ISO/IEC JTC 1/SC 29/WG 04 N0411

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

## MPEG Video Coding

## Convenorship: CN

|  |  |
|---|---|
| **Document type:** | Output Document |
| **Title:** | IV-PSNR 5.0 manual |
| **Status:** | Approved |
| **Date of document:** | 2023-11-03 |
| **Source:** | ISO/IEC JTC 1/SC 29/WG 04 |
| **Expected action:** | None |
| **Action due date:** | None |
| **No. of pages:** | 9 (without cover page) |
| **Email of Convenor:** | yul@zju.edu.cn |
| **Committee URL:** | https://isotc.iso.org/livelink/livelink/open/jtc1sc29wg4 |

**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 0411**
**October 2022, Hannover, DE**

| | |
|---|---|
| **Title** | **IV-PSNR 5.0 manual** |
| **Source** | **WG 04, MPEG Video Coding** |
| **Status** | **Approved** |
| **Editor** | **Adrian Dziembowski** |
| **Serial Number** | **23279** |

# 1. Introduction

IV-PSNR [Dziembo22] is a PSNR-based objective quality metric adapted for Immersive Video applications. Compared to PSNR, two major modifications were added: Corresponding Pixel Shift and Global Color Difference. Corresponding Pixel Shift eliminates the influence of a slight shift of objects' edges caused by reprojection errors. Global Color Difference reduces the influence of different color characteristics of different input views.

Version 5.0 introduces new application parameters available at runtime (InvalidPelActn and NameMismatchActn), as well as several performance improvements. When masked mode is not used, output of IV-PSNR v5.0 is exactly the same, as for IV-PSNR v2.1.1 and higher.

Detailed description of the IV-PSNR metric can be found in [Dziembo22]. Below, the general and simplified idea of the IV-PSNR is presented.

IV-PSNR for YUV file is calculated as:

$$\mathrm{IVPSNR}_{\mathrm{YUV}} = \frac{\sum_{c=0}^{2} \mathrm{IVPSNR}(c) \cdot \mathrm{CCW}(c)}{\sum_{c=0}^{2} \mathrm{CCW}(c)},$$

where $\mathrm{CCW}(c)$ is the Color Component Weight for each color component $c$ and $\mathrm{IVPSNR}(c)$ is the IV-PSNR for that component:

$$\mathrm{IVPSNR}(c) = 10 \cdot \log\left(\frac{\mathrm{MAX}^2}{\mathrm{IVMSE}(c)}\right),$$

where MAX is the maximum value of the color component (e.g., 1023 for 10-bit video) and:

$$\mathrm{IVMSE}(c) = \frac{1}{W \cdot H} \sum_{y=0}^{H-1} \sum_{x=0}^{W-1} \min_{\substack{x_R \in [x-\mathrm{CPS}, x+\mathrm{CPS}] \\ y_R \in [y-\mathrm{CPS}, y+\mathrm{CPS}]}} \left(c_T(x,y,c) - c_R(x_R,y_R,c) + \mathrm{GCD(c)}\right)^2 ,$$

where $W$ and $H$ are width and height of the image, $c_T(x, y, c)$ and $c_R(x, y, c)$ are values of color component $c$ in the position $(x, y)$ in the test image and the reference image, respectively, CPS is the maximum Corresponding Pixel Shift between reference and test image, and GCD is the Global Color Difference for component $c$:

$$\mathrm{GCD}(c) = \max\left(\frac{1}{W \cdot H}\sum_{y=0}^{H-1}\sum_{x=0}^{W-1}\big(c_R(x, y, c) - c_T(x, y, c)\big)\ , \mathrm{MUD}(c)\right),$$

where $\mathrm{MUD}(c)$ is the Maximum Unnoticeable Difference for color component $c$.

