value 64 at position 0), whereas those for intra frames are designed such that the inverse weighting matrix has unity energy gain (i.e. normalized sum-squared of the scaling factors is 1).

7.1.2. Quantization matrix design

Further details on the quantization matrix and implementation can be found in the separate QMTX draft [I-D.davies-netvc-qmtx].

8. Loop Filtering

8.1. Deblocking

8.1.1. Luma deblocking

Luma deblocking is performed on an 8x8 grid as follows:

1. For each vertical edge between two 8x8 blocks, calculate the following for each of line 2 and line 5 respectively:

   d = abs(a-b) + abs(c-d),

   where a and b, are on the left hand side of the block edge and c and d are on the right hand side of the block edge:

   a b | c d

2. For each line crossing the vertical edge, perform deblocking if and only if all of the following conditions are true:

   * d2+d5 < beta(QP)

   * The edge is also a transform block edge

   * abs(mvx(left)) > 2, or abs(mvx(right)) > 2, or

     abs(mvy(left)) > 2, or abs(mvy(right)) > 2, or

     One of the transform blocks on each side of the edge has non-zero coefficients, or

     One of the transform blocks on each side of the edge is coded using intra mode.

3. If deblocking is performed, calculate a delta value as follows:

   delta = clip((18*(c-b) - 6*(d-a) + 16)/32,tc,-tc),

   where tc is a QP-dependent value.

4. Next, modify two pixels on each side of the block edge as follows:

a′ = a + delta/2

b′ = b + delta

c′ = c + delta

d′ = d + delta/2

5. The same procedure is followed for horizontal block edges.

The relative positions of the samples, a, b, c, d and the motion vectors, MV, are illustrated in Figure 10.

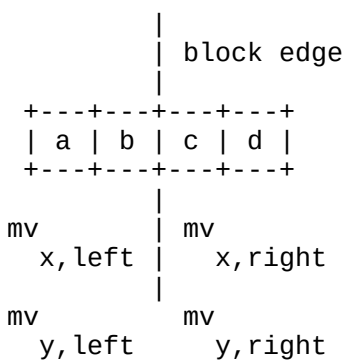

Figure 10: Deblocking filter pixel positions

8.1.2. Chroma Deblocking

Chroma deblocking is performed on a 4x4 grid as follows:

1. Delocking of the edge between two 4x4 blocks is performed if and only if:

   * The pixels on either side of the block edge belongs to an intra block.

   * The block edge is also an edge between two transform blocks.

2. If deblocking is performed, calculate a delta value as follows:

   delta = clip((4*(c-b) + (d-a) + 4)/8,tc,-tc),

   where tc is a QP-dependent value.

3. Next, modify one pixel on each side of the block edge as follows:

   b' = b + delta

   c' = c + delta

8.2. Constrained Low Pass Filter (CLPF)

A low-pass filter is applied after the deblocking filter if signaled in the sequence header. It can still be switched off for individual frames in the frame header. Also signaled in the frame header is whether to apply the filter for all qualified 128x128 blocks or to transmit a flag for each such block. A super block does not qualify if it only contains Inter0 (skip) coding block and no signal is transmitted for these blocks.

The filter is described in the separate CLPF draft [I-D.midtskogen-netvc-clpf].

9. Entropy coding

9.1. Overview

The following information is signaled at the sequence level:

- Sequence header

The following information is signaled at the frame level:

- Frame header

The following information is signaled at the CB level:

- Super-mode (mode, split, reference index for uni-prediction)
- Intra prediction mode
- PB-split (none, hor, ver, quad)
- TB-split (none or quad)
- Reference frame indices for bi-prediction
- Motion vector candidate index
- Transform coefficients if TB-split=0

The following information is signaled at the TB level:

- CBP (8 combinations of CBPY, CBPU, and CBPV)
- Transform coefficients

The following information is signaled at the PB level:

- Motion vector differences

9.2. Low Level Syntax

9.2.1. CB Level

```
    super-mode          (inter0/split/inter1/inter2-ref0/intra/inter2-ref1/inter
2-ref2/inter2-ref3,..)

    if (mode == inter0 || mode == inter1)

      mv_idx                    (one of up to 2 motion vector candidates)

    else if (mode == INTRA)

      intra_mode                (one of up to 8 intra modes)

      tb_split                  (NONE or QUAD, coded jointly with CBP for tb_spl
it=NONE)

    else if (mode == INTER)

      pb_split          (NONE,VER,HOR,QUAD)

      tb_split_and_cbp  (NONE or QUAD and CBP)

    else if (mode == BIPRED)

      mvd_x0, mvd_y0    (motion vector difference for first vector)

      mvd_x1, mvd_y1    (motion vector difference for second vector)

      ref_idx0, ref_idx1        (two reference indices)
```

9.2.2. PB Level

```
       if (mode == INTER2 || mode == BIPRED)

         mvd_x, mvd_y              (motion vector differences)
```

9.2.3. TB Level

```
if (mode != INTER0 and tb_split == 1)

  cbp                      (8 possibilities for CBPY/CBPU/CBPV)

if (mode != INTER0)

  transform coefficients
```

9.2.4. Super Mode

For each block of size NxN (64>=N>8), the following mutually exclusive events are jointly encoded using a single VLC code as follows (example using 4 reference frames):

If there is no interpolated reference frame:

```
INTER0      1
SPLIT       01
INTER1      001
INTER2-REF0 0001
BIPRED      00001
INTRA       000001
INTER2-REF1     0000001
INTER2-REF2     00000001
INTER2-REF3     00000000
```

If there is an interpolated reference frame:

```
INTER0      1
SPLIT       01
INTER1      001
BIPRED      0001
INTRA       00001
INTER2-REF1     000001
INTER2-REF2     0000001
INTER2-REF3     00000001
INTER2-REF0 00000000
```

If less than 4 reference frames is used, a shorter VLC table is used. If bi-pred is not possible, or split is not possible, they are omitted from the table and shorter codes are used for subsequent elements.

Additionally, depending on information from the blocks to the left and above (meta data and CBP), a different sorting of the events can be used, e.g.:

```
SPLIT       1
INTER1      01
INTER2-REF0 001
INTER0      0001
INTRA       00001
INTER2-REF1 000001
INTER2-REF2 0000001
INTER2-REF3 00000001
BIPRED      00000000
```

9.2.5. CBP

Calculate code as follows:

```
if (tb-split == 0)

  N = 4*CBPV + 2*CBPU + CBPY

else

  N = 8
```

Map the value of N to code through a table lookup:

```
code = table[N]
```

where the purpose of the table lookup is the sort the different values of code according to decreasing probability (typically CBPY=1, CBPU=0, CBPV=0 having the highest probability).

Use a different table depending on the values of CBPY in neighbor blocks (left and above).

Encode the value of code using a systematic VLC code.

9.2.6. Transform Coefficients

Transform coefficient coding uses a traditional zig-zag scan pattern to convert a 2D array of quantized transform coefficients, coeff, to a 1D array of samples. VLC coding of quantized transform coefficients starts from the low frequency end of the 1D array using two different modes; level-mode and run-mode, starting in level-mode:

- Level-mode
  - Encode each coefficient, coeff, separately
  - Each coefficient is encoded by:

+ The absolute value, level=abs(coeff), using a VLC code and

+ If level > 0, the sign bit (sign=0 or sign=1 for coeff>0 and coeff<0 respectively).

* If coefficient N is zero, switch to run-mode, starting from coefficient N+1.

o Run-mode

* For each non-zero coefficient, encode the combined event of:

1. Length of the zero-run, i.e. the number of zeros since the last non-zero coefficient.

2. Whether or not level=abs(coeff) is greater than 1.

3. End of block (EOB) indicating that there are no more non-zero coefficients.

* Additionally, if level = 1, code the sign bit.

* Additionally, if level > 1 define code = 2*(level-2)+sign,

* If the absolute value of coefficient N is larger than 1, switch to level-mode, starting from coefficient N+1.

Example

Figure 11 illustrates an example where 16 quantized transform coefficients are encoded.

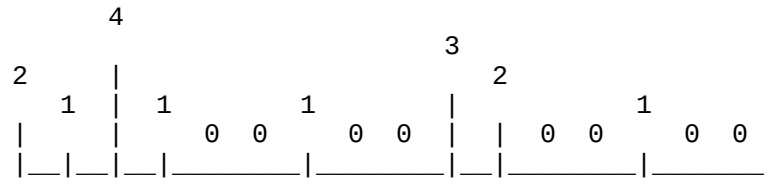

Figure 11: Coefficients to encode

Table 3 shows the mode, VLC number and symbols to be coded for each coefficient.

Index	abs(coeff)	Mode	Encoded symbols
0	2	level-mode	level=2,sign
1	1	level-mode	level=1,sign
2	4	level-mode	level=4,sign
3	1	level-mode	level=1,sign
4	0	level-mode	level=0
5	0	run-mode	
6	1	run-mode	(run=1,level=1)
7	0	run-mode	
8	0	run-mode	
9	3	run-mode	(run=1,level>1), 2*(3-2)+sign
10	2	level-mode	level=2, sign
11	0	level-mode	level=0
12	0	run-mode	
13	1	run-mode	(run=1,level=1)
14	0	run-mode	EOB
15	0	run-mode	

Table 3: Transform coefficient encoding for the example above.

10. High Level Syntax

High level syntax is currently very simple and rudimentary as the primary focus so far has been on compression performance. It is expected to evolve as functionality is added.

