 ISO/IEC JTC 1/SC 29/WG 04 N0660

**ISO/IEC JTC 1/SC 29/WG 04  
MPEG Video Coding   
Convenorship: CN**

**Document type:** Output Document

**Title:** Encoder guidelines for MPEG immersive video

**Status:** Approved

**Date of document:** 2025-04-04

**Source:** ISO/IEC JTC 1/SC 29/WG 04

# Expected action: None

# Action due date: None

**No. of pages:** 1

**Email of Convenor:** yul@zju.edu.cn

**Committee URL:** <https://sd.iso.org/documents/ui/#!/browse/iso/iso-iec-jtc-1/iso-iec-jtc-1-sc-29/iso-iec-jtc-1-sc-29-wg-4>

**INTERNATIONAL ORGANIZATION FOR STANDARDIZATION**

**ORGANISATION INTERNATIONALE DE NORMALISATION**

**ISO/IEC JTC 1/SC 29/WG 04 MPEG VIDEO CODING**

**ISO/IEC JTC 1/SC 29/WG 04 N** **0660**

**April 2025, Online**

|  |  |
| --- | --- |
| **Title** | **Encoder guidelines for MPEG immersive video** |
| **Source** | **WG 04, MPEG Video Coding** |
| **Status** | **Approved** |
| **Serial Number** | **24083** |

# Introduction

ISO/IEC 23090-12 MPEG immersive video (MIV) [1, 2] is a standard that leverages legacy 2D video codecs for representing immersive video capable of 6DoF novel view synthesis. This document provides guidelines on how to encode video sub-bitstreams, and provides instructions for using publicly available video encoder software to encode V3C sample streams for MIV profiles: TMIV [3], HM [4], VVenC [5] and Kvazaar [6].

An MIV bitstream consists of the following sub-bitstreams:

* Exactly one common atlas sub-bitstream with view parameters
* One or more atlas sub-bitstreams with patch parameters
* One or more video sub-bitstreams per atlas

It is expected that the coding of the video sub-bitstreams is handled by a regular 2D video encoder, but there are some MIV-specific constraints that need to be addressed.

# General encoder guidelines

The MIV standard [1, 2] imposes requirements on sub-bitstreams to allow for parallel decoding and 6DoF novel view synthesis.

## Alignment of access units

The access unit concept is followed, whereby sub-bitstream access units of one frame order count form an MIV access unit [1:3.9]. An MIV decoder has to synchronize the decoded sub-bitstream access units to form the decoded MIV access unit. A coded MIV sequence [1:3.4] starts with an MIV independent random access (IRAP) access unit [1:3.11], followed by zero or more MIV access units [1:3.9]. Common atlas data and atlas data may persist for multiple frames, and this is achieved by signalling non-consecutive frame order counts.

When independently encoding sub-bitstreams, it is important that some constraints are followed, otherwise the resulting MIV bitstream will be invalid:

* Frame order counts of sub-bitstream access units that are present need to match between sub-bitstreams.
* Decoding order of sub-bitstream access units that are present need to match between sub-bitstreams.
* For an MIV IRAP access unit all sub-bitstream access units need to be present, and they need to be IRAP access units, so their frame order count is zero (0).

It is not a requirement that access units have the same reference frame structure.

The MPEG test models follow a regular intra period and GOP structure. Practical (commercial) video encoders may have a more adaptive frame structure and this can make it challenging to use off-the-shelve video encoders to encode MIV bitstreams.

## Visual usability information

The MIV standard specifies that all video components except for texture need to be encoded with full range and linear transfer function. This is also the case for packed video. To make sure that video decoders honor the intention of the coded video, each non-texture video sub-bitstream should include visual usability information (VUI) to signal full range and linear transfer characteristics.

## Profiles, tiers and levels

The MIV bitstream has profile, tier and level (PTL) information in the V3C parameter set (VPS), indicating what decoder capabilities are needed to handle the bitstream. A conformant bitstream does not exceed those requirements. Similarly, the video sub-bitstreams typically also have a PTL, and the video sub-bitstreams must be conformant.

# Encoder guidelines for TMIV

Encoding an MIV bitstream with TMIV [3] requires three steps:

1. Run the TMIV encoder or TMIV MPI encoder to output a partial MIV bitstream without video sub-bitstreams, raw video (YUV 4:2:0), and a JSON with video sub-bitstream (VSB) information.
2. Run multiple instances of a video encoder, one for each raw video file, using the VSB information to configure each video encoder, to output video sub-bitstream files.
3. Run the TMIV multiplexer to combine the partial MIV bitstream and video sub-bitstram files into a MIV bitstream.

Since version 24.0 of TMIV there is a script encode.py that combines all three steps and it calculates parameters for HM, Kvazaar or VVenC based on the TMIV (MPI) encoder configuration to ensure that the mentioned requirements are fulfilled. If a video encoder configuration is incompatible with the TMIV (MPI) encoder configuration then the script will give an error and stop.

The guideline is to use the script when possible because it implements the guidelines. When combining TMIV with a different video encoder, it is advised to study the script and the general encoding guidelines. MPEG contributions to extend the script to support different video encoders may be considered.

# Encoder guidelines for HM

The HM software provides some example configuration files that can be used. The parameters in Table 1 are relevant to the coding of MIV video sub-bitstreams using HM.

