|  |  |
| --- | --- |
| **ITU – Telecommunications Standardization Sector**  STUDY GROUP 16 Question 6  **Video Coding Experts Group (VCEG)**  74th Meeting: 12-19 July 2024, Sapporo, JP | Document VCEG-BV03-v1 |

|  |  |  |  |
| --- | --- | --- | --- |
| Question: | Q.6/SG16 (VCEG) | | |
| Source: | **Elena Alshina (Huawei),**  **João Ascenso (Instituto Superior Técnico),**  **Touradj Ebrahimi (EPFL)** | Tel:  Email: | [elena.alshina@huawei.com](mailto:elena.alshina@huawei.com)  [joao.ascenso@lx.it.pt](mailto:joao.ascenso@lx.it.pt)  [touradj.ebrahimi@epfl.ch](mailto:touradj.ebrahimi@epfl.ch) |
| Title: | **Summary of the design, features and performance of the draft JPEG AI standard** | | |
| Purpose: | Report | | |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Abstract

JPEG AI is joint project of JPEG and VCEG Q.6/SG16, initiate in 2020. The project has successfully reached DIS stage. This document summarizes key feature of JPEG AI codec, reports performance and provides visual quality examples.

# JPEG AI key design elements

JPEG AI operates with 4:4:4, 4:2:2 or 4:2:0 images, 8 or 10 bits. It could be YUV, RGB or any other colour space. Images which are 4:4:4 can be encoded as 4:4:4 or 4:2:0.

For coding purposes images are converted to YUV BT.709. Colour planes are separated. The luminance component is encoded independently with more powerful neural network. Chrominance is encoded and decoded using information from luminance as auxiliary input.

The bitstream structure of JPEG AI is shown on Figure 1. There are several sub-streams, each of those starts with unique marker, followed by sub-stream size. The start and the end of each sub-stream are easy to identify and (except picture header) sub-streams can appear in arbitrary order.

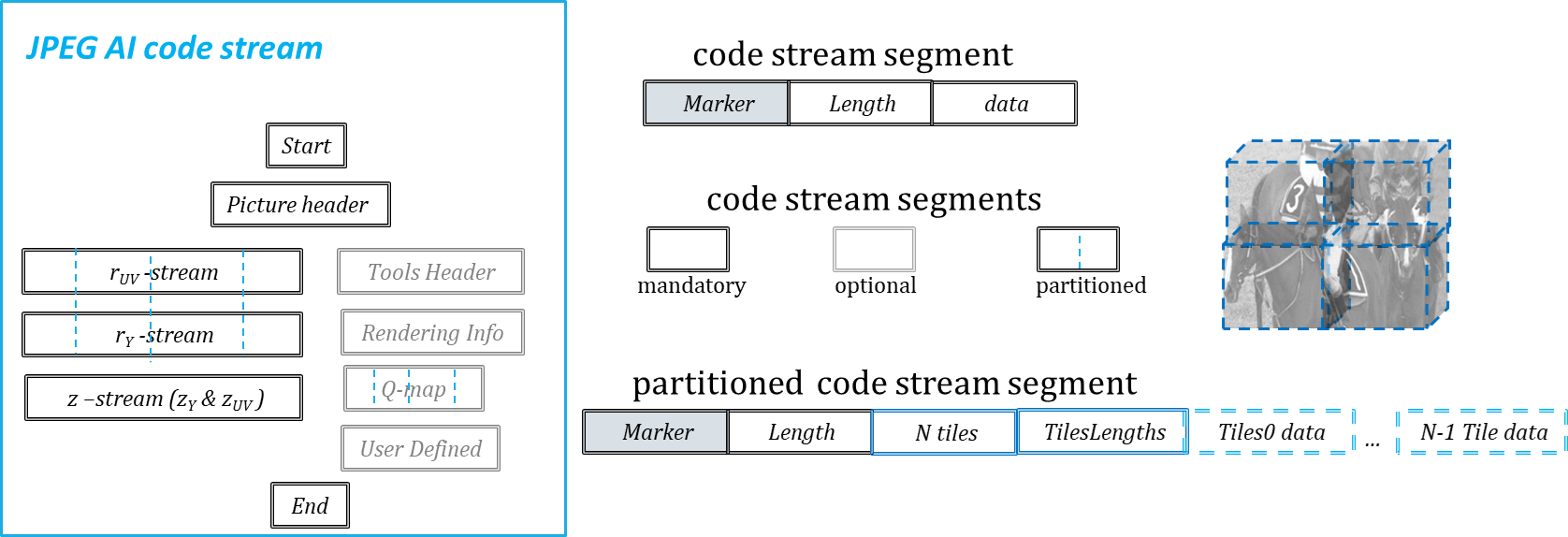


Figure 1 JPEG AI code stream structure

Some sub-streams are mandatory to be present, some are optional. Big in size sub streams are partitioned (the table of offsets is signaled). Parts of those sub streams (residual tiles) correspond to the parts of the picture and so partial reconstruction of picture is possible by parsing only needed tiles of residual sub-streams.

As usually only decoder operations are specified, examples of encoder are available in the reference SW. Encoder generates tensor representations of Luminance and Chrominance. Luminance tensor has 160 channels, chrominance tensor has 96 channels.