10.1. Sequence Header

- Width - 16 bits
- Height - 16 bits
- Enable/disable PB-split - 1 bit
- SB size - 3 bits
- Enable/disable TB-split - 1 bit
- Number of active reference frames (may go into frame header) - 2 bits (max 4)
- Enable/disable interpolated reference frames - 1 bit
- Enable/disable delta qp - 1 bit

- Enable/disable deblocking - 1 bit
- Constrained low-pass filter (CLPF) enable/disable - 1 bit
- Enable/disable block context coding - 1 bit
- Enable/disable bi-prediction - 1 bit
- Enable/disable quantization matrices - 1 bit
- If quantization matrices enabled: quantization matrix offset - 6 bits
- Select 420 or 444 input - 1 bit
- Number of reordered frames - 4 bits
- Enable/disable chroma intra prediction from luma - 1 bit
- Enable/disable chroma inter prediction from luma - 1 bit
- Internal frame bitdepth (8, 10 or 12 bits) - 2 bits
- Input video bitdepth (8, 10 or 12 bits) - 2 bits

10.2. Frame Header

- Frame type - 1 bit
- QP - 8 bits
- Identification of active reference frames - num_ref*4 bits
- Number of intra modes - 4 bits
- Number of active reference frames - 2 bits
- Active reference frames - number of active reference frames * 6 bits
- Frame number - 16 bits
- If CLPF is enabled in the sequence header: Constrained low-pass filter (CLPF) strength - 2 bits (00 = off, 01 = strength 1, 10 = strength 2, 11 = strength 4)
- IF CLPF is enabled in the sequence header: Enable/disable CLPF signal for each qualified filter block

11. IANA Considerations

This document has no IANA considerations yet. TBD

12. Security Considerations

This document has no security considerations yet. TBD

13. Normative References

[I-D.davies-netvc-irfvc]
Davies, T., "Interpolated reference frames for video coding", draft-davies-netvc-irfvc-00 (work in progress), October 2015.

[I-D.davies-netvc-qmtx]
Davies, T., "Quantisation matrices for Thor video coding", draft-davies-netvc-qmtx-00 (work in progress), March 2016.

[I-D.midtskogen-netvc-chromapred]
Midtskogen, S., "Improved chroma prediction", draft-midtskogen-netvc-chromapred-02 (work in progress), October 2016.

[I-D.midtskogen-netvc-clpf]
Midtskogen, S., Fuldseth, A., and M. Zanaty, "Constrained Low Pass Filter", draft-midtskogen-netvc-clpf-02 (work in progress), April 2016.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

Authors' Addresses

Arild Fuldseth
Cisco
Lysaker
Norway

Email: arilfuld@cisco.com

Gisle Bjontegaard
Cisco
Lysaker
Norway

Email: gbjonteg@cisco.com

Steinar Midtskogen
Cisco
Lysaker
Norway

Email: stemidts@cisco.com

Thomas Davies
Cisco
London
UK

Email: thdavies@cisco.com

Mo Zanaty
Cisco
RTP,NC
USA

Email: mzanaty@cisco.com

Network Working Group
Internet Draft
Intended status: Informational

A. Filippov
Huawei Technologies
A. Norkin
Netflix
J.R. Alvarez
Huawei Technologies

Expires: April 20, 2020

November 21, 2019

Video Codec Requirements and Evaluation Methodology
draft-ietf-netvc-requirements-10.txt

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts can be accessed at http://datatracker.ietf.org/drafts/current/

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on April 21, 2020.

Abstract

This document provides requirements for a video codec designed mainly for use over the Internet. In addition, this document describes an evaluation methodology needed for measuring the compression efficiency to ensure whether the stated requirements are fulfilled or not.

Table of Contents

1. Introduction

In this document, the requirements for a video codec designed mainly for use over the Internet are presented. The requirements encompass a wide range of applications that use data transmission over the Internet including Internet video streaming, IPTV, peer-to-peer video conferencing, video sharing, screencasting, game streaming and video monitoring / surveillance. For each application, typical resolutions, frame-rates and picture access modes are presented. Specific requirements related to data transmission over packet-loss networks are considered as well. In this document, when we discuss data protection techniques we only refer to methods designed and implemented to protect data inside the video codec since there are many existing techniques that protect generic data transmitted over networks with packet losses. From the theoretical point of view, both packet-loss and bit-error robustness can be beneficial for video codecs. In practice, packet losses are a more significant problem than bit corruption in IP networks. It is worth noting that there is an evident interdependence between possible amount of delay and the necessity of error robust video streams:

- If an amount of delay is not crucial for an application, then reliable transport protocols such as TCP that retransmits undelivered packets can be used to guarantee correct decoding of transmitted data.

- If the amount of delay must be kept low, then either data transmission should be error free (e.g., by using managed networks) or compressed video stream should be error resilient.

Thus, error resilience can be useful for delay-critical applications to provide low delay in packet-loss environment.

2. Definitions and abbreviations used in this document

Term	Meaning
High dynamic range imaging	is a set of techniques that allow a greater dynamic range of exposures or values (i.e., a wide range of values between light and dark areas) than normal digital imaging techniques. The intention is to accurately represent the wide range of intensity levels found in such examples as exterior scenes that include light-colored items struck by direct sunlight and areas of deep shadow [7].
Random access period	is the period of time between two closest independently decodable frames (pictures).
RD-point	A point in a two dimensional rate-distortion space where the values of bitrate and quality metric are used as x- and y-coordinates, respectively
Visually lossless compression	is a form or manner of lossy compression where the data that are lost after the file is compressed and decompressed is not detectable to the eye; the compressed data appearing identical to the uncompressed data [8].
Wide color gamut	is a certain complete color subset (e.g., considered in ITU-R BT.2020) that supports a wider range of colors (i.e., an extended range of colors that can be generated by a specific input or output device such as a video camera, monitor or printer and can be interpreted by a color model) than conventional color gamuts (e.g., considered in ITU-R BT.601 or BT.709).

Table 1. Definitions used in the text of this document

Abbreviation	Meaning
AI	All-Intra (each picture is intra-coded)
BD-Rate	Bjontegaard Delta Rate
FIZD	just the First picture is Intra-coded, Zero structural Delay
GOP	Group of Picture
HBR	High Bitrate Range
HDR	High Dynamic Range
HRD	Hypothetical Reference Decoder
IPTV	Internet Protocol Television
LBR	Low Bitrate Range
MBR	Medium Bitrate Range
MOS	Mean Opinion Score
MS-SSIM	Multi-Scale Structural Similarity quality index
PAM	Picture Access Mode
PSNR	Peak Signal-to-Noise Ratio
QoS	Quality of Service
QP	Quantization Parameter
RA	Random Access
RAP	Random Access Period
RD	Rate-Distortion
SEI	Supplemental Enhancement Information
UGC	User-Generated Content
VDI	Virtual Desktop Infrastructure
VUI	Video Usability Information
WCG	Wide Color Gamut

Table 2. Abbreviations used in the text of this document

3. Applications

In this chapter, an overview of video codec applications that are currently available on the Internet market is presented. It is worth noting that there are different use cases for each application that define a target platform, and hence there are different types of communication channels involved (e.g., wired or wireless channels) that are characterized by different quality of service as well as bandwidth; for instance, wired channels are considerably more error-free than wireless channels and therefore require different QoS approaches. The target platform, the channel bandwidth and the channel quality determine resolutions, frame-rates and quality or bit-rates for video streams to be encoded or decoded. By default, color format YCbCr 4:2:0 is assumed for the application scenarios listed below.

3.1. Internet Video Streaming

Typical content for this application is movies, TV-series and shows, and animation. Internet video streaming uses a variety of client devices and has to operate under changing network conditions. For this reason, an adaptive streaming model has been widely adopted. Video material is encoded at different quality levels and different resolutions, which are then chosen by a client depending on its capabilities and current network bandwidth. An example combination of resolutions and bitrates is shown in Table 3.

A video encoding pipeline in on-demand Internet video streaming typically operates as follows:

- Video is encoded in the cloud by software encoders.
- Source video is split into chunks, each of which is encoded separately, in parallel.
- Closed-GOP encoding with 2-5 second intra-picture intervals (or more) is used.
- Encoding is perceptually optimized. Perceptual quality is important and should be considered during the codec development.

Resolution *	Frame-rate, fps	PAM
4K, 3840x2160	24/1.001, 24, 25,	RA
2K (1080p), 1920x1080	30/1.001, 30, 50,	RA
1080i, 1920x1080*	60/1.001, 60, 100,	RA
720p, 1280x720	120/1.001, 120	RA
576p (EDTV), 720x576	The set of frame-rates	RA
576i (SDTV), 720x576*	presented in this table	RA
480p (EDTV), 720x480	is taken from Table 2	RA
480i (SDTV), 720x480*	in [1]	RA
512x384		RA
QVGA, 320x240		RA

Table 3. Internet Video Streaming: typical values of resolutions, frame-rates, and RAPs

NB *: Interlaced content can be handled at the higher system level and not necessarily by using specialized video coding tools. It is included in this table only for the sake of completeness as most video content today is in the progressive format.

Characteristics and requirements of this application scenario are as follows:

- High encoder complexity (up to 10x and more) can be tolerated since encoding happens once and in parallel for different segments.

- Decoding complexity should be kept at reasonable levels to enable efficient decoder implementation.

- Support and efficient encoding of a wide range of content types and formats is required:

. High Dynamic Range (HDR), Wide Color Gamut (WCG), high resolution (currently, up to 4K), high frame-rate content are important use cases, the codec should be able to encode such content efficiently.

. Coding efficiency improvement at both lower and higher resolutions is important since low resolutions are used when streaming in low bandwidth conditions.

. Improvement on both "easy" and "difficult" content in terms of compression efficiency at the same quality level contributes to the overall bitrate/storage savings.

. Film grain (and sometimes other types of noise) is often present in the streaming movie-type content and is usually a part of the creative intent.

- Significant improvements in compression efficiency between generations of video standards are desirable since this scenario typically assumes long-term support of legacy video codecs.

- Random access points are inserted frequently (one per 2-5 seconds) to enable switching between resolutions and fast-forward playback.

- Elementary stream should have a model that allows easy parsing and identification of the sample components.

- Middle QP values are normally used in streaming, this is also the range where compression efficiency is important for this scenario.

- Scalability or other forms of supporting multiple quality representations are beneficial if they do not incur significant bitrate overhead and if mandated in the first version.

3.2. Internet Protocol Television (IPTV)

This is a service for delivering television content over IP-based networks. IPTV may be classified into two main groups based on the type of delivery, as follows:

- unicast (e.g., for video on demand), where delay is not crucial;

- multicast/broadcast (e.g., for transmitting news) where zapping, i.e. stream changing, delay is important.

In the IPTV scenario, traffic is transmitted over managed (QoS-based) networks. Typical content used in this application is news, movies, cartoons, series, TV shows, etc. One important requirement for both groups is Random access to pictures, i.e. random access period (RAP) should be kept small enough (approximately, 1-5 seconds). Optional requirements are as follows:

- Temporal (frame-rate) scalability;

- Resolution and quality (SNR) scalability.

For this application, typical values of resolutions, frame-rates, and RAPs are presented in Table 4.

