Thor update

High Efficiency, Moderate Complexity Video Codec using only RF IPR

(https://datatracker.ietf.org/ipr/2636/)

draft-fuldseth-netvc-thor-03

Steinar Midtskogen (Cisco)

IETF 97 – Seoul, KR – November 2016

Topics for this update

- Most important changes since IETF96/July 2016
  - Support for 10 and 12 bit video
  - Support for high internal precision regardless of input
  - Optimisations:
    - support for 256 bit generic intrinsics
    - optimisations for AVX2
    - SIMD optimisations for high bitdepths
  - Source code reworked a bit to allow different bitdepths while staying reasonably efficient

Support for higher bitdepths

- Input video can be 8, 10 or 12 bit
- The internal bitdepth can be 8, 10 or 12 bit independently of the input video
  - A compression gain is achieved for 8 bit video if the internal bitdepth is 10 or 12 bit (but adds ~20-50% more complexity)
  - Also possible to encode 10 or 12 bit video with 8 bit internal bitdepth, which reduces the computational complexity but of course also reduces the quality
- Video bitdepth and internal bitdepth signalled in the sequence header

Support for higher bitdepths

- Gains (Y) for 8 bit video with 10 bit internal depth:

Sequence	BDR	BDR-low	BDR-high
Kimono	-2.269	-1.223	-3.782
BasketballDrive	-2.036	-1.017	-3.478
BQTerrace	-1.890	-0.998	-2.679
FourPeople	-3.116	-2.054	-4.784
Johnny	-5.381	-2.956	-8.745
ChangeSeats	-3.497	-1.267	-6.734
HeadAndShoulder	-6.674	-2.212	-14.300
TelePresence	-5.289	-3.111	-8.910
Average	**-3.769**	**-1.855**	**-6.676**

(BDR-low = low bitrates, BDR-high = high bitrates)

Support for higher bitdepths

- Gains (Y) for 8 bit video with 12 bit internal depth:

Sequence	BDR-Y	BDR-low	BDR-high
Kimono	-2.706	-1.599	-4.335
BasketballDrive	-2.380	-1.334	-3.861
BQTerrace	-2.296	-1.336	-3.073
FourPeople	-3.269	-1.755	-5.541
Johnny	-6.192	-3.544	-9.900
ChangeSeats	-4.351	-1.793	-8.175
HeadAndShoulder	-7.367	-1.921	-16.501
TelePresence	-6.197	-3.073	-11.285
Average	**-4.345**	**-2.044**	**-7.834**

(BDR-low = low bitrates, BDR-high = high bitrates)

Support for higher bitdepths

- Gains (U) for 8 bit video with 12 bit internal depth:

Sequence	BDR-U	BDR-low	BDR-high
Kimono	-5.743	-6.006	-5.814
BasketballDrive	-6.626	-5.789	-8.461
BQTerrace	-21.039	-21.194	-23.060
FourPeople	-7.446	-5.596	-9.375
Johnny	-12.751	-7.147	-20.150
ChangeSeats	-10.610	-6.024	-14.908
HeadAndShoulder	-20.791	-3.984	-35.228
TelePresence	-18.174	-10.597	-28.663
Average	**-12.898**	**-8.292**	**-18.207**

(BDR calculated using U PSNR and bitrate dominated by Y)

Support for higher bitdepths

- Gains (V) for 8 bit video with 12 bit internal depth:

Sequence	BDR-V	BDR-low	BDR-high
Kimono	-5.140	-4.864	-5.457
BasketballDrive	-5.167	-3.505	-7.229
BQTerrace	-29.056	-24.813	-35.843
FourPeople	-4.875	-2.212	-7.777
Johnny	-15.950	-11.395	-21.148
ChangeSeats	-10.956	-4.079	-21.376
HeadAndShoulder	-17.933	-8.220	-40.296
TelePresence	-13.992	-5.361	-25.787
Average	**-12.884**	**-8.056**	**-20.614**

(BDR calculated using V PSNR and bitrate dominated by Y)

Support for higher bitdepths

- Higher internal bitdepth means that the following must be adjusted according to the bitdepth:
  - The shift after the first pass of the forward transform
  - The shift after the second pass of the inverse transform
  - The strengths of the constrained low-pass filter
  - The threshold and beta of the deblocking filter
  - The pixel clipping
- The quantisation and dequantisation are unchanged, and the qp range remains the same
- Small changes on paper, but large implications for the source code

Support for higher bitdepths

- Since we want to keep the 8 bit optimisations, the frame buffers can be either 8 bit or 16 bit
- In C++ the use of templates would have been handy, but since Thor is written in C99, simple preprocessor magic is used to mimic C++ templates
- Much of the code in the encoder and decoder gets instantiated twice: once for 8 bit frame buffers and once for 16 bit frame buffers
- The amount of source code remains roughly the same, but the executables increase in size

256 bit generic intrinsics

- Thor uses an abstraction of common SIMD intrinsics to support SIMD optimisations for several architectures (NEON, SSE2, SSSE3, etc)
  - Presented at IETF94
- Not required for standardisation, but useful during standardisation to assess the complexity in real applications, and it makes a reference codec more “shipping ready”
- SIMD optimised code for 8 bit needs a 10/12 bit counterpart and writing SIMD code is tedious
  - Several thousand lines of code