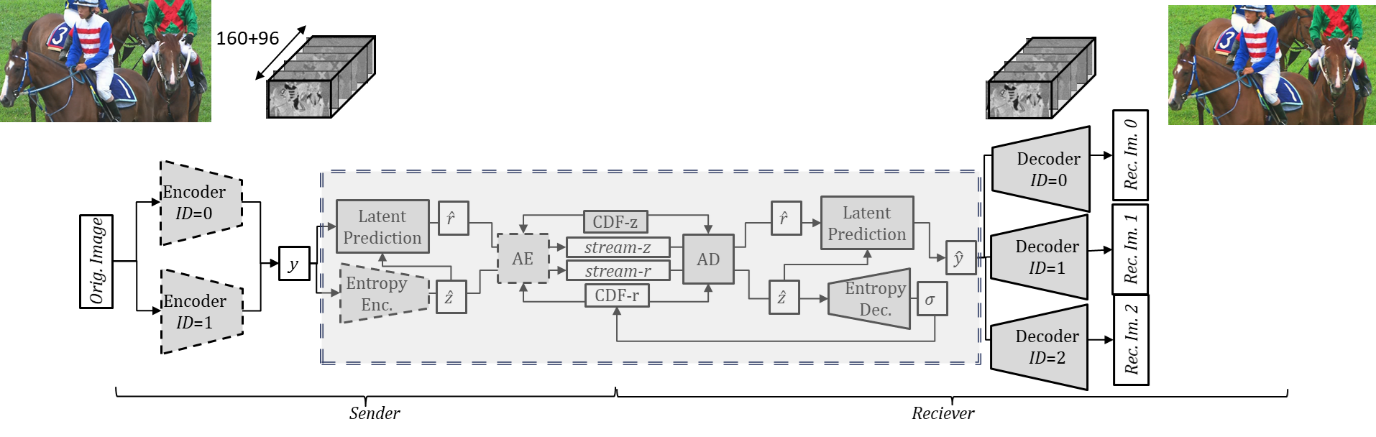


Figure 2 JPEG AI architecture

There are in total four models trained for different level of quality from nearly lossless to very lossy (operating at ultra-low rate). Quality is controlled by selecting one of the four models (the model choice signaled to the decoder) and so-called beta parameter which controls the ratio between rate and quality and enables the fine rate adaptation (if needed) by changing beta for Luminance and Chrominance independently.

Tensor representation of Luminance and Chrominance – latent tensor – is coded using identical networks (the only difference is number of channels). Tensor representation of image is coded using prediction, which is subtracted from latent tensor.

Luminance and Chrominance residuals are coded in two sub-streams, which occupies the major part of the bit-stream.

Relatively small in size sub-stream (‘stream-z’ in Figure 2) contains information which is used for 1) latent tensor prediction generation, 2) generation of entropy parameters for residual stream decoding.

Latent domain prediction (Figure 4) consists of 1) hyper decoder network which generates the prediction from hyper tensor z, and 2) multi-stage context modelling (MCM), which combines residual with prediction., utilizing correlation with previously reconstructed latent tensor elements. Inside MCM the tensor is re-shuffled into four sub-tensors, each of those reconstructed depending on no more than three previously reconstructed ones. The depth of recursion is four.