Resolution *	Frame-rate, fps	PAM
2160p (4K),3840x2160	24/1.001, 24, 25,	RA
1080p, 1920x1080	30/1.001, 30, 50,	RA
1080i, 1920x1080*	60/1.001, 60, 100,	RA
720p, 1280x720	120/1.001, 120	RA
576p (EDTV), 720x576	The set of frame-rates	RA
576i (SDTV), 720x576*	presented in this table	RA
480p (EDTV), 720x480	is taken from Table 2	RA
480i (SDTV), 720x480*	in [1]	RA

Table 4. IPTV: typical values of resolutions, frame-rates, and RAPs

NB *: Interlaced content can be handled at the higher system level and not necessarily by using specialized video coding tools. It is included in this table only for the sake of completeness as most video content today is in progressive format.

3.3. Video conferencing

This is a form of video connection over the Internet. This form allows users to establish connections to two or more people by two-way video and audio transmission for communication in real-time. For this application, both stationary and mobile devices can be used. The main requirements are as follows:

- Delay should be kept as low as possible (the preferable and maximum end-to-end delay values should be less than 100 ms [9] and 320 ms [2], respectively);
- Temporal (frame-rate) scalability;
- Error robustness.

Support of resolution and quality (SNR) scalability is highly desirable. For this application, typical values of resolutions, frame-rates, and RAPs are presented in Table 5.

Resolution	Frame-rate, fps	PAM
1080p, 1920x1080	15, 30	FIZD
720p, 1280x720	30, 60	FIZD
4CIF, 704x576	30, 60	FIZD
4SIF, 704x480	30, 60	FIZD
VGA, 640x480	30, 60	FIZD
360p, 640x360	30, 60	FIZD

Table 5. Video conferencing: typical values of resolutions, frame-rates, and RAPs

3.4. Video sharing

This is a service that allows people to upload and share video data (using live streaming or not) and to watch them. It is also known as video hosting. A typical User-generated Content (UGC) scenario for this application is to capture video using mobile cameras such as

GoPro or cameras integrated into smartphones (amateur video). The main requirements are as follows:

- Random access to pictures for downloaded video data;
- Temporal (frame-rate) scalability;
- Error robustness.

Support of resolution and quality (SNR) scalability is highly desirable. For this application, typical values of resolutions, frame-rates, and RAPs are presented in Table 6.

Resolution	Frame-rate, fps	PAM
2160p (4K),3840x2160	24, 25, 30, 48, 50, 60	RA
1440p (2K),2560x1440	24, 25, 30, 48, 50, 60	RA
1080p, 1920x1080	24, 25, 30, 48, 50, 60	RA
720p, 1280x720	24, 25, 30, 48, 50, 60	RA
480p, 854x480	24, 25, 30, 48, 50, 60	RA
360p, 640x360	24, 25, 30, 48, 50, 60	RA

Table 6. Video sharing: typical values of resolutions, frame-rates [10], and RAPs

3.5. Screencasting

This is a service that allows users to record and distribute computer desktop screen output. This service requires efficient compression of computer-generated content with high visual quality up to visually and mathematically (numerically) lossless [11]. Currently, this application includes business presentations (powerpoint, word documents, email messages, etc.), animation (cartoons), gaming content, data visualization, i.e. such type of content that is characterized by fast motion, rotation, smooth shade, 3D effect, highly saturated colors with full resolution, clear textures and sharp edges with distinct colors [11]), virtual desktop infrastructure (VDI), screen/desktop sharing and collaboration, supervisory control and data acquisition (SCADA) display, automotive/navigation display, cloud gaming, factory

automation display, wireless display, display wall, digital operating room (DiOR), etc. For this application, an important requirement is the support of low-delay configurations with zero structural delay, a wide range of video formats (e.g., RGB) in addition to YCbCr 4:2:0 and YCbCr 4:4:4 [11]. For this application, typical values of resolutions, frame-rates, and RAPs are presented in Table 7.

Resolution	Frame-rate, fps	PAM
Input color format: RGB 4:4:4		
5k, 5120x2880	15, 30, 60	AI, RA, FIZD
4k, 3840x2160	15, 30, 60	AI, RA, FIZD
WQXGA, 2560x1600	15, 30, 60	AI, RA, FIZD
WUXGA, 1920x1200	15, 30, 60	AI, RA, FIZD
WSXGA+, 1680x1050	15, 30, 60	AI, RA, FIZD
WXGA, 1280x800	15, 30, 60	AI, RA, FIZD
XGA, 1024x768	15, 30, 60	AI, RA, FIZD
SVGA, 800x600	15, 30, 60	AI, RA, FIZD
VGA, 640x480	15, 30, 60	AI, RA, FIZD
Input color format: YCbCr 4:4:4		
5k, 5120x2880	15, 30, 60	AI, RA, FIZD
4k, 3840x2160	15, 30, 60	AI, RA, FIZD
1440p (2K), 2560x1440	15, 30, 60	AI, RA, FIZD
1080p, 1920x1080	15, 30, 60	AI, RA, FIZD
720p, 1280x720	15, 30, 60	AI, RA, FIZD

Table 7. Screencasting for RGB and YCbCr 4:4:4 format: typical values of resolutions, frame-rates, and RAPs

3.6. Game streaming

This is a service that provides game content over the Internet to different local devices such as notebooks, gaming tablets, etc. In this category of applications, server renders 3D games in cloud server, and streams the game to any device with a wired or wireless broadband connection [12]. There are low latency requirements for transmitting user interactions and receiving game data in less than a turn-around delay of 100 ms. This allows anyone to play (or resume) full featured games from anywhere in the Internet [12]. An example of this application is Nvidia Grid [12]. Another category application is broadcast of video games played by people over the Internet in real time or for later viewing [12]. There are many companies such as Twitch, YY in China enable game broadcasting [12]. Games typically contain a lot of sharp edges and large motion [12]. The main requirements are as follows:

- Random access to pictures for game broadcasting;
- Temporal (frame-rate) scalability;
- Error robustness.

Support of resolution and quality (SNR) scalability is highly desirable. For this application, typical values of resolutions, frame-rates, and RAPs are similar to ones presented in Table 5.

3.7. Video monitoring / surveillance

This is a type of live broadcasting over IP-based networks. Video streams are sent to many receivers at the same time. A new receiver may connect to the stream at an arbitrary moment, so random access period should be kept small enough (approximately, ~1-5 seconds). Data are transmitted publicly in the case of video monitoring and privately in the case of video surveillance, respectively. For IP-cameras that have to capture, process and encode video data, complexity including computational and hardware complexity as well as memory bandwidth should be kept low to allow real-time processing. In addition, support of high dynamic range and a monochrome mode (e.g., for infrared cameras) as well as resolution and quality (SNR) scalability is an essential requirement for video surveillance. In some use-cases, high video signal fidelity is required even after lossy compression. Typical values of resolutions, frame-rates, and RAPs for video monitoring / surveillance applications are presented in Table 8.

Resolution	Frame-rate, fps	PAM
2160p (4K),3840x2160	12, 25, 30	RA, FIZD
5Mpixels, 2560x1920	12, 25, 30	RA, FIZD
1080p, 1920x1080	25, 30	RA, FIZD
1.3Mpixels, 1280x960	25, 30	RA, FIZD
720p, 1280x720	25, 30	RA, FIZD
SVGA, 800x600	25, 30	RA, FIZD

Table 8. Video monitoring / surveillance: typical values of resolutions, frame-rates, and RAPs

4. Requirements

Taking the requirements discussed above for specific video applications, this chapter proposes requirements for an internet video codec.

4.1. General requirements

4.1.1. The most basic requirement is coding efficiency, i.e. compression performance on both "easy" and "difficult" content for applications and use cases in Section 2. The codec should provide higher coding efficiency over state-of-the-art video codecs such as HEVC/H.265 and VP9, at least by 25% in accordance with the methodology described in Section 4.1 of this document. For higher resolutions, the coding efficiency improvements are expected to be higher than for lower resolutions.

4.1.2. Good quality specification and well-defined profiles and levels are required to enable device interoperability and facilitate decoder implementations. A profile consists of a subset of entire bitstream syntax elements and consequently it also defines the necessary tools for decoding a conforming bitstream of that profile. A level imposes a set of numerical limits to the values of some syntax elements. An example of codec levels to be supported is presented in Table 9. An actual level definition should include constraints on features that impact the decoder complexity. For example, these features might be as follows: maximum bit-rate, line buffer size, memory usage, etc.

Level	Example picture resolution at highest frame rate
1	128x96(12,288*)@30.0 176x144(25,344*)@15.0
2	352x288(101,376*)@30.0
3	352x288(101,376*)@60.0 640x360(230,400*)@30.0
4	640x360(230,400*)@60.0 960x540(518,400*)@30.0
5	720x576(414,720*)@75.0 960x540(518,400*)@60.0 1280x720(921,600*)@30.0
6	1,280x720(921,600*)@68.0 2,048x1,080(2,211,840*)@30.0
7	1,280x720(921,600*)@120.0 2,048x1,080(2,211,840*)@60.0
8	1,920x1,080(2,073,600*)@120.0 3,840x2,160(8,294,400*)@30.0 4,096x2,160(8,847,360*)@30.0
9	1,920x1,080(2,073,600*)@250.0 4,096x2,160(8,847,360*)@60.0
10	1,920x1,080(2,073,600*)@300.0 4,096x2,160(8,847,360*)@120.0
11	3,840x2,160(8,294,400*)@120.0 8,192x4,320(35,389,440*)@30.0
12	3,840x2,160(8,294,400*)@250.0 8,192x4,320(35,389,440*)@60.0
13	3,840x2,160(8,294,400*)@300.0 8,192x4,320(35,389,440*)@120.0

Table 9. Codec levels

NB *: The quantities of pixels are presented for such applications where a picture can have an arbitrary size (e.g., screencasting)

4.1.3. Bitstream syntax should allow extensibility and backward compatibility. New features can be supported easily by using metadata (e.g., such as SEI messages, VUI, headers) without affecting the bitstream compatibility with legacy decoders. A newer version of the decoder shall be able to play bitstreams of an older version of the same or lower profile and level.

4.1.4. A bitstream should have a model that allows easy parsing and identification of the sample components (such as ISO/IEC14496-10, Annex B or ISO/IEC 14496-15). In particular, information needed for packet handling (e.g., frame type) should not require parsing anything below the header level.

4.1.5. Perceptual quality tools (such as adaptive QP and quantization matrices) should be supported by the codec bit-stream.