In order to provide better quality assessment for omnidirectional video, WS-PSNR technique [Sun17] was applied (however, in the current version of the IV-PSNR software only the equirectangular projection is supported):

$$\text{WS-IVMSE}(c) = \frac{\sum_{y=0}^{H-1}\sum_{x=0}^{W-1} \min_{\substack{x_R \in [x-\mathrm{CPS}, x+\mathrm{CPS}] \\ y_R \in [y-\mathrm{CPS}, y+\mathrm{CPS}]}} \big(c_T(x, y, c) - c_R(x_R, y_R, c) + \mathrm{GCD}(c)\big)^2 \cdot w_{x,y}}{\sum_{y=0}^{H-1}\sum_{x=0}^{W-1} w_{x,y}},$$

where weight $w_{x,y}$ is calculated as:

$$w_{x,y} = \cos\frac{\left(y + 0.5 - \frac{H}{2}\right) \cdot \pi}{H},$$

where $x, y$ is a position of the pixel in ERP image and $H$ is the height of this image.

$\mathrm{CCW}(c)$, $\mathrm{MUD}(c)$ and $\mathrm{CPS}$ values are predefined:

- $\mathrm{CCW}(c)$:
  - $\mathrm{CCW}(0) = 1$ (luma component),
  - $\mathrm{CCW}(1) = 0.25$ (1st chroma component),
  - $\mathrm{CCW}(2) = 0.25$ (2nd chroma component),
- $\mathrm{MUD}(c) = 1\%$ for all the color components,
- $\mathrm{CPS} = 2$.

IV-PSNR is calculated separately for each frame of the sequence. In the end, the mean IV-PSNR value is returned.

The IV-PSNR quality metric is based on PSNR, therefore, the higher the number, the better is the quality.

# 2. Software manual

IV-PSNR v5.0 accepts commandline parameters listed in section 2.1:

## 2.1 Commandline parameters

### General parameters

| Cmd | ParamName | Description |
|---|---|---|
| -i0 | InputFile0 | YUV file path – reference |
| -i1 | InputFile1 | YUV file path – tested |
| -w | PictureWidth | Width of sequence |
| -h | PictureHeight | Height of sequence |
| -bd | BitDepth | Bit depth (optional, default: 8, up to 14) |
| -cf | ChromaFormat | Chroma format (optional, default: 420) [420, 444] |
| -s0 | StartFrame0 | Start frame (optional, default: 0) |
| -s1 | StartFrame1 | Start frame (optional, default: 0) |
| -l | NumberOfFrames | Number of frames to be processed (optional, default: -1 = all) |
| -o | OutputFile | Output file path (optional) |

### Equirectangular parameters

| Cmd | ParamName | Description |
|---|---|---|
| -erp | Equirectangular | Equirectangular sequence (flag, default disabled) |
| -lor | LonRangeDeg | Longitudinal range of ERP sequence [°] (optional, default: 360) |
| -lar | LatRangeDeg | Lateral range of ERP sequence [°] (optional, default: 180) |

### IV-PSNR specific parameters

| Cmd | ParamName | Description |
|---|---|---|
| -sr | SearchRange | IV-PSNR search range around center point (optional, default: 2 = 5×5) |
| -cws | ComponentWeights | IV-PSNR component weights ("Lm:Cb:Cr:0" – per component integer weight, default: "4:1:1:0", quotes required, requires USE_RUNTIME_CMPWEIGHTS = 1) |
| -unc | UnnoticeableCoef | IV-PSNR unnoticeable color difference threshold coeff ("Lm:Cb:Cr:0" – per component coeff, default: "0.01:0.01:0.01:0", quotes required) |

### WS-PSNR specific parameters

| Cmd | ParamName | Description |
|---|---|---|
| -ws8 | Legacy8bitWSPSNR | Use 1020 as peak value for 10-bps videos in WSPSNR metric (provides compatibility with original WSPSNR implementation, optional, default: 1) |