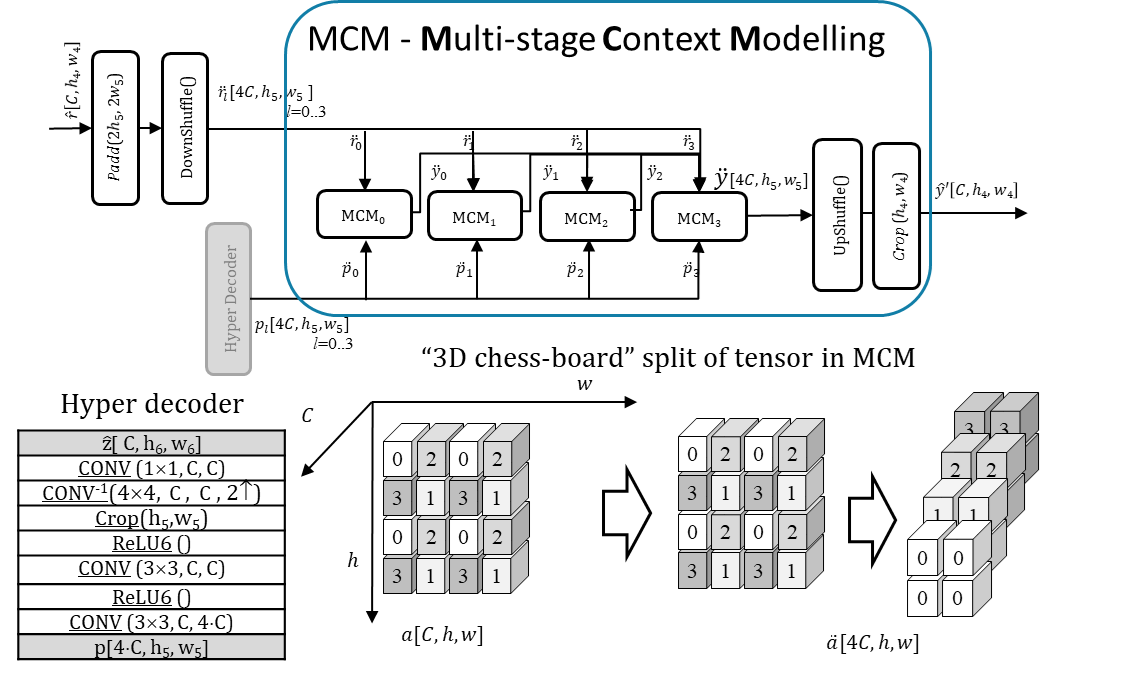


Figure 3 Latent tensor prediction in JPEG AI

Entropy decoder of JPEG AI consists of hyper scale decoder network. It generates entropy parameters from hyper tensor z. For device interoperability the bit-exact behavior of the entropy decoder is required, otherwise parsed symbols are misinterpreted and decoder image is totally broken. Bit exact behavior was achieved by using only ReLU as activation, replacing all convolution with quantized convolution (which include de-scaling and clipping) and special quantizer of neural network parameters, which ensures no overflow 32 bits register. Mathematical proof of no overflow is documented.

Knowing that many AI-accelerators do not allow full control and so entropy decoder might be run on CPU for some implementations Hyper-scale decoder network design is simple enough to be performed on CPU: only one 3×3 convolution and two 1×1 convolution (Figure 4).

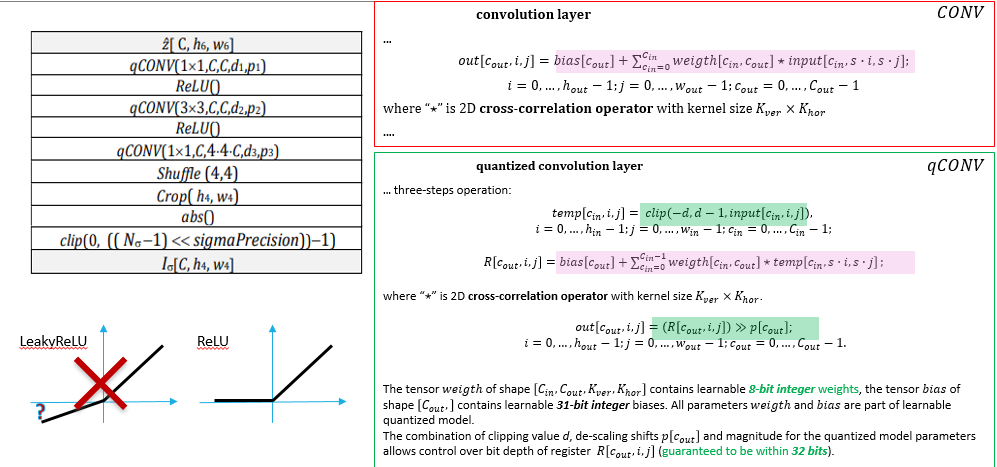


Figure 4 Hyper scale decode.

For decoding hyper sub-stream and residual sub-streams the specific version of Asymmetric Numeral Systems (ANS) is used. It is so-called memory efficient tabulated ANS (me-tANS). At decoder side it requires one table look up, five bit-shift and one add operations. Enabling multi-threading for me-tANS shows significant speed-up with only minor bit-stream size increase: 16 threads provides ×3 run-time reduction with only 0.1% bits overhead.

Reconstructed latent tensor goes to synthesis transform neural network. As shown on Figure 1 any stream produced by any of two encoders in reference software are decoded producing latent tensor representation which can be sent to any of three synthesis transform neural networks and reconstructed image will be close of the encoded image.

Specification describes three versions of synthesis transform with different level of complexity. Below the Figure 5 illustrates how difference to original image (blue-red image is a difference) decreases if more complex synthesis transform is applied.