256 bit generic intrinsics

- Instead of spending weeks or months on writing such code, it's possible to convert existing code automatically and get a reasonably good result
- Intrinsics like v64_add_8 becomes v128_add_16, v64_load_aligned becomes, v128_load_aligned etc.
- 256 bit intrinsics were added so that v128_add_8 could become v256_add_16, etc
- Native support on AVX2, otherwise emulated (mostly by duplicating 128 bit intrinsics)
- Just one set of SIMD optimised code to maintain regardless of bitdepth and architecture

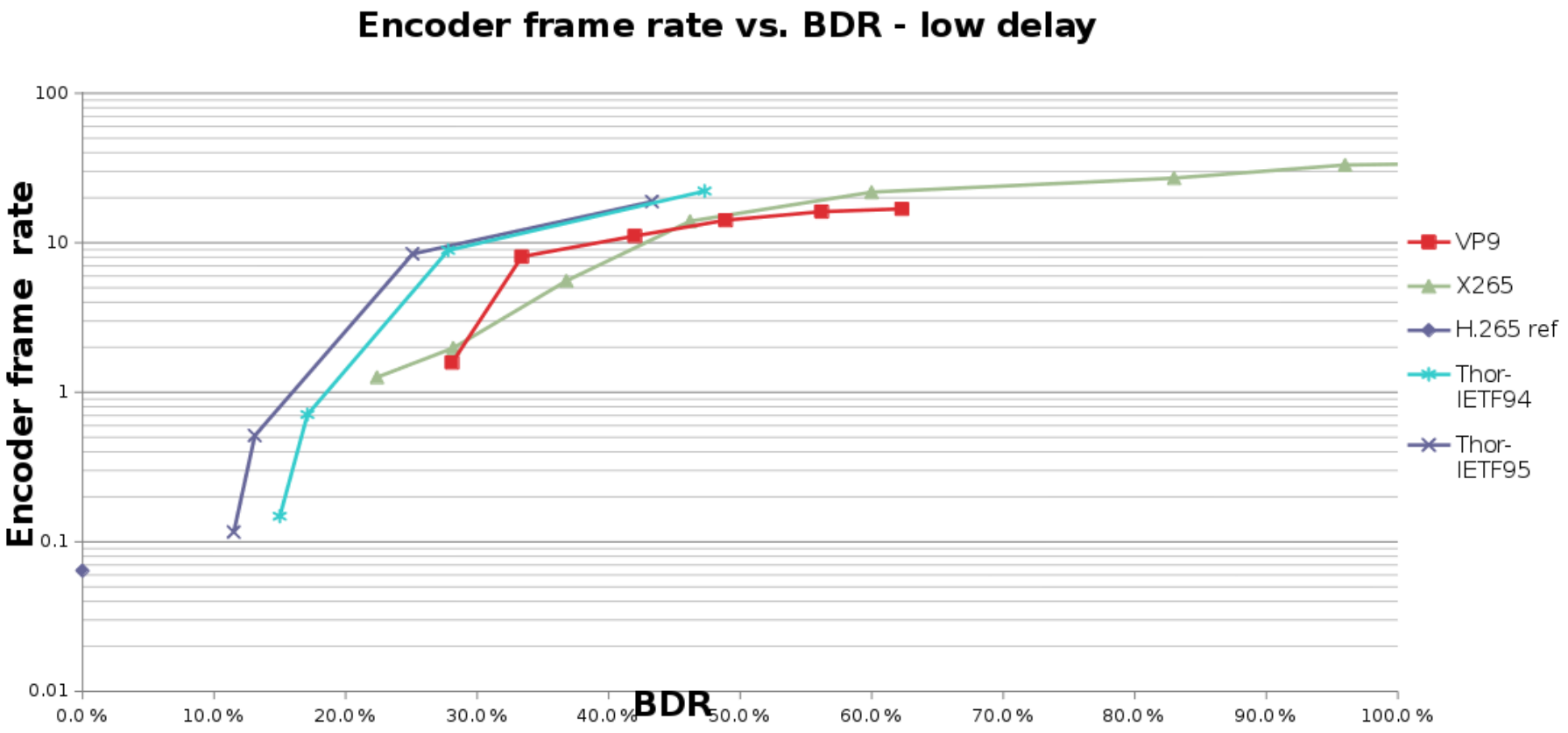
256 bit generic intrinsics

- Instead of writing entirely new code for 16 bit frame buffers, the existing code for 8 bit was reworked slightly to make it trivial to convert
- SIMD code for 16 bit frame buffers generated by a sed script (“search and replace”)
- The addition of 256 bit intrinsics also opens up for AVX2 optimisations of 8 bit code (not yet done)

Summary

- Thor supports:
  - 8, 10 or 12 bit video
  - 4:2:0 or 4:4:4 chroma sampling
  - SIMD optimisations for 8, 10 and 12 bit configurations
    - NEON, SSE2, SSSE3, SSE4.1, AVX2
- Not directly supported but can be trivially handled:
  - Interlaced video
  - RGB 4:4:4
- https://github.com/cisco/thor

Frame rate vs compression LD



Frame rate vs compression HD