### Application parameters

| Cmd | ParamName | Description |
|---|---|---|
| -t | NumberOfThreads | Number of worker threads if compiled with OpenMP (optional, default: -1 = all, suggested 4-8) |
| -ilp | InterleavedPic | Use additional image buffer with interleaved layout (improves performance at a cost of increased memory usage, optional, default: 1) |
| -ipa | InvalidPelActn | Select action taken if invalid pixel value (larger than [(1<<BitDepth)-1]) is detected (optional, default STOP) [SKIP – disable pixel value checking, WARN – print warning and ignore, STOP – stop execution, CNCL – try to conceal by clipping to bit depth range] |
| -nma | NameMismatchActn | Select action taken if parameters derived from filename are different than provided as input parameters. Checks resolution, bit depth and chroma format. (optional, default WARN) [SKIP – disable checking, WARN – print warning and ignore, STOP – stop execution] |
| -v | VerboseLevel | Verbose level (optional, default: 2), cf. section 2.3 |

### External config file

| Cmd | ParamName | Description |
|---|---|---|
| -c | | Valid path to external config file – in INI format (optional), cf. section 2.5 |

### Masked mode parameters

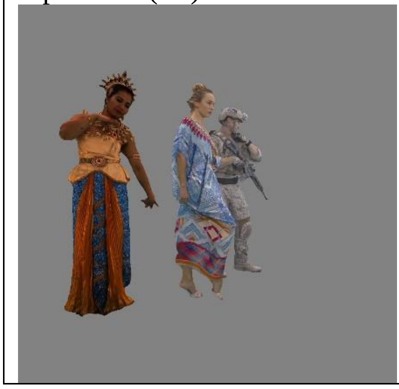
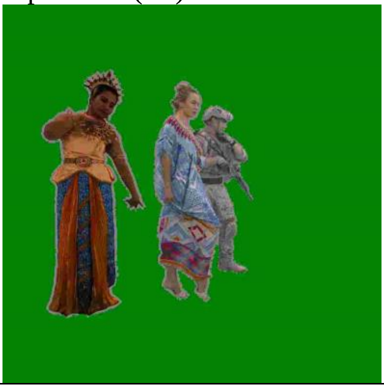
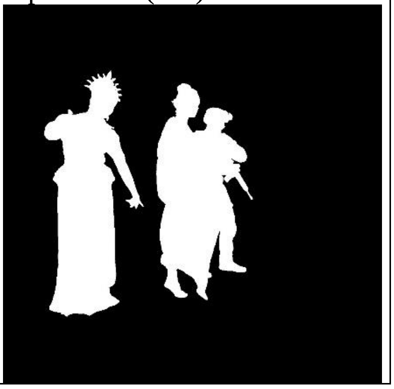
| Cmd | ParamName | Description |
|---|---|---|

| | | |
|---|---|---|
| -im | InputFileM | YUV file path – mask (optional, same resolution as InputFile0 and InputFile1) |
| -bdm | BitDepthM | Bit depth for mask (optional, default: BitDepth, up to 16) |
| -cfm | ChromaFormatM | Chroma format for mask (optional, default: ChromaFormat) [400, 420, 444] |

- The commandline parameters are position-intependent.
- When no parameters are used, syntax help is outputted.

## 2.2 Masked mode

Optional mode of the IV-PSNR 5.0 allows to calculate IV-PSNR value only for specified areas. In order to use masked mode, InputFileM (-im) parameter has to be set, indicating a path of mask YUV file.

| | | |
|---|---|---|
| InputFile0 (-i0):<br> | InputFile1 (-i1):<br> | InputFileM (-im):<br> |

In an example above, the IV-PSNR value is calculated only for occupied pixels (as indicated by mask), so different color of the unoccupied background does not impact outputted quality.

**Requirements and notes**

- Resolution of mask file has to be identical as input file.
- Allowed mask values are 0 (interpreted as inactive pixel) and (1<<BitDepthM)-1) (interpreted as active pixel). Behavior for other values is undefined at this moment.
- The data processing functions for masked mode are not implemented with the use of SIMD instructions.