4.1.6. The codec specification shall define a buffer model such as hypothetical reference decoder (HRD).

4.1.7. Specifications providing integration with system and delivery layers should be developed.

4.2. Basic requirements

4.2.1. Input source formats:

- Bit depth: 8- and 10-bits (up to 12-bits for a high profile) per color component;
- Color sampling formats:
  - YCbCr 4:2:0;
  - YCbCr 4:4:4, YCbCr 4:2:2 and YCbCr 4:0:0 (preferably in different profile(s)).
- For profiles with bit depth of 10 bits per sample or higher, support of high dynamic range and wide color gamut.
- Support of arbitrary resolution according to the level constraints for such applications where a picture can have an arbitrary size (e.g., in screencasting).
- Exemplary input source formats for codec profiles are shown in Table 10.

Profile	Bit-depths per color component	Color sampling formats
1	8 and 10	4:0:0 and 4:2:0
2	8 and 10	4:0:0, 4:2:0 and 4:4:4
3	8, 10 and 12	4:0:0, 4:2:0, 4:2:2 and 4:4:4

Table 10. Exemplary input source formats for codec profiles

4.2.2. Coding delay:

- Support of configurations with zero structural delay also referred to as "low-delay" configurations.

  . Note 1: end-to-end delay should be up to 320 ms [2] but its preferable value should be less than 100 ms [9]

- Support of efficient random access point encoding (such as intra coding and resetting of context variables) as well as efficient switching between multiple quality representations.

- Support of configurations with non-zero structural delay (such as out-of-order or multi-pass encoding) for applications without low-delay requirements if such configurations provide additional compression efficiency improvements.

4.2.3. Complexity:

- Feasible real-time implementation of both an encoder and a decoder supporting a chosen subset of tools for hardware and software implementation on a wide range of state-of-the-art platforms. The real-time encoder tools subset should provide meaningful improvement in compression efficiency at reasonable complexity of hardware and software encoder implementations as compared to real-time implementations of state-of-the-art video compression technologies such as HEVC/H.265 and VP9.

- High-complexity software encoder implementations used by offline encoding applications can have 10x or more complexity increase compared to state-of-the-art video compression technologies such as HEVC/H.265 and VP9.

4.2.4. Scalability:

- Temporal (frame-rate) scalability should be supported.

4.2.5. Error resilience:

- Error resilience tools that are complementary to the error protection mechanisms implemented on transport level should be supported.
- The codec should support mechanisms that facilitate packetization of a bitstream for common network protocols.
- Packetization mechanisms should enable frame-level error recovery by means of retransmission or error concealment.
- The codec should support effective mechanisms for allowing decoding and reconstruction of significant parts of pictures in the event that parts of the picture data are lost in transmission.
- The bitstream specification shall support independently decodable sub-frame units similar to slices or independent tiles. It shall be possible for the encoder to restrict the bit-stream to allow parsing of the bit-stream after a packet-loss and to communicate it to the decoder.

4.3. Optional requirements

4.3.1. Input source formats

- Bit depth: up to 16-bits per color component.
- Color sampling formats: RGB 4:4:4.
- Auxiliary channel (e.g., alpha channel) support.

4.3.2. Scalability:

- Resolution and quality (SNR) scalability that provide low compression efficiency penalty (up to 5% of BD-rate [13] increase per layer with reasonable increase of both computational and hardware complexity) can be supported in the main profile of the codec being developed by the NETVC WG. Otherwise, a separate profile is needed to support these types of scalability.

- Computational complexity scalability(i.e. computational complexity is decreasing along with degrading picture quality) is desirable.

4.3.3. Complexity:

Tools that enable parallel processing (e.g., slices, tiles, wave front propagation processing) at both encoder and decoder sides are highly desirable for many applications.

- High-level multi-core parallelism: encoder and decoder operation, especially entropy encoding and decoding, should allow multiple frames or sub-frame regions (e.g. 1D slices, 2D tiles, or partitions) to be processed concurrently, either independently or with deterministic dependencies that can be efficiently pipelined

- Low-level instruction set parallelism: favor algorithms that are SIMD/GPU friendly over inherently serial algorithms

4.3.4. Coding efficiency

Compression efficiency on noisy content, content with film grain, computer generated content, and low resolution materials is desirable.

5. Evaluation methodology

As shown in Fig.1, compression performance testing is performed in 3 overlapped ranges that encompass 10 different bitrate values:

- Low bitrate range (LBR) is the range that contains the 4 lowest bitrates of the 10 specified bitrates (1 of the 4 bitrate values is shared with the neighboring range);

- Medium bitrate range (MBR) is the range that contains the 4 medium bitrates of the 10 specified bitrates (2 of the 4 bitrate values are shared with the neighboring ranges);

- High bitrate range (HBR) is the range that contains the 4 highest bitrates of the 10 specified bitrates (1 of the 4 bitrate values is shared with the neighboring range).

Initially, for the codec selected as a reference one (e.g., HEVC or VP9), a set of 10 QP (quantization parameter) values should be specified in [14] and corresponding quality values should be calculated. In Fig.1, QP and quality values are denoted as QP0, QP1, QP2,..., QP8, QP9 and Q0, Q1, Q2,..., Q8, Q9, respectively. To

guarantee the overlaps of quality levels between the bitrate ranges of the reference and tested codecs, a quality alignment procedure should be performed for each range's outermost (left- and rightmost) quality levels Qk of the reference codec (i.e. for Q0, Q3, Q6, and Q9) and the quality levels Q'k (i.e. Q'0, Q'3, Q'6, and Q'9) of the tested codec. Thus, these quality levels Q'k and, hence, the corresponding QP value QP'k (i.e. QP'0, QP'3, QP'6, and QP'9) of the tested codec should be selected using the following formulas:

$$Q'k = \min_{i \in R} \{ abs(Q'i - Qk) \},$$

$$QP'k = \operatorname{argmin}_{i \in R} \{ abs(Q'i(QP'i) - Qk(QPk)) \},$$

where R is the range of the QP indexes of the tested codec, i.e. the candidate Internet video codec. The inner quality levels (i.e. Q'1, Q'2, Q'4, Q'5, Q'7, and Q'8) as well as their corresponding QP values of each range (i.e. QP'1, QP'2, QP'4, QP'5, QP'7, and QP'8) should be as equidistantly spaced as possible between the left- and rightmost quality levels without explicitly mapping their values using the above described procedure.

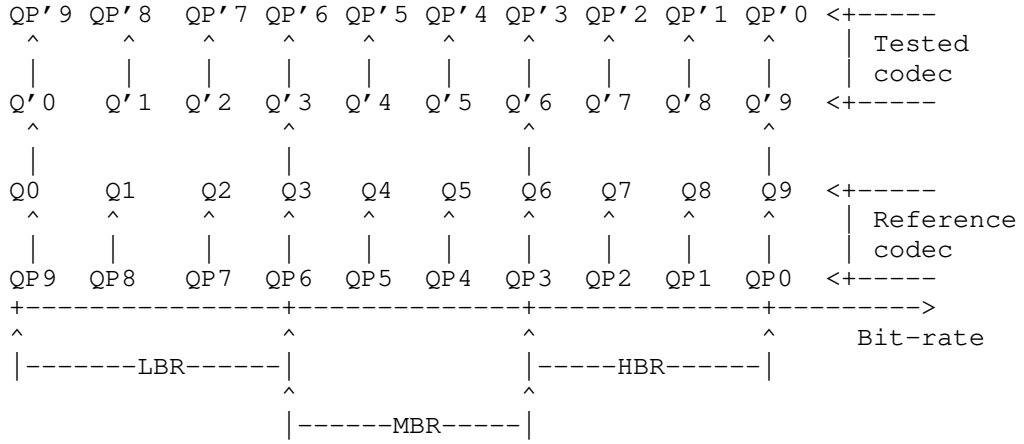

Figure 1 Quality/QP alignment for compression performance evaluation

Since the QP mapping results may vary for different sequences, eventually, this quality alignment procedure needs to be separately performed for each quality assessment index and each sequence used for codec performance evaluation to fulfill the above described requirements.

To assess the quality of output (decoded) sequences, two indexes, PSNR [3] and MS-SSIM [3,15] are separately computed. In the case of the YCbCr color format, PSNR should be calculated for each color plane whereas MS-SSIM is calculated for luma channel only. In the case of the RGB color format, both metrics are computed for R, G and B channels. Thus, for each sequence, 30 RD-points for PSNR (i.e. three RD-curves, one for each channel) and 10 RD-points for MS-SSIM (i.e. one RD-curve, for luma channel only) should be calculated in the case of YCbCr. If content is encoded as RGB, 60 RD-points (30 for PSNR and 30 for MS-SSIM) should be calculated, i.e. three RD-curves (one for each channel) are computed for PSNR as well as three RD-curves (one for each channel) for MS-SSIM.

Finally, to obtain an integral estimation, BD-rate savings [13] should be computed for each range and each quality index. In addition, average values over all the 3 ranges should be provided for both PSNR and MS-SSIM. A list of video sequences that should be used for testing as well as the 10 QP values for the reference codec are defined in [14]. Testing processes should use the information on the codec applications presented in this document. As the reference for evaluation, state-of-the-art video codecs such as HEVC/H.265 [4,5] or VP9 must be used. The reference source code of the HEVC/H.265 codec can be found at [6]. The HEVC/H.265 codec must be configured according to [16] and Table 11.

Intra-period, second	HEVC/H.265 encoding mode according to [16]
AI	Intra Main or Intra Main10
RA	Random access Main or Random access Main10
FIZD	Low delay Main or Low delay Main10

Table 11. Intra-periods for different HEVC/H.265 encoding modes according to [16]

According to the coding efficiency requirement described in Section 3.1.1, BD-rate savings calculated for each color plane and averaged for all the video sequences used to test the NETVC codec should be, at least,

o 25% if calculated over the whole bitrate range;

o 15% if calculated for each bitrate subrange (LBR, MBR, HBR).

Since values of the two objective metrics (PSNR and MS-SSIM) are available for some color planes, each value should meet these coding efficiency requirements, i.e. the final BD-rate saving denoted as S is calculated for a given color plane as follows:

S = min { S_psnr, S_ms-ssim },

where S_psnr and S_ms-ssim are BD-rate savings calculated for the given color plane using PSNR and MS-SSIM metrics, respectively.