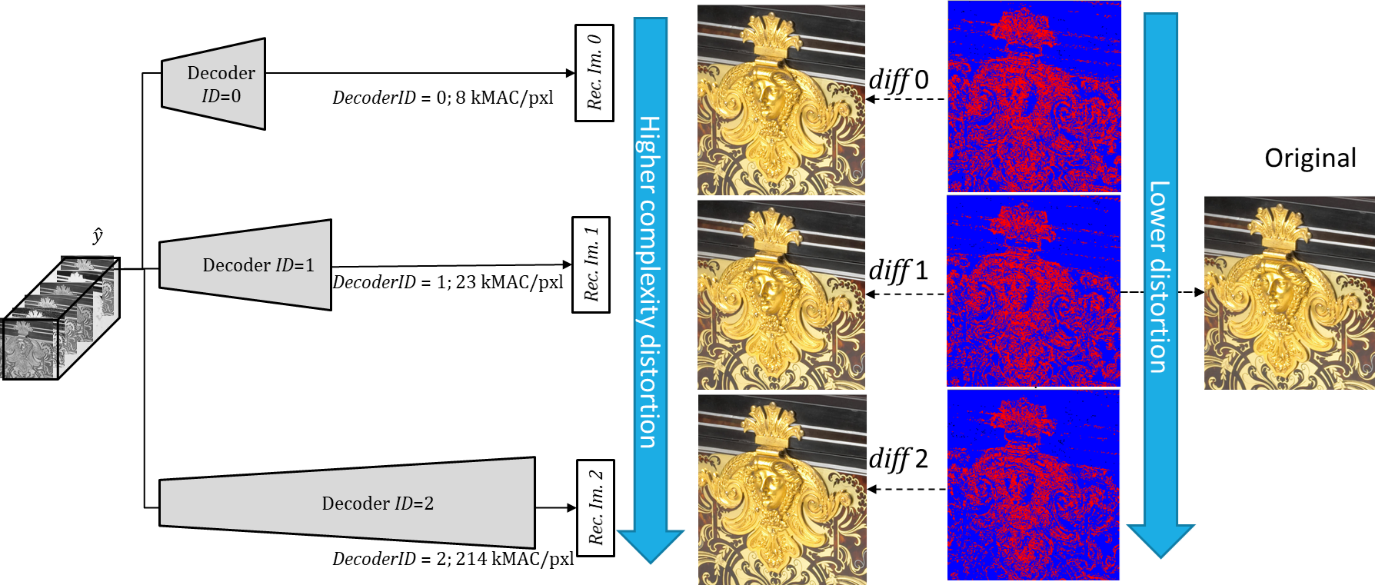


Figure 5 Multiple synthesis transform from the same tensor representation of image

Total computational complexity in kMAC/pixel for different synthesis transforms is 8 for the lowest complexity, 23 for middle and 214 for highest complexity synthesis transform. The key difference of the highest performance synthesis transform network (DecoderID=2) from others is existence of attention mechanism (which includes transformers).

# JPEG AI functionalities

Some functionalities are quite easy to realize with end-to-end AI codec architecture.

The channels of latent tensor are ordered on a way the strongest channels come first. On devices with low computational capability or for image pre-view generation the higher channels can be discarded (not even parsed, or even not encoded) and reconstructed image is viewable (of cause has lower quality). The progressive decoding is illustrated on Figure 6: with 8-10% of bit-stream decoded the quality is poor, with 24% of bit-stream decoded quality is already good for pre-view, with 93% bitstream decoded quality is almost the same as for full stream decoding.

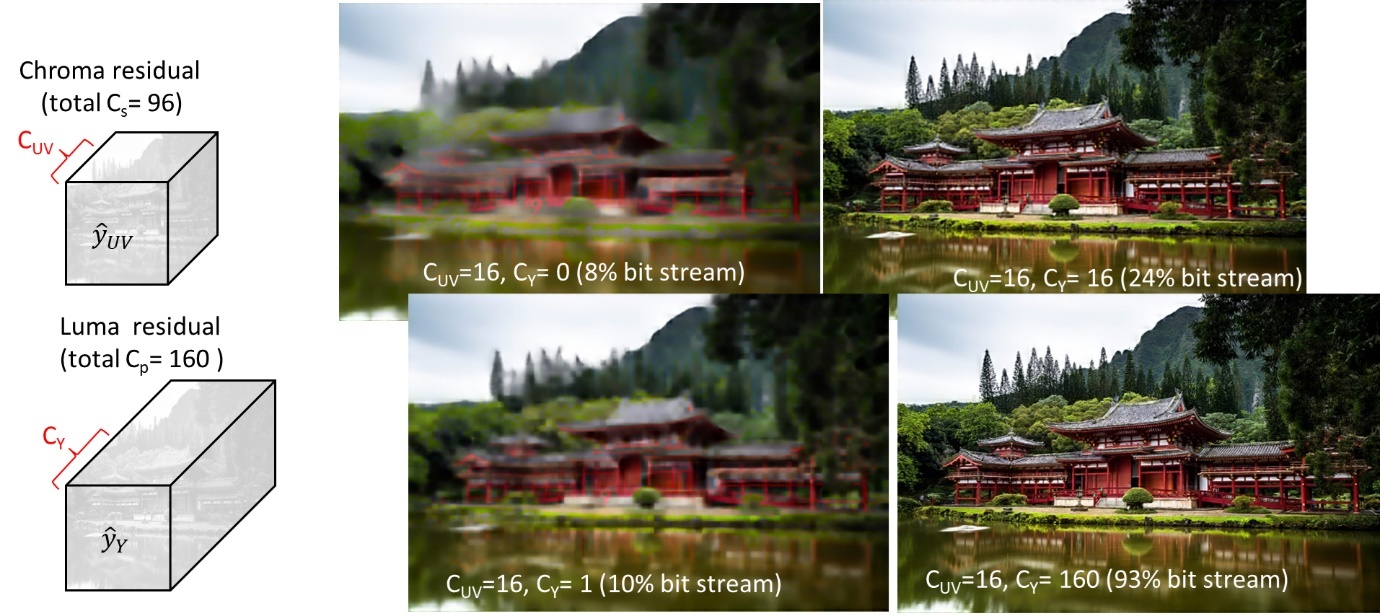


Figure 6 Progressive decoding

Sub stream with Chrominance data be discarded and grey scale image will be reconstructed (this is enough for some application and most of computer vision tasks).

