Quo Vadis JPEG: Future of ISO 10918-1/T.81

- 10918/T.81 is still the dominant standard for photographic images.
- An entire toolchain exists to record, manipulate and display images encoded in this specification.
- Market penetration of JPEG 2000 and JPEG XR is very low in the digital camera market.
- Can we extend 10918 *carefully* without breaking “too much” of this toolchain to address the main deficiencies of the specs?
Compatibility Issues

- Only a small subset of 10918 is in wide use
  - only baseline, sequential and progressive mode are popular
  - only up to four components (single scan sequential or progressive) are seen, though up to 255 components are possible
  - only 8bpp images are in popular use, though 12bpp is available
  - Medical industry uses 12bpp and lossless (“predictive”) mode
  - Arithmetic coding mode is not seen in the wild
  - Hierarchical mode is not seen in the wild.
10918 Issues

• Alpha channel support missing
  - this is a file format issue to annotate the components
• Backwards compatible lossless mode is missing
  - the “predictive coding” mode is an entirely different codec that is not understood by the typical JPEG toolchain
• Backwards compatible support for >8bpp is missing
  - 12bpp mode is not supported by most implementations.
Goals:

- Can we extend 10918 *carefully* such that
  - the current toolchain remains working, though does probably not support all features of the new standard
  - for example, decoding a losslessly encoded image with some minor loss
  - for example decode a >8bpp image to 8bpp (automatic LDR rendering of HDR images)
- compression performance is *not* the primary goal, but compatibility is. Performance is secondary.
- several possibilities will be discussed and compared with the state of the art
- Teaser: Backwards compatible lossless JPEG with the performance of PNG *is possible.*
Composition of several designs:
Refinement coding
Additive Residual coding (Stuttgart proposal)
Multiplicative Residual Coding (Dolby proposal)
Tone Mapping

- A lookup table that maps HDR data to LDR data, and vice versa at the decoder.
- How to arrive at a tone mapping curve is not defined in the specifications – this is a matter of the implementation.
- Only the inverse tone mapping curve at the decoder is included in the bitstream. The encoder has to find the inverse of the inverse.
- Note that the input data could be even floating point without any change in the coder design – it is just a matter of how to interpret the bit pattern and how to build the TMO.
Color Transformation

- Is (mathematically) identical to the RGB to YcbCr transformation specified in JFIF
- But coefficients are specified:
  - Fixpoint algebra, 13 fractional bits, output is not integer, but fixpoint with 4 fractional bits
- Must become part of the standard (unlike in 10918-1)
DCT Operation

- Is an unscaled fixpoint DCT
- Similar to the color transformation, the coefficients and the algorithm must be specified precisely (for residual and lossless coding)
- Coefficients are specified in fixpoint with 9 fractional bits
- 32 bit implementation precision is sufficient for 12bpp input data
Quantization

- Unmodified from 10918
- Followed by a split-off in MSBs and LSBs.
  - Split off is a shift for DC (round to $-\infty$) and a division for AC (round to 0)
- This is the same rule used for the “point transformation” of 10918
Scan Types

Legacy Scan Types:
- Transmits the MSBs of the tone mapped data
- All block based scan types 10918 has to offer:
  - Sequential, progressive (Huffman and AC)

Refinement Scan:
- Transmits the LSBs of the tone mapped data
- MSB/LSB split off by means of a “point transformation” not signaled in the legacy coder, but in a separate marker
- Scan type is a 10918-1 compliant “refinement” scan (subsequent scan of a progressive scan) extending the legacy scan type.
Scan Types

Residual Scan:
- Transmits the quantized coding error
- Uses a traditional progressive scan, on 8x8 blocks in the spatial domain, except that the left-top pixel ("DC") is not handled separately, but AC coding is used for all pixels.
- Smarter coding schemes (context modeling) do not improve the coding performance substantially (<1%)
Additive Residual Coding

- Optimal R/D performance
- Lossless Coding
- Floating Point Coding Supported
Refinement Coding

- Simplest possibility for HDR coding
- Very easy to implement
- R/D performance is non-ideal
Multiplicative Coding

- Design is compatible to Dolby JPEG-HDR
- Codestream syntax is a bit different
- Floating Point Coding Supported
Encoder Design (Refinement & add. residual coding)

- TMO
- RGB to YCbCr
- FixPt FDCT
- QNT
- Legacy Scan Types

Refinement Scan
- APP9 FINE marker

TMO⁻¹
- YcbCr to RGB
- FixPt IDCT
- Inverse QNT

Residual Coding
- RCT
- Noise Shaping
- QNT
- Residual Scan
- APP9 RESI marker
Experiments

Input data was taken from the core2 test set and the OpenEXR test image set, first converted to 16-bit float (half-float) for consistency.

- Note that due to the tone mapping the codec does not even need to know whether the input data is integer or floating point. TMO input is the “bit pattern” of the floating point numbers.

Participating codecs:

- JPEG 2000 (floating point coding as in AMD.3 of 15444-2)
- JPEG XR  (half-float RGB pixel type)
- JPEG-HDR (hdrcvt by Greg Ward, M. Simmons)
- JPEG extensions using residual coding only
- JPEG extensions using residual and refinement coding
- External rate allocation for JPEG, JPEG XR
- JPEG extensions uses a simplified version of the Reinhard tonemapper, with or without gamma adaption.
Notes on hdrcvt

Published by G. Ward and M. Simmons in ACM Siggraph 2005.
• Builds a LDR version of the HDR image by TMO similar to here,
• But encodes the quotient of HDR over LDR
• A precursor of Dolby's JPEG-HDR.
Measurements

Identical to the JPEG XR core experiments on HDR

Measure SNR in the HDR regime, MS-MSSIM and PSNR after tonemapping with exrpptm
Tree

SNR/dB

bpp

jpeg residual
jpeg refinement
jpeg-hdr
JPEG XR
jpeg2000
jpeg2000 (visual)
Thomas Richter, University of Stuttgart
Future of 10918 (JPEG) – Quo Vadis JPEG

![Graph showing MS-SSIM/dB vs. bpp for different JPEG variants](image-url)
Conclusions

The JPEG extensions work generally well, with residual coding working better than refinement coding. For low bitrates, the coders are not competitive (not a surprise).
In general, the JPEG extensions work better than hdrcvt (JPEG-HDR).
Results depend on how extreme the lighting conditions are and how far the automatic TMO provided by the codec differs from the intended TMO.

JPEG-Extensions Demo Implementation plus Working Draft: https://github.com/thorfdbg/libjpeg

JPEG-Online Test: http://jpegonline.rus.uni-stuttgart.de/index.py