In addition to the objective quality measures defined above, subjective evaluation must also be performed for the final NETVC codec adoption. For subjective tests, the MOS-based evaluation procedure must be used as described in section 2.1 of [3]. For perception-oriented tools that primarily impact subjective quality, additional tests may also be individually assigned even for intermediate evaluation, subject to a decision of the NETVC WG.

6. Security Considerations

This document itself does not address any security considerations. However, it is worth noting that a codec implementation (for both an encoder and a decoder) should take into consideration the worst-case computational complexity, memory bandwidth, and physical memory size needed to processes the potentially untrusted input (e.g., the decoded pictures used as references).

7. IANA Considerations

This document has no IANA actions.

8. References

8.1. Normative References

[1] Recommendation ITU-R BT.2020-2: Parameter values for ultra-high definition television systems for production and international programme exchange, 2015.

[2] Recommendation ITU-T G.1091: Quality of Experience requirements for telepresence services, 2014.

[3] ISO/IEC PDTR 29170-1: Information technology -- Advanced image coding and evaluation methodologies -- Part 1: Guidelines for codec evaluation.

[4] ISO/IEC 23008-2:2015. Information technology -- High efficiency coding and media delivery in heterogeneous environments -- Part 2: High efficiency video coding

[5] Recommendation ITU-T H.265: High efficiency video coding, 2013.

[6] High Efficiency Video Coding (HEVC) reference software (HEVC Test Model also known as HM) at the web-site of Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute (HHI): https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/

8.2. Informative References

[7] Definition of the term "high dynamic range imaging" at the web-site of Federal Agencies Digital Guidelines Initiative: http://www.digitizationguidelines.gov/term.php?term=highdynamicrangeimaging

[8] Definition of the term "compression, visually lossless" at the web-site of Federal Agencies Digital Guidelines Initiative: http://www.digitizationguidelines.gov/term.php?term=compressionvisuallylossless

[9] S. Wenger, "The case for scalability support in version 1 of Future Video Coding," Document COM 16-C 988 R1-E of ITU-T Video Coding Experts Group (ITU-T Q.6/SG 16), Geneva, Switzerland, September 2015.

[10] "Recommended upload encoding settings (Advanced)" for the YouTube video-sharing service: https://support.google.com/youtube/answer/1722171?hl=en

[11] H. Yu, K. McCann, R. Cohen, and P. Amon, "Requirements for future extensions of HEVC in coding screen content", Document N14174 of Moving Picture Experts Group (ISO/IEC JTC 1/SC 29/ WG 11), San Jose, USA, January 2014.

[12] Manindra Parhy, "Game streaming requirement for Future Video Coding," Document N36771 of ISO/IEC Moving Picture Experts Group (ISO/IEC JTC 1/SC 29/WG 11), Warsaw, Poland, June 2015.

[13] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," Document VCEG-M33 of ITU-T Video Coding Experts Group (ITU-T Q.6/SG 16), Austin, Texas, USA, April 2001.

[14] T. Daede, A. Norkin, and I. Brailovskiy, "Video Codec Testing and Quality Measurement", draft-ietf-netvc-testing-08(work in progress), January 2019, p.23.

[15] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multi-scale structural similarity for image quality assessment," Invited Paper, IEEE Asilomar Conference on Signals, Systems and Computers, Nov. 2003, Vol. 2, pp. 1398-1402.

[16] F. Bossen, "Common test conditions and software reference configurations," Document JCTVC-L1100 of Joint Collaborative Team on Video Coding (JCT-VC) of the ITU-T Video Coding Experts Group (ITU-T Q.6/SG 16) and ISO/IEC Moving Picture Experts Group (ISO/IEC JTC 1/SC 29/WG 11), Geneva, Switzerland, January 2013.

9. Acknowledgments

The authors would like to thank Mr. Paul Coverdale, Mr. Vasily Rufitskiy, and Dr. Jianle Chen for many useful discussions on this document and their help while preparing it as well as Mr. Mo Zanaty, Dr. Minhua Zhou, Dr. Ali Begen, Mr. Thomas Daede, Mr. Adam Roach, Dr. Thomas Davies, Mr. Jonathan Lennox, Dr. Timothy Terriberry, Mr. Peter Thatcher, Dr. Jean-Marc Valin, Mr. Roman Danyliw, Mr. Jack Moffitt, Mr. Greg Coppa, and Mr. Andrew Krupiczka for their valuable comments on different revisions of this document.

This document was prepared using 2-Word-v2.0.template.dot.

Authors' Addresses

Alexey Filippov
Huawei Technologies

Email: alexey.filippov@huawei.com

Andrey Norkin
Netflix

Email: anorkin@netflix.com

Jose Roberto Alvarez
Huawei Technologies

Email: j.alvarez@ieee.org

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 3, 2020

T. Daede
Mozilla
A. Norkin
Netflix
I. Brailovskiy
Amazon Lab126

January 31, 2020

Video Codec Testing and Quality Measurement
draft-ietf-netvc-testing-09

Abstract

This document describes guidelines and procedures for evaluating a video codec. This covers subjective and objective tests, test conditions, and materials used for the test.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 3, 2020.

Copyright Notice

Table of Contents

1. Introduction

When developing a video codec, changes and additions to the codec need to be decided based on their performance tradeoffs. In addition, measurements are needed to determine when the codec has met its performance goals. This document specifies how the tests are to be carried about to ensure valid comparisons when evaluating changes under consideration. Authors of features or changes should provide the results of the appropriate test when proposing codec modifications.

2. Subjective quality tests

Subjective testing uses human viewers to rate and compare the quality of videos. It is the preferable method of testing video codecs.

Subjective testing results take priority over objective testing results, when available. Subjective testing is recommended especially when taking advantage of psychovisual effects that may not be well represented by objective metrics, or when different objective metrics disagree.

Selection of a testing methodology depends on the feature being tested and the resources available. Test methodologies are presented in order of increasing accuracy and cost.

Testing relies on the resources of participants. If a participant requires a subjective test for a particular feature or improvement, they are responsible for ensuring that resources are available. This ensures that only important tests be done; in particular, the tests that are important to participants.

Subjective tests should use the same operating points as the objective tests.

2.1. Still Image Pair Comparison

A simple way to determine superiority of one compressed image is to visually compare two compressed images, and have the viewer judge which one has a higher quality. For example, this test may be suitable for an intra de-ringing filter, but not for a new inter prediction mode. For this test, the two compressed images should have similar compressed file sizes, with one image being no more than 5% larger than the other. In addition, at least 5 different images should be compared.

Once testing is complete, a p-value can be computed using the binomial test. A significant result should have a resulting p-value less than or equal to 0.5. For example:

p_value = binom_test(a,a+b)

where a is the number of votes for one video, b is the number of votes for the second video, and binom_test(x,y) returns the binomial PMF (probability mass function) with x observed tests, y total tests, and expected probability 0.5.

If ties are allowed to be reported, then the equation is modified:

p_value = binom_test(a+floor(t/2),a+b+t)

where t is the number of tie votes.

Still image pair comparison is used for rapid comparisons during development - the viewer may be either a developer or user, for example. As the results are only relative, it is effective even with an inconsistent viewing environment. Because this test only uses still images (keyframes), this is only suitable for changes with similar or no effect on inter frames.

2.2. Video Pair Comparison

The still image pair comparison method can be modified to also compare vidoes. This is necessary when making changes with temporal effects, such as changes to inter-frame prediction. Video pair comparisons follow the same procedure as still images. Videos used for testing should be limited to 10 seconds in length, and can be rewatched an unlimited number of times.

2.3. Mean Opinion Score

A Mean Opinion Score (MOS) viewing test is the preferred method of evaluating the quality. The subjective test should be performed as either consecutively showing the video sequences on one screen or on two screens located side-by-side. The testing procedure should normally follow rules described in [BT500] and be performed with non-expert test subjects. The result of the test will be (depending on the test procedure) mean opinion scores (MOS) or differential mean opinion scores (DMOS). Confidence intervals are also calculated to judge whether the difference between two encodings is statistically significant. In certain cases, a viewing test with expert test subjects can be performed, for example if a test should evaluate technologies with similar performance with respect to a particular artifact (e.g. loop filters or motion prediction). Unlike pair

comparisions, a MOS test requires a consistent testing environment. This means that for large scale or distributed tests, pair comparisons are preferred.

3. Objective Metrics

Objective metrics are used in place of subjective metrics for easy and repeatable experiments. Most objective metrics have been designed to correlate with subjective scores.

The following descriptions give an overview of the operation of each of the metrics. Because implementation details can sometimes vary, the exact implementation is specified in C in the Daala tools repository [DAALA-GIT]. Implementations of metrics must directly support the input's resolution, bit depth, and sampling format.

Unless otherwise specified, all of the metrics described below only apply to the luma plane, individually by frame. When applied to the video, the scores of each frame are averaged to create the final score.

Codecs must output the same resolution, bit depth, and sampling format as the input.

3.1. Overall PSNR

PSNR is a traditional signal quality metric, measured in decibels. It is directly drived from mean square error (MSE), or its square root (RMSE). The formula used is:

20 * log10 (MAX / RMSE)

or, equivalently:

10 * log10 (MAX^2 / MSE)

where the error is computed over all the pixels in the video, which is the method used in the dump_psnr.c reference implementation.

This metric may be applied to both the luma and chroma planes, with all planes reported separately.

3.2. Frame-averaged PSNR

PSNR can also be calculated per-frame, and then the values averaged together. This is reported in the same way as overall PSNR.

3.3. PSNR-HVS-M

The PSNR-HVS [PSNRHVS] metric performs a DCT transform of 8x8 blocks of the image, weights the coefficients, and then calculates the PSNR of those coefficients. Several different sets of weights have been considered. The weights used by the dump_pnsrhvs.c tool in the Daala repository have been found to be the best match to real MOS scores.

3.4. SSIM

SSIM (Structural Similarity Image Metric) is a still image quality metric introduced in 2004 [SSIM]. It computes a score for each individual pixel, using a window of neighboring pixels. These scores can then be averaged to produce a global score for the entire image. The original paper produces scores ranging between 0 and 1.

To linearize the metric for BD-Rate computation, the score is converted into a nonlinear decibel scale:

-10 * log10 (1 - SSIM)

3.5. Multi-Scale SSIM

Multi-Scale SSIM is SSIM extended to multiple window sizes [MSSSIM]. The metric score is converted to decibels in the same way as SSIM.