At any stage of decoding process tensor representation of image is tiled, which allows independent processing of parts of latent tensor. Latent space tiles are overlapping (in order to avoid artifacts on tile boundary). There is no overlap in residual tiles (since there is no convolution performed on residual).

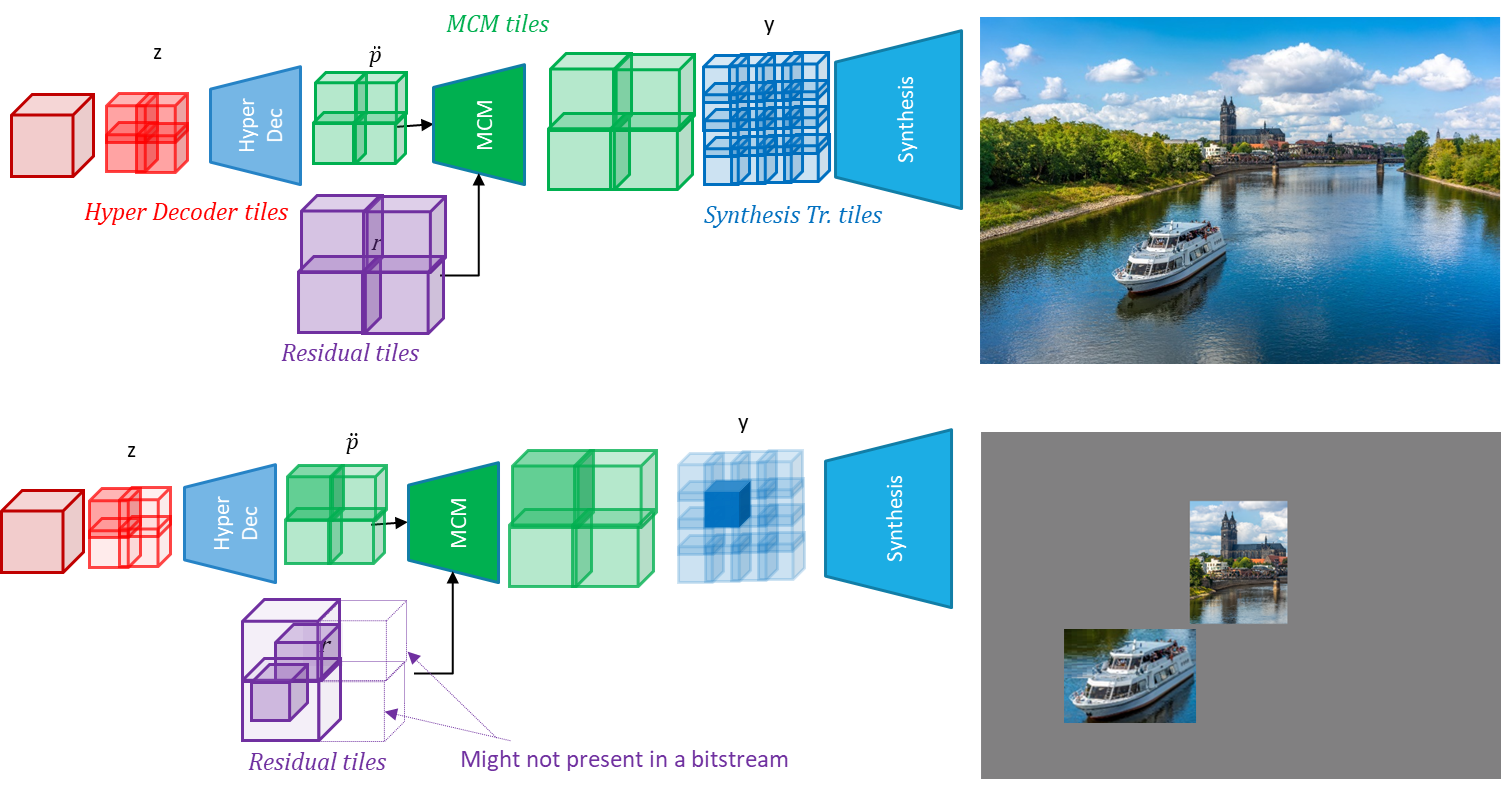


Figure 7 Region of interest extraction

The start and the end of each residual tile are known (offset table is signaled). Independent parsing of residual tiles, or parsing of just some of residual tiles is possible. There is an option to have not all but only some residual tiles in a bit-stream. This could be useful, for example, for VR application, so encoder which might not need to send whole big image, but just a view port(s). For region(s) of interest reconstruction only corresponding residual tile(s) should be to be parsed, then cropped (if needed) and only needed part(s) of the image will be reconstructed. The process for region of interest reconstruction is illustrated in Figure 7.

As shown in Figure 2 from the code stream the latent tensor representation of image reconstructed via entropy decoder, residual and latent domain prediction. And from latent tensor image can be reconstructed with difference quality. From JPEG AI CfP results JPEG got an evidence of beside image reconstruction some computer vision tasks (for example, image classification) can be performed directly from latent tensor (w/o image reconstruction). Training scripts of JPEG AI perform training of neural network with multi-branches decoder. Training scripts and can be modified on straight forward way to train one more additional decoder(s) for other applications: different computer vision and image processing tasks. This expected to be the major focus of JPEG AI version 2.