Table 1: Relevant HM parameters

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Value** | **Motivation** |
| Profile | main10 | Select HEVC Main 10 profile. |
| Level | calculate | HM does not automatically calculate the level value. |
| IntraPeriod | configurable | This parameter determines which frames are IRAP frames. Its value has to match with the intraPeriod parameter of TMIV. |
| ReWriteParamSetsFlag | 0, 1 | Control if the VPS is only written on the first frame or on each IRAP frame. Its value has to match with the rewriteParameterSets parameter of TMIV. |
| ConformanceWindowMode | 1 | Some video components, especially occupancy maps, may not have a smooth size. This parameter ensures correct decoding in those cases. No harm to always set it. |
| VuiParametersPresent\* | 1 | Enable the VUI. |
| VideoSignalTypePresent\* | 1 | Enables presence of full range flag in VUI. |
| VideoFullRange\* | 1 | Signals full range. |
| TransferCharacteristics\* | 8 | Signals linear transfer. |

\*: Only for non-texture video sub-bitstreams.

# Encoder guidelines for Kvazaar

The command-line parameters in Table 2 are relevant to the coding of MIV video sub-bitstreams using Kvazaar.

Table 2: Relevant Kvazaar command-line parameters

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Value** | **Motivation** |
| --level | calculate | Kvazaar does not automatically calculate the level value. |
| --intra-period | configurable | This parameter determines which frames are IRAP frames. Its value has to match with the intraPeriod parameter of TMIV. |
| --gop | 8 or 16 | The intraPeriod parameter of TMIV has to be divisible by the value of this parameter. |
| --vps-period | 0, 1 | Control if the VPS is only written on the first frame or on each IRAP frame. Its value has to match with the rewriteParameterSets parameter of TMIV. |
| --range\* | pc | Signals full range. |
| --transfer\* | linear | Signals linear transfer. |

\*: Only for non-texture video sub-bitstreams.

For packed video sub-bitsteams it is advised to use a ROI file to specify delta QP's per region because typically occupancy and geometry need a lower QP than texture. As an example, for a 2048 x 5440 packed frame with a 2048 x 4352 texture region and two 1024 x 1088 geometry regions, forming a 1024 x 2176 geometry frame, the following ROI definition can be used to indicate a delta QP of -17 for geometry and a delta QP of 0 for texture:

32 85

0 ... 0

⁞ ⁞

32 x 68: texture region

0 ... 0

-17 ... -17

32 x 17: geometry regions

⁞ ⁞

-17 ... -17

It was experimentally verified in the 150th MPEG meeting that with TMIV and Kvazaar it is possible to use packed video without a loss in IV-SSIM BD-rate [6] as compared to encoding the same pixel data as multiple video components.

# Encoder guidelines for VVenC

The VVenC software has two command-line interfaces. The vvencapp CLI is an easy-to-use interface for general purpose. The vvencFFapp CLI aimed to be a drop-in replacement for the VVC test model (VTM) that uses similar configuration files. It does not appear to be possible to use vvencapp for encoding valid MIV video sub-bitstreams because there is less control on frame structure and VUI.

VVenC provides some example vvencFFapp configuration files that can be used. The parameters in Table 3 are relevant to the coding of MIV video sub-bitstreams using vvencFFapp.

Table 3: Relevant VVenC parameters

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Value** | **Motivation** |
| Profile | auto | No need to calculate. |
| Level | auto | No need to calculate. |
| IntraPeriod | configurable | This parameter determines which frames are IRAP frames. Its value has to match with the intraPeriod parameter of TMIV. |
| GOPSize | 1, 16 or 32 | The intraPeriod parameter of TMIV has to be divisible by the value of this parameter. |
| ReWriteParamSets | 0, 1 | Control if the VPS is only written on the first frame or on each IRAP frame. Its value has to match with the rewriteParameterSets parameter of TMIV. |
| VuiParametersPresent\* | 1 | Enable the VUI. |
| VideoFullRange\* | 1 | Signals full range. |
| TransferCharacteristics\* | linear | Signals linear transfer. |
| CrCbOffset\*\* | 6 | Use finer quantization for chroma planes. |
| CrQpOffset\*\* | 6 | Use finer quantization for chroma planes. |

\*: Only for non-texture video sub-bitstreams.

\*\*: Only for geometry video sub-bitstreams when colourized geometry is enabled.

# References

[1] ISO/IEC 23090-12:2023 Information technology — Coded representation of immersive media, Part 12: MPEG immersive video, to be replaced by [2],  
url: <https://www.iso.org/standard/79113.html>

[2] ISO/IEC FDIS 23090-12 Information technology — Coded representation of immersive media, Part 12: MPEG immersive video,  
url: <https://www.iso.org/standard/87643.html>

[3] Test model for MPEG immersive video 24.0,  
url: <https://gitlab.com/mpeg-i-visual/tmiv/-/tree/v24.0>

[4] HM reference software for HEVC 18.0,  
url: <https://vcgit.hhi.fraunhofer.de/jvet/HM/-/tree/HM-18.0>

[5] Fraunhofer Versatile Video Encoder (VVenC) 1.12.0,  
url: <https://github.com/fraunhoferhhi/vvenc/tree/v1.12.0>

[6] Kvazaar 2.3.1,  
url: <https://github.com/ultravideo/kvazaar/tree/v2.3.1>