3.6. CIEDE2000

CIEDE2000 is a metric based on CIEDE color distances [CIEDE2000]. It generates a single score taking into account all three chroma planes. It does not take into consideration any structural similarity or other psychovisual effects.

3.7. VMAF

Video Multi-method Assessment Fusion (VMAF) is a full-reference perceptual video quality metric that aims to approximate human perception of video quality [VMAF]. This metric is focused on quality degradation due to compression and rescaling. VMAF estimates the perceived quality score by computing scores from multiple quality assessment algorithms, and fusing them using a support vector machine (SVM). Currently, three image fidelity metrics and one temporal signal have been chosen as features to the SVM, namely Anti-noise SNR (ANSNR), Detail Loss Measure (DLM), Visual Information Fidelity (VIF), and the mean co-located pixel difference of a frame with respect to the previous frame.

The quality score from VMAF is used directly to calculate BD-Rate, without any conversions.

4. Comparing and Interpreting Results

4.1. Graphing

When displayed on a graph, bitrate is shown on the X axis, and the quality metric is on the Y axis. For publication, the X axis should be linear. The Y axis metric should be plotted in decibels. If the quality metric does not natively report quality in decibels, it should be converted as described in the previous section.

4.2. BD-Rate

The Bjontegaard rate difference, also known as BD-rate, allows the measurement of the bitrate reduction offered by a codec or codec feature, while maintaining the same quality as measured by objective metrics. The rate change is computed as the average percent difference in rate over a range of qualities. Metric score ranges are not static - they are calculated either from a range of bitrates of the reference codec, or from quantizers of a third, anchor codec. Given a reference codec and test codec, BD-rate values are calculated as follows:

- Rate/distortion points are calculated for the reference and test codec.
  - At least four points must be computed. These points should be the same quantizers when comparing two versions of the same codec.
  - Additional points outside of the range should be discarded.
- The rates are converted into log-rates.
- A piecewise cubic hermite interpolating polynomial is fit to the points for each codec to produce functions of log-rate in terms of distortion.
- Metric score ranges are computed:
  - If comparing two versions of the same codec, the overlap is the intersection of the two curves, bound by the chosen quantizer points.
  - If comparing dissimilar codecs, a third anchor codec's metric scores at fixed quantizers are used directly as the bounds.

- The log-rate is numerically integrated over the metric range for each curve, using at least 1000 samples and trapezoidal integration.

- The resulting integrated log-rates are converted back into linear rate, and then the percent difference is calculated from the reference to the test codec.

4.3. Ranges

For individual feature changes in libaom or libvpx, the overlap BD-Rate method with quantizers 20, 32, 43, and 55 must be used.

For the final evaluation described in [I-D.ietf-netvc-requirements], the quantizers used are 20, 24, 28, 32, 36, 39, 43, 47, 51, and 55.

5. Test Sequences

5.1. Sources

Lossless test clips are preferred for most tests, because the structure of compression artifacts in already-compressed clips may introduce extra noise in the test results. However, a large amount of content on the internet needs to be recompressed at least once, so some sources of this nature are useful. The encoder should run at the same bit depth as the original source. In addition, metrics need to support operation at high bit depth. If one or more codecs in a comparison do not support high bit depth, sources need to be converted once before entering the encoder.

5.2. Test Sets

Sources are divided into several categories to test different scenarios the codec will be required to operate in. For easier comparison, all videos in each set should have the same color subsampling, same resolution, and same number of frames. In addition, all test videos must be publicly available for testing use, to allow for reproducibility of results. All current test sets are available for download [TESTSEQUENCES].

Test sequences should be downloaded in whole. They should not be recreated from the original sources.

Each clip is labeled with its resolution, bit depth, color subsampling, and length.

5.2.1. regression-1

This test set is used for basic regression testing. It contains a very small number of clips.

- kirlandvga (640x360, 8bit, 4:2:0, 300 frames)
- FourPeople (1280x720, 8bit, 4:2:0, 60 frames)
- Narrarator (4096x2160, 10bit, 4:2:0, 15 frames)
- CSGO (1920x1080, 8bit, 4:4:4 60 frames)

5.2.2. objective-2-slow

This test set is a comprehensive test set, grouped by resolution. These test clips were created from originals at [TESTSEQUENCES]. They have been scaled and cropped to match the resolution of their category. This test set requires a codec that supports both 8 and 10 bit video.

4096x2160, 4:2:0, 60 frames:

- Netflix_BarScene_4096x2160_60fps_10bit_420_60f
- Netflix_BoxingPractice_4096x2160_60fps_10bit_420_60f
- Netflix_Dancers_4096x2160_60fps_10bit_420_60f
- Netflix_Narrator_4096x2160_60fps_10bit_420_60f
- Netflix_RitualDance_4096x2160_60fps_10bit_420_60f
- Netflix_ToddlerFountain_4096x2160_60fps_10bit_420_60f
- Netflix_WindAndNature_4096x2160_60fps_10bit_420_60f
- street_hdr_amazon_2160p

1920x1080, 4:2:0, 60 frames:

- aspen_1080p_60f
- crowd_run_1080p50_60f
- ducks_take_off_1080p50_60f
- guitar_hdr_amazon_1080p

- life_1080p30_60f
- Netflix_Aerial_1920x1080_60fps_8bit_420_60f
- Netflix_Boat_1920x1080_60fps_8bit_420_60f
- Netflix_Crosswalk_1920x1080_60fps_8bit_420_60f
- Netflix_FoodMarket_1920x1080_60fps_8bit_420_60f
- Netflix_PierSeaside_1920x1080_60fps_8bit_420_60f
- Netflix_SquareAndTimelapse_1920x1080_60fps_8bit_420_60f
- Netflix_TunnelFlag_1920x1080_60fps_8bit_420_60f
- old_town_cross_1080p50_60f
- pan_hdr_amazon_1080p
- park_joy_1080p50_60f
- pedestrian_area_1080p25_60f
- rush_field_cuts_1080p_60f
- rush_hour_1080p25_60f
- seaplane_hdr_amazon_1080p
- station2_1080p25_60f
- touchdown_pass_1080p_60f

1280x720, 4:2:0, 120 frames:

- boat_hdr_amazon_720p
- dark720p_120f
- FourPeople_1280x720_60_120f
- gipsrestat720p_120f
- Johnny_1280x720_60_120f
- KristenAndSara_1280x720_60_120f

- Netflix_DinnerScene_1280x720_60fps_8bit_420_120f
- Netflix_DrivingPOV_1280x720_60fps_8bit_420_120f
- Netflix_FoodMarket2_1280x720_60fps_8bit_420_120f
- Netflix_RollerCoaster_1280x720_60fps_8bit_420_120f
- Netflix_Tango_1280x720_60fps_8bit_420_120f
- rain_hdr_amazon_720p
- vidyo1_720p_60fps_120f
- vidyo3_720p_60fps_120f
- vidyo4_720p_60fps_120f

640x360, 4:2:0, 120 frames:

- blue_sky_360p_120f
- controlled_burn_640x360_120f
- desktop2360p_120f
- kirland360p_120f
- mmstationary360p_120f
- niklas360p_120f
- rain2_hdr_amazon_360p
- red_kayak_360p_120f
- riverbed_360p25_120f
- shields2_640x360_120f
- snow_mnt_640x360_120f
- speed_bag_640x360_120f
- stockholm_640x360_120f
- tacomanarrows360p_120f

- thaloundeskmtg360p_120f
- water_hdr_amazon_360p

426x240, 4:2:0, 120 frames:

- bqfree_240p_120f
- bqhighway_240p_120f
- bqzoom_240p_120f
- chairlift_240p_120f
- dirtbike_240p_120f
- mozzoom_240p_120f

1920x1080, 4:4:4 or 4:2:0, 60 frames:

- CSGO_60f.y4m
- DOTA2_60f_420.y4m
- MINECRAFT_60f_420.y4m
- STARCRAFT_60f_420.y4m
- EuroTruckSimulator2_60f.y4m
- Hearthstone_60f.y4m
- wikipedia_420.y4m
- pvq_slideshow.y4m

5.2.3. objective-2-fast

This test set is a strict subset of objective-2-slow. It is designed for faster runtime. This test set requires compiling with high bit depth support.

1920x1080, 4:2:0, 60 frames:

- aspen_1080p_60f
- ducks_take_off_1080p50_60f

- life_1080p30_60f
- Netflix_Aerial_1920x1080_60fps_8bit_420_60f
- Netflix_Boat_1920x1080_60fps_8bit_420_60f
- Netflix_FoodMarket_1920x1080_60fps_8bit_420_60f
- Netflix_PierSeaside_1920x1080_60fps_8bit_420_60f
- Netflix_SquareAndTimelapse_1920x1080_60fps_8bit_420_60f
- Netflix_TunnelFlag_1920x1080_60fps_8bit_420_60f
- rush_hour_1080p25_60f
- seaplane_hdr_amazon_1080p
- touchdown_pass_1080p_60f

1280x720, 4:2:0, 120 frames:

- boat_hdr_amazon_720p
- dark720p_120f
- gipsrestat720p_120f
- KristenAndSara_1280x720_60_120f
- Netflix_DrivingPOV_1280x720_60fps_8bit_420_60f
- Netflix_RollerCoaster_1280x720_60fps_8bit_420_60f
- vidyo1_720p_60fps_120f
- vidyo4_720p_60fps_120f

640x360, 4:2:0, 120 frames:

- blue_sky_360p_120f
- controlled_burn_640x360_120f
- kirland360p_120f
- niklas360p_120f

- rain2_hdr_amazon_360p
- red_kayak_360p_120f
- riverbed_360p25_120f
- shields2_640x360_120f
- speed_bag_640x360_120f
- thaloundeskmtg360p_120f

426x240, 4:2:0, 120 frames:

- bqfree_240p_120f
- bqzoom_240p_120f
- dirtbike_240p_120f

1290x1080, 4:2:0, 60 frames:

- DOTA2_60f_420.y4m
- MINECRAFT_60f_420.y4m
- STARCRAFT_60f_420.y4m
- wikipedia_420.y4m

5.2.4. objective-1.1

This test set is an old version of objective-2-slow.