## 2.3 Verbose levels

| Value | Printed data |
|---|---|
| 0 | final PSNR, WS-PSNR, IV-PSNR values only |
| 1 | 0 + configuration + detected number of frames |
| 2 | 1 + argc/argv + frame level PSNR, WS-PSNR, IV-PSNR |
| 3 | 2 + computing time (LOAD, PSNR, WS-PSNR, IV-PSNR)<br>(uses high resolution clock, could slightly slow down computations) |
| 4 | 3 + IV-PSNR specific debug data (GlobalColorShift, R2T+T2R) |

## 2.4 Compile-time parameters

| Parameter name | Default value | Description |
|---|---|---|
| USE_SIMD | 1 | use SIMD (to be precise... use SSE 4.1 or AVX2) |
| USE_KBNS | 1 | use Kahan-Babuška-Neumaier floating point summation algorithm (reduces accumulation errors) |
| USE_RUNTIME_CMPWEIGHTS | 1 | use component weights provided at runtime |

## 2.5 Config file example

```
InputFile0      = "SA_ref.yuv"
InputFile1      = "SA_test.yuv"
PictureWidth    = 4096
PictureHeight   = 2048
BitDepth        = 10
ChromaFormat    = 420
VerboseLevel    = 3
OutputFile      = "IV-PSNR.txt"
```

## 2.6 Compilation requirements

The IVPSNR v5.0 software uses following external components:

- “Formatting library for C++” (libfmt) – distributed under BSD license and included in IVPSNR source package.

In order to build the software, the ISO C++17 conformant compiler is required.

# 3. Building

Building the IV-PSNR software requires using CMake (https://cmake.org/) and C++17 conformant compiler (e.g., GCC >= 8.0, clang >= 5.0, MSVC >= 19.15). For user convenience, a set of scripts for easy "one click" configure & build is prepared:

- configure_and_build.bat - for Windows users,
- configure_and_build.sh - for Unix/Linux users.

The IV-PSNR application and its build system is designed to create the fastest possible binary. On x86-64 microarchitectures the build system can create four version of compiled application, each optimized for one predefined x86-64 Microarchitecture Feature Levels [x86-64, x86-64-v2, x86-64-v3, x86-64-v4] (defined in https://gitlab.com/x86-psABIs/x86-64-ABI). The final binary consists of these four optimized variants and a runtime dynamic dispatcher. The dispatcher uses the CPUID instruction to detect available instruction set extensions and selects the fastest possible code path.

The IV-PSNR CMake project defines the following parameters:

| **Variable** | **Type** | **Description** |
|---|---|---|
| PMBB_GENERATE_MULTI _MICROARCH_LEVEL_BINARIES | BOOL | Enables generation of multiple code paths, optimized for each variant of x86-64 Microarchitecture Feature Levels. |
| PMBB_GENERATE_SINGLE_APP _WITH_WITH_RUNTIME_DISPATCH | BOOL | Enables building single application with runtime dynamic dispatch. Requires PMBB_GENERATE _MULTI_MICROARCH_LEVEL_BINARIES=True. |
| PMBB_GENERATE_DEDICATED _APPS_FOR_EVERY_MFL | BOOL | Enables building multiple applications, each optimized for selected x86-64 Microarchitecture Feature Level. Requires PMBB_GENERATE_MULTI_MICROARCH _LEVEL_BINARIES=True. |
| PMBB_BUILD_WITH_MARCH _NATIVE | BOOL | Enable option to force compiler to tune generated code for the micro-architecture and ISA extensions of the host CPU. Conflicts with PMBB_GENERATE_MULTI_MICROARCH_LEVEL _BINARIES. Generated binary is not portable across different microarchitecures. |

## 4. Examples

1. IV-PSNR of SA_ref.yuv and SA_test.yuv. Sequence resolution is 4096×2048, YUV420, 10 bits per sample. Sequence format is ERP. Mean IV-PSNR calculated for the first 20 frames will be written into IV-PSNR.txt:

```
IV-PSNR -i0 SA_ref.yuv -i1 SA_test.yuv -w 4096 -h 2048 -bd 10 -erp -l 20 -o IV-PSNR.txt
```

2. IV-PSNR of SD_ref.yuv and SD_test.yuv. Sequence resolution is 2048×1088, YUV420, 8 bits per sample. Sequence format is perspective. Mean IV-PSNR calculated for all frames will be written into results.txt:

```
IV-PSNR -i