# Training and testing of JPEG AI.

Training set of JPEG AI consist of 120K patches extracted from more than 5K different camera captured images (all CC0 license) and 17K patches extracted from synthetic and screen content images. The training uses weighted sum of MSE and SSIM for distortion measure in loss-function. Training was always performed with more than one parties who cross-checked each other.

Testing was performed using test set of 50 camera captured images (Figure 8). Testing images were announced after CfP responses from 10 different companies were submitted.

For those images seven quality metrics: MS-SSIM, VMAF, VIF, psnrHVS, IW-SSIM, NLDP; FSIM were evaluated: The BD-rate relatively to the anchors was computed for all those seven metrics. For the simplicity only average of all those BD-rates is reported below. It must be noted that quality metrics for JPEG AI were carefully selected after study of correlation with MOS, collected during massive visual quality assessment experiments. Due to the lack of correlation with perceptual visual quality PSNR was not the quality metric in JPEG AI process development.

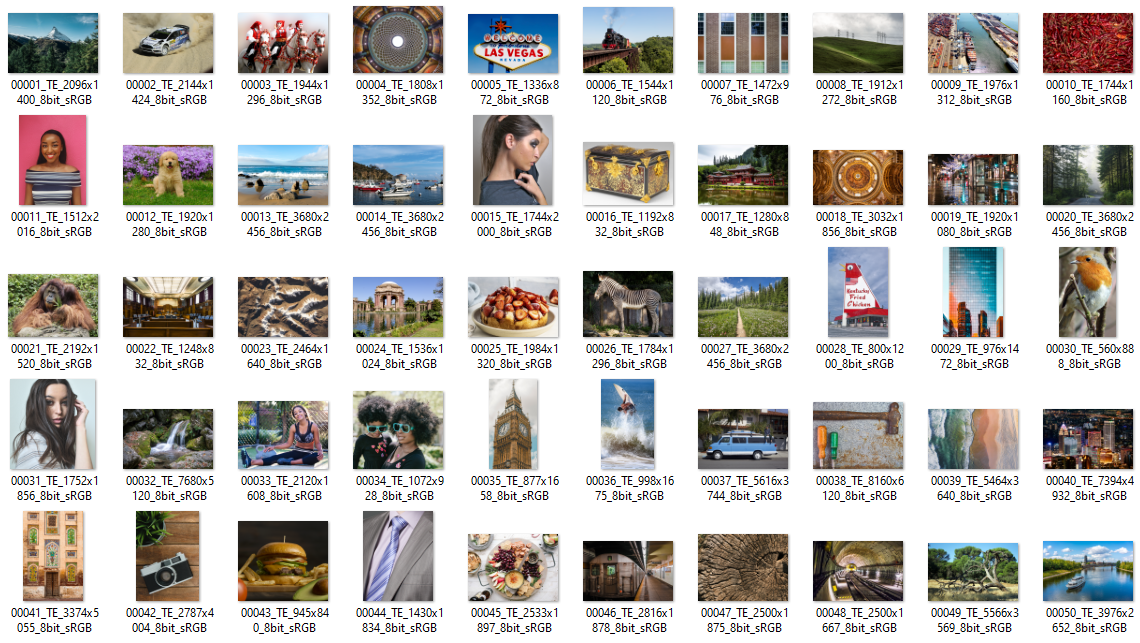
****

Figure 8 Testing set of 50 camera captured images

Additionally, to 50 camera captured images the performance of JPEG AI was regularly checked for synthetic images testing set, HDR testing set and even for crash data set. Crash data set is shown on Figure 9. All those images are very challenging for coding due to high gradients, ‘extreme’ colours. At early stage of JPEG AI development some artifacts been rarely observed for some images in ‘crash data set’, version submitted to DIS is free of artifacts: inclusion synthetic images into the training and some encoder side clipping helped to resolve all issues.

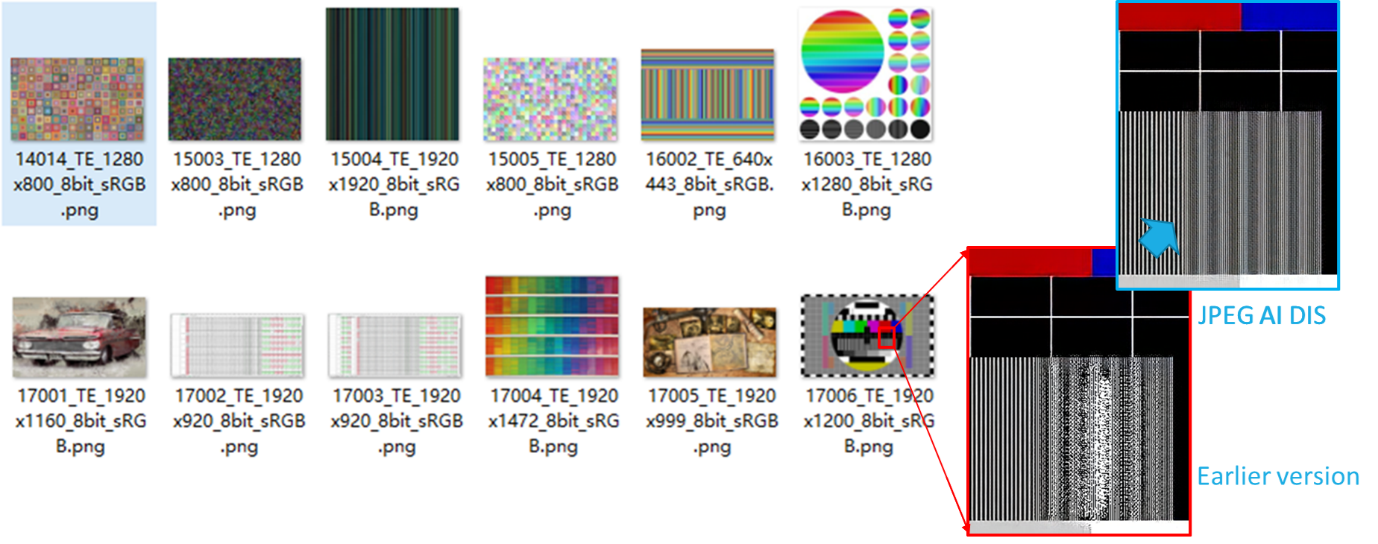


Figure 9 ‘Crash’ data set

# Performance of JPEG AI.

Performance test results are shown in Table 1. BD-rate performance numbers are reported for 50 images test set. BD-rate measured across five rate points: 0.12, 0.25, 0.5, 0.75, 1.0 bpp. Typical rate images are compressed for exchange inside different messengers is 0.5 bpp.

Table 1 Summary of performance test results under JPEG AI test conditions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test | BD-Rate  AVG | **Dec.** | | | **Enc.** |
| kMAC/pxl | Time GPU, × | Time CPU, × | Time GPU, × |
| **HM-16.20+SCM-8.8 - Intra** | **7.5%** | - | 0.8 (CPU) | 0.8 | 0.6 (CPU) |
| **VTM-11 - Intra** | **0.0%** | - | 1.0 (CPU) | 1.0 | 1.0 (CPU) |
| **VM6.1-Enc0Dec0-tools-off** | **-12.0%** | 8 | 0.36 | 1.1 | 0.0005 |
| **VM6.1-Enc0Dec0-tools-on** | **-16.2%** | 14 | 0.41 | 2.4 | 0.0011 |
| **VM6.1-Enc0Dec1-tools-off** | **-16.7%** | 23 | 0.38 | 2.1 | 0.0005 |
| **VM6.1-Enc0Dec1-tools-on** | **-20.2%** | 28 | 0.41 | 3.3 | 0.0011 |
| **VM6.1-Enc1Dec2-tools-off** | **-24.0%** | 214 | 0.61 | 28 | 0.0012 |
| **VM6.1-Enc1Dec2-tools-on** | **-27.0%** | 215 | 0.64 | 29 | 0.0018 |

For the reference performance of HEVC and VVC (still picture profile) is shown in Table 1. Namely HM-16.20+SCC-8.8 and VTM-11 were used. For most of 50 images in this test set 1.0 bpp corresponds to QP=20…25, visual quality is close to transparent. At such a high rates HM with screen content tools really powerful and BD-rate losses of HM relatively VTM (in metrics agreed in JPEG AI) is just 7.5%.

For JPEG AI results generation JPEG AI verification model VM6.1 was used (it corresponds the version of specification submitted for DIS ballot). HM and VTM are c++ code (with some SIMD), JPEG AI VM is python/pytorch code.

All simulations were performed on a same machine, which has CPU and GPU. There are no speed benefits of using GPU for VTM or HM, so they were encoded and decoded on CPU. VM6.1 (JPEG AI) was tested both on CPU and GPU.

Configuration marked as ‘tool-off’ indicates content adaptation and so no search on encoder side. There are also minimal number of tools in JPEG AI, which provide very limited content adaptation, they are enabled in ‘tools-on’ configuration (which gives additionally 3-4% BD-rate gain).

There are two encoders in JPEG AI reference SW. The simplest one ‘Enc0’ doesn’t have attention mechanism, it is ×2000 faster than VTM encoder. For example, 8K image encoding takes 5 seconds (same time needed for JPEG encoder Kakadu, v8.0.5). The more computationally complex encoder ‘Enc1’ has attention (which includes transformers), it is ×800 faster than VTM encoder.

The simplest decoder of JPEG AI (Dec0) has roughly same speed with VTM-Intra if both run on CPU. The middle complexity decoder (Dec 1) twice slower than VTM-Intra on CPU. The decoder with transformers (Dec2) is ×28 slower than VTM-Intra on CPU. If decoded on GPU all versions of JPEG AI decoder are ***faster*** than VTM-Intra decoder on CPU.

Bit exact reconstruction of image is NOT guaranteed, due to float point operations in synthesis transform. There is strong conformance point after residual reconstruction. There is no strong conformance point in reconstruction image. Float point operations are allowed in synthesis transform and latent domain prediction. Practically the different between two reconstructions on different platforms is no more than 1 (in sample value) in no more than fraction of percent of image pixels.

Visual quality examples are shown in Figure 10 for one of test images from JPEG AI test set and in Figure 11 for first frame of one of video in JVET test set.

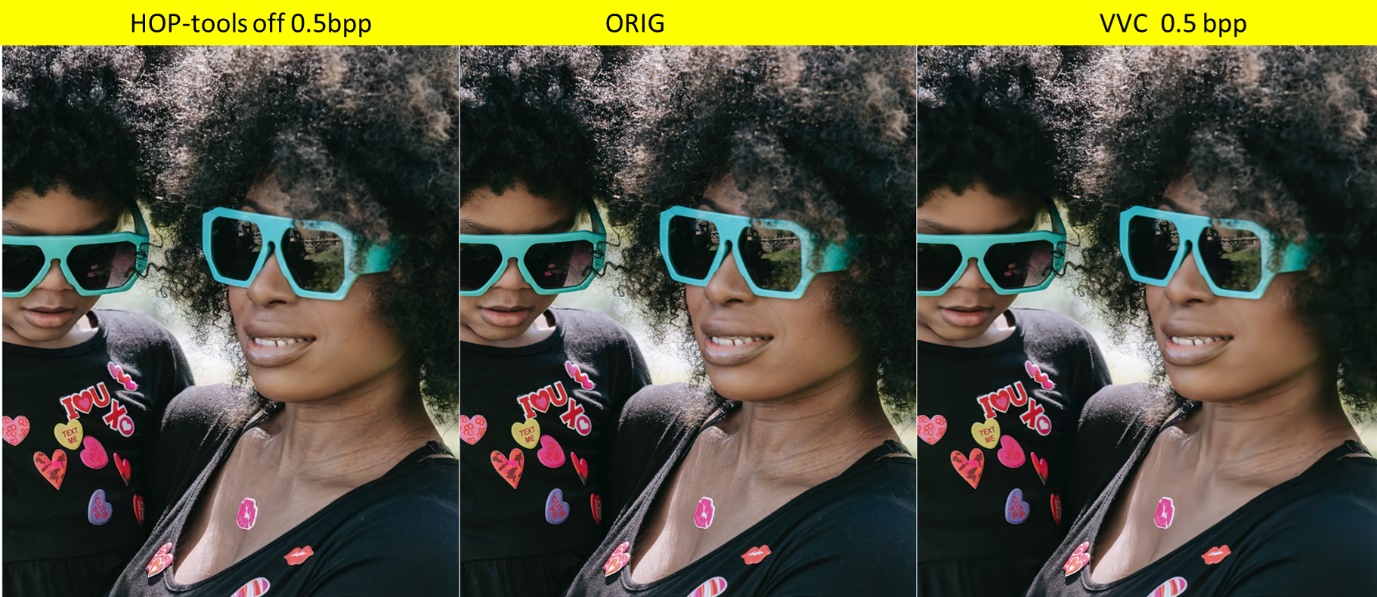


Figure 10 Visual quality comparison at the same rate (0.5 bpp) with VTM-Intra.

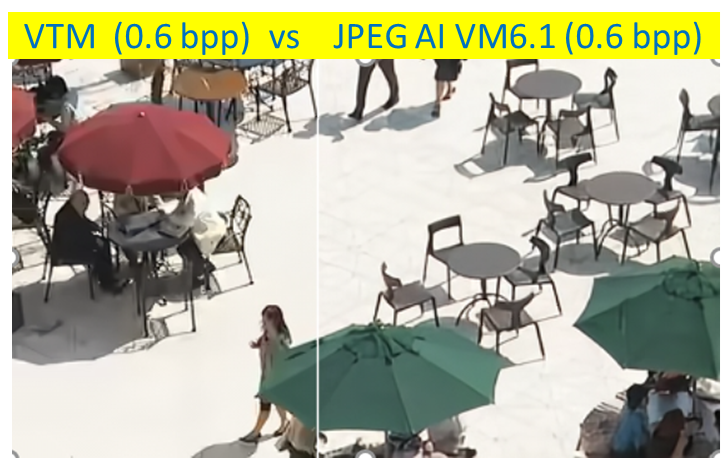


Figure 11 Visual quality comparison at the same rate (0.6 bpp) with VTM-Intra.

# Mobile device implementation

In parallel with standard development group studied possibility of implementation of JPEG AI on mobile device. At least two companies provided the trial implementation of decoder mobile devices using AI accelerator from Apple and Qualcomm. On relatively old smartphone with Qualcomm Snapdragon 8+ Gen1 decoding of 4K image takes no longer than 50 ms. The implementation stack: SNPE AI acceleration framework for neural network part, C++ (arithmetic coder), Java (gallery management).

# Conclusion

JPEG AI is entirely neural network-based image codec. JPEG AI DIS currently is under ISO ballot. All materials about JPEG AI can be found on <https://jpeg-git.lx.it.pt/jpegai> (accessible for JPEG members). Reference SW expected to be publicly available shortly. JPEG AI was developed targeting extremely fast images encoding, optimized for perceptual visual quality. Multiple functionalities such as progressive decoding, spatial random access are easy to realize for end-to-end AI-based codec architecture.