4096x2160, 10bit, 4:2:0, 60 frames:

- Aerial (start frame 600)
- BarScene (start frame 120)
- Boat (start frame 0)
- BoxingPractice (start frame 0)
- Crosswalk (start frame 0)
- Dancers (start frame 120)

- FoodMarket
- Narrator
- PierSeaside
- RitualDance
- SquareAndTimelapse
- ToddlerFountain (start frame 120)
- TunnelFlag
- WindAndNature (start frame 120)

1920x1080, 8bit, 4:4:4, 60 frames:

- CSGO
- DOTA2
- EuroTruckSimulator2
- Hearthstone
- MINECRAFT
- STARCRAFT
- wikipedia
- pvq_slideshow

1920x1080, 8bit, 4:2:0, 60 frames:

- ducks_take_off
- life
- aspen
- crowd_run
- old_town_cross
- park_joy

- pedestrian_area
- rush_field_cuts
- rush_hour
- station2
- touchdown_pass

1280x720, 8bit, 4:2:0, 60 frames:

- Netflix_FoodMarket2
- Netflix_Tango
- DrivingPOV (start frame 120)
- DinnerScene (start frame 120)
- RollerCoaster (start frame 600)
- FourPeople
- Johnny
- KristenAndSara
- vidyo1
- vidyo3
- vidyo4
- dark720p
- gipsrecmotion720p
- gipsrestat720p
- controlled_burn
- stockholm
- speed_bag
- snow_mnt

- shields

640x360, 8bit, 4:2:0, 60 frames:

- red_kayak
- blue_sky
- riverbed
- thaloundeskmtgvga
- kirlandvga
- tacomanarrowsvga
- tacomascmvvga
- desktop2360p
- mmmovingvga
- mmstationaryvga
- niklasvga

5.2.5. objective-1-fast

This is an old version of objective-2-fast.

1920x1080, 8bit, 4:2:0, 60 frames:

- Aerial (start frame 600)
- Boat (start frame 0)
- Crosswalk (start frame 0)
- FoodMarket
- PierSeaside
- SquareAndTimelapse
- TunnelFlag

1920x1080, 8bit, 4:2:0, 60 frames:

- CSGO
- EuroTruckSimulator2
- MINECRAFT
- wikipedia

1920x1080, 8bit, 4:2:0, 60 frames:

- ducks_take_off
- aspen
- old_town_cross
- pedestrian_area
- rush_hour
- touchdown_pass

1280x720, 8bit, 4:2:0, 60 frames:

- Netflix_FoodMarket2
- DrivingPOV (start frame 120)
- RollerCoaster (start frame 600)
- Johnny
- vidyo1
- vidyo4
- gipsrecmotion720p
- speed_bag
- shields

640x360, 8bit, 4:2:0, 60 frames:

- red_kayak
- riverbed

- kirlandvga
- tacomascmvvga
- mmmovingvga
- niklasvga

5.3. Operating Points

Four operating modes are defined. High latency is intended for on demand streaming, one-to-many live streaming, and stored video. Low latency is intended for videoconferencing and remote access. Both of these modes come in CQP (constant quantizer parameter) and unconstrained variants. When testing still image sets, such as subset1, high latency CQP mode should be used.

5.3.1. Common settings

Encoders should be configured to their best settings when being compared against each other:

- av1: -codec=av1 -ivf -frame-parallel=0 -tile-columns=0 -cpu-used=0 -threads=1

5.3.2. High Latency CQP

High Latency CQP is used for evaluating incremental changes to a codec. This method is well suited to compare codecs with similar coding tools. It allows codec features with intrinsic frame delay.

- daala: -v=x -b 2
- vp9: -end-usage=q -cq-level=x -lag-in-frames=25 -auto-alt-ref=2
- av1: -end-usage=q -cq-level=x -auto-alt-ref=2

5.3.3. Low Latency CQP

Low Latency CQP is used for evaluating incremental changes to a codec. This method is well suited to compare codecs with similar coding tools. It requires the codec to be set for zero intrinsic frame delay.

- daala: -v=x
- av1: -end-usage=q -cq-level=x -lag-in-frames=0

5.3.4. Unconstrained High Latency

The encoder should be run at the best quality mode available, using the mode that will provide the best quality per bitrate (VBR or constant quality mode). Lookahead and/or two-pass are allowed, if supported. One parameter is provided to adjust bitrate, but the units are arbitrary. Example configurations follow:

- x264: --crf=x
- x265: --crf=x
- daala: -v=x -b 2
- av1: --end-usage=q --cq-level=x --lag-in-frames=25 --auto-alt-ref=2

5.3.5. Unconstrained Low Latency

The encoder should be run at the best quality mode available, using the mode that will provide the best quality per bitrate (VBR or constant quality mode), but no frame delay, buffering, or lookahead is allowed. One parameter is provided to adjust bitrate, but the units are arbitrary. Example configurations follow:

- x264: --crf-x --tune zerolatency
- x265: --crf=x --tune zerolatency
- daala: -v=x
- av1: --end-usage=q --cq-level=x --lag-in-frames=0

6. Automation

Frequent objective comparisons are extremely beneficial while developing a new codec. Several tools exist in order to automate the process of objective comparisons. The Compare-Codecs tool allows BD-rate curves to be generated for a wide variety of codecs [COMPARECODECS]. The Daala source repository contains a set of scripts that can be used to automate the various metrics used. In addition, these scripts can be run automatically utilizing distributed computers for fast results, with rd_tool [RD_TOOL]. This tool can be run via a web interface called AreWeCompressedYet [AWCY], or locally.

Because of computational constraints, several levels of testing are specified.

6.1. Regression tests

Regression tests run on a small number of short sequences - regression-test-1. The regression tests should include a number of various test conditions. The purpose of regression tests is to ensure bug fixes (and similar patches) do not negatively affect the performance. The anchor in regression tests is the previous revision of the codec in source control. Regression tests are run on both high and low latency CQP modes

6.2. Objective performance tests

Changes that are expected to affect the quality of encode or bitstream should run an objective performance test. The performance tests should be run on a wider number of sequences. The following data should be reported:

- Identifying information for the encoder used, such as the git commit hash.
- Command line options to the encoder, configure script, and anything else necessary to replicate the experiment.
- The name of the test set run (objective-1-fast)
- For both high and low latency CQP modes, and for each objective metric:
  - The BD-Rate score, in percent, for each clip.
  - The average of all BD-Rate scores, equally weighted, for each resolution category in the test set.
  - The average of all BD-Rate scores for all videos in all categories.

Normally, the encoder should always be run at the slowest, highest quality speed setting (cpu-used=0 in the case of AV1 and VP9). However, in the case of computation time, both the reference and changed encoder can be built with some options disabled. For AV1, -disable-ext_partition and -disable-ext_partition_types can be passed to the configure script to substantially speed up encoding, but the usage of these options must be reported in the test results.

6.3. Periodic tests

Periodic tests are run on a wide range of bitrates in order to gauge progress over time, as well as detect potential regressions missed by other tests.

7. IANA Considerations

This document does not require any IANA actions.

8. Security Considerations

This document describes the methodologies an procedures for qualitative testing, therefore does not iteself have implications for network of decoder security.

9. Informative References

[AWCY] Xiph.Org, "Are We Compressed Yet?", 2016, <https://arewecompressedyet.com/>.

[BT500] ITU-R, "Recommendation ITU-R BT.500-13", 2012, <https://www.itu.int/dms_pubrec/itu-r/rec/bt/R-REC-BT.500-13-201201-I!!PDF-E.pdf>.

[CIEDE2000] Yang, Y., Ming, J., and N. Yu, "Color Image Quality Assessment Based on CIEDE2000", 2012, <http://dx.doi.org/10.1155/2012/273723>.

[COMPARECODECS] Alvestrand, H., "Compare Codecs", 2015, <http://compare-codecs.appspot.com/>.

[DAALA-GIT] Xiph.Org, "Daala Git Repository", 2015, <http://git.xiph.org/?p=daala.git;a=summary>.

[I-D.ietf-netvc-requirements] Filippov, A., Norkin, A., and j. jose.roberto.alvarez@huawei.com, "Video Codec Requirements and Evaluation Methodology", draft-ietf-netvc-requirements-10 (work in progress), November 2019.

[MSSSIM] Wang, Z., Simoncelli, E., and A. Bovik, "Multi-Scale Structural Similarity for Image Quality Assessment", n.d., <http://www.cns.nyu.edu/~zwang/files/papers/msssim.pdf>.

[PSNRHVS] Egiazarian, K., Astola, J., Ponomarenko, N., Lukin, V., Battisti, F., and M. Carli, "A New Full-Reference Quality Metrics Based on HVS", 2002.

[RD_TOOL] Xiph.Org, "rd_tool", 2016, <https://github.com/tdaede/rd_tool>.

[SSIM] Wang, Z., Bovik, A., Sheikh, H., and E. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity", 2004, <http://www.cns.nyu.edu/pub/eero/wang03-reprint.pdf>.

[TESTSEQUENCES] Daede, T., "Test Sets", n.d., <https://people.xiph.org/~tdaede/sets/>.

[VMAF] Aaron, A., Li, Z., Manohara, M., Lin, J., Wu, E., and C. Kuo, "VMAF - Video Multi-Method Assessment Fusion", 2015, <https://github.com/Netflix/vmaf>.

Authors' Addresses

Thomas Daede
Mozilla

Email: tdaede@mozilla.com

Andrey Norkin
Netflix

Email: anorkin@netflix.com

Ilya Brailovskiy
Amazon Lab126

Email: brailovs@lab126.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 4, 2017

S. Midtskogen
Cisco

October 31, 2016

Improved chroma prediction
draft-midtskogen-netvc-chromapred-02

Abstract

This document describes the technique used to improve the chroma prediction in the Thor video codec.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 4, 2017.

Copyright Notice

Table of Contents

1. Introduction

Modern video coding standards such as Thor [I-D.fuldseth-netvc-thor] form predictions for the luma channel (Y) and chroma channels (U and V) which are encoded separately (in that order). The prediction for each channel has spatial or temporal dependencies only in its own channel. Most of the perceived information of a video is to be found in the luma channel, but there still remain correlations between the luma and chroma channels. For instance, the same shape of an object can often be seen in all three channels, and if this correlation is not exploited, some structural information will be transmitted three times. Thor will attempt to improve the chroma prediction by finding linear relationships between the each of the initial chroma predictions and the luma prediction, and if certain criteria are satisfied, use that relationship to form a new prediction based on the reconstructed luma samples.

2. Definitions

2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. Background

The improved predictions are derived from the reconstructed luma samples using a mapping. The underlying assumption is that the colours can be identified by their luminosities. Informally we can say that a new chroma prediction is formed from the reconstructed luma block painted with the colours of the initial chroma prediction.

There is often a linear correlation between the luma and chroma channel, so that a chroma sample c can be expressed by the linear function

```
c = a*y + b
```

Figure 1: Linear relationship

where y is the corresponding luma sample. This observation has been previously been used in techniques to convert YUV 4:2:0 and YUV 4:2:2 images to YUV 4:4:4, and in a (rejected) proposal for HEVC as a special intra mode. Thor, however, generalises the prediction, so it does not depend on the coding mode (i.e. whether inter or intra, or the kind of inter/intra mode).

Since it would be too costly to transmit the values a and b in the linear mapping, and since both the encoder and decoder must be able to compute identical predictions, a and b are derived from data available to both using linear regression.

4. Computing the improved prediction

Since the assumption that the correlation is the same in the predicted block and in the reconstructed block is not always true, the new prediction from luma might not be better even when there is a very good correlation in the predicted block. Therefore, we can only expected an improvement if the initial prediction is bad, and the luma residual is used as an estimate for this. The initial chroma prediction is kept unless the average squared difference between the reconstructed luma samples yr and the predicted y samples for an N*N prediction block is above 64:

```
 _N_ _N_
 \   \
 /__ /__ (yr(i, j) - y(i, j)) ^ 2
 i=1 j=1
 -------------------------------- > 64
                N*N
```

Figure 2: Requirement for improvement 1

The encoder and decoder must compute a and b using the same least square fit for an N*N prediction block, where y and c denote the luma and chroma samples in the initial prediction:

$$Ysum = \sum_{i=1}^{N}\sum_{j=1}^{N} y(i, j) \qquad Csum = \sum_{i=1}^{N}\sum_{j=1}^{N} c(i, j)$$

$$YYsum = \sum_{i=1}^{N}\sum_{j=1}^{N} y(i, j) \wedge 2 \qquad CCsum = \sum_{i=1}^{N}\sum_{j=1}^{N} c(i, j) \wedge 2$$

$$YCsum = \sum_{i=1}^{N}\sum_{j=1}^{N} y(i, j) * c(i, j)$$

Figure 3: Equations for linear regression 1

These sums will all be contained within a 32 bit signed integer when the internal bitdepth is 8. Otherwise 64 bit integers must be used. Then the following must be computed using 64 bit arithmetic regardless of bitdepth:

```
SSyy = YYsum - ((Ysum * Ysum) >> 2*log2(N))
SScc = CCsum - ((Csum * Csum) >> 2*log2(N))
SSyc = YCsum - ((Ysum * Csum) >> 2*log2(N))
```

Figure 4: Equations for linear regression 2

Still using 64 bit arithmetic, if

```
SSyy > 0 /\ 2 * SSyy * SSyy > SSyy * SScc
```

Figure 5: Requirement for improvement 2

then it is assumed that the correlation is reasonably good and a new prediction will be computed and used. Otherwise, the initial prediction will be kept. First, a and b must be computed. 2^15 is added to b to ensure correct rounding later on.

```
a = (SSyc << 16) / SSyy
b = (((Csum << 16) - a * Ysum) >> 2*log2(N)) + (1 << 15)
```

Figure 6: Equation for linear regression 3

The final operations are performed with 32 bit arithmetic, so a must be clipped to [-2^(31-B), 2^(31-B)], where B is the bitdepth, and b must be clipped to [-2^31, 2^31-1]. The a new chroma prediction c' is computed using the reconstructed luma samples yr, a and b, and a clipping function saturating the results to an 8 bit value:

```
c'(i, j) = clip((a * yr(i, j) + b) >> 16)
```

Figure 7: Improved chroma prediction

The above assumes 4:4:4 format. For the 4:2:0 format the predicted luma block must be subsampled first:

```
y'(i,j) = (y(2*i, 2*j)   + y(2*i+i, 2j) +
           y(2*i, 2*j+1) + y(2*i+1, 2*j+1) + 2) >> 2
```

Figure 8: Subsampling of predicted luma block

The resulting new chroma prediction must also be subsampled. The clipping is performed before the subsampling.

```
c'(i, j) = (clip((a*yr(2*i, 2*j) + b) >> 16) +
            clip((a*yr(2*i+1, 2*j) + b) >> 16) +
            clip((a*yr(2*i, 2*j+1) + b) >> 16) +
            clip((a*yr(2*i+1, 2*j+1) + b) >> 16) + 2) >> 2
```

Figure 9: Subsampling of improved chroma prediction

In intra mode the chroma prediction improvement must be performed right after each transform, since the new chroma reconstruction will be used to predict the next block.

5. Performance

The improved chroma prediction may significantly improve the compression efficiency for images or video containing high correlations between the channels. It is particularly useful for encoding screen content, 4:4:4 content, high frequency content and "difficult" content where traditional prediction techniques perform poorly. Little quality change is seen for content not in these categories, but there is a general small increase in chroma PSNR.

An encoded configured for low delay and high complexity was used for the following results. The numbers have been computed using the Bjontegaard Delta Rate (BDR [BDR]). The rates for Y, U and V have been shown separately.

	4:4:4			4:2:0		
Sequence	Y	U	V	Y	U	V
cad_waveform	-21.3%	-27.0%	-24.0%	0.5%	-1.3%	-1.1%
pcb_layout	-9.2%	-13.3%	-10.6%	-1.6%	-3.1%	-3.5%
ppt_doc_xls	-6.3%	-14.1%	-12.7%	-0.1%	-0.8%	-0.8%
vc_doc_sharing	-2.9%	-6.4%	-6.9%	0.3%	-1.2%	-0.6%
web_browsing	-0.5%	-1.1%	-1.5%	0.3%	-0.5%	-1.0%
wordEditing	-1.8%	-5.9%	-4.8%	1.5%	1.2%	1.1%
park_joy	-0.5%	-2.6%	-0.9%	-0.0%	-0.8%	0.4%
old_town_cross	-0.1%	-2.2%	-1.2%	0.0%	-0.6%	-0.2%
Average	-5.3%	-9.1%	-7.8%	0.1%	-0.9%	-0.7%

Figure 10: Compression Performance, improved prediction for intra blocks only

	4:4:4			4:2:0		
Sequence	Y	U	V	Y	U	V
cad_waveform	-23.1%	-28.9%	-26.1%	-2.6%	-3.6%	-3.5%
pcb_layout	-21.0%	-29.0%	-21.0%	-5.4%	-7.9%	-5.4%
ppt_doc_xls	-9.0%	-19.0%	-17.5%	-0.2%	-0.2%	-1.2%
vc_doc_sharing	-4.7%	-9.6%	-9.6%	-0.1%	-1.0%	-0.4%
web_browsing	-0.6%	-1.5%	-1.5%	-0.5%	-1.2%	-1.2%
wordEditing	-11.3%	-13.7%	-11.7%	-3.0%	-4.2%	-3.2%
park_joy	-5.5%	-7.4%	-7.1%	-0.9%	-1.9%	-1.6%
old_town_cross	-1.7%	-3.6%	-2.2%	-0.3%	-4.1%	-1.6%
Average	-9.6%	-14.1%	-12.1%	-1.6%	-3.0%	-2.3%

Figure 11: Compression Performance, improved prediction using intra only coding

	4:4:4			4:2:0		
Sequence	Y	U	V	Y	U	V
cad_waveform	-11.5%	-14.4%	-12.7%	0.0%	-1.8%	-1.7%
pcb_layout	-3.2%	-5.5%	-4.8%	-0.9%	-2.4%	-3.4%
ppt_doc_xls	-0.1%	-0.7%	-0.3%	0.0%	-0.2%	-0.6%
vc_doc_sharing	-0.4%	-0.6%	-1.6%	-0.0%	-0.4%	-0.6%
web_browsing	0.1%	0.2%	0.1%	0.5%	-0.0%	-0.9%
wordEditing	-3.7%	-5.8%	-6.2%	0.4%	-0.9%	-1.4%
park_joy	-1.6%	-8.6%	-1.5%	0.0%	-3.5%	-0.2%
old_town_cross	-0.0%	-0.4%	-0.1%	0.0%	0.1%	-0.2%
Average	-2.5%	-4.5%	-3.4%	0.0%	-1.1%	-1.1%

Figure 12: Compression Performance, improved prediction for inter blocks only

	4:4:4			4:2:0		
Sequence	Y	U	V	Y	U	V
cad_waveform	-25.8%	-31.7%	-28.2%	-2.4%	-5.5%	-5.4%
pcb_layout	-11.5%	-16.1%	-13.5%	-2.4%	-4.1%	-5.6%
ppt_doc_xls	-6.3%	-14.3%	-13.2%	-0.2%	-0.8%	-0.8%
vc_doc_sharing	-3.0%	-6.7%	-8.2%	0.1%	-0.9%	-1.1%
web_browsing	-0.5%	-1.2%	-1.5%	0.2%	-0.3%	-2.0%
wordEditing	-3.4%	-6.8%	-6.6%	0.6%	-0.5%	-1.4%
park_joy	-1.7%	-9.2%	-1.7%	-0.0%	-4.0%	0.0%
old_town_cross	-0.1%	-2.2%	-1.0%	0.1%	-0.5%	-0.1%
Average	-6.5%	-11.0%	-9.2%	-0.5%	-2.1%	-2.0%

Figure 13: Compression Performance, improved prediction for intra and inter blocks

6. IANA Considerations

This document has no IANA considerations yet. TBD

7. Security Considerations

This document has no security considerations yet. TBD

8. Acknowledgments

The author would like to thank Arild Fuldseth and Mo Zanaty for reviewing this document, and Timothy Terriberry for pointing a couple of errors in the first draft.

9. References

9.1. Normative References

[I-D.fuldseth-netvc-thor]
Fuldseth, A., Bjontegaard, G., Midtskogen, S., Davies, T., and M. Zanaty, "Thor Video Codec", draft-fuldseth-netvc-thor-02 (work in progress), March 2016.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

9.2. Informative References

[BDR] Bjontegaard, G., "Calculation of average PSNR differences between RD-curves", ITU-T SG16 Q6 VCEG-M33 , April 2001.

Author’s Address

Steinar Midtskogen
Cisco
Lysaker
Norway

Email: stemidts@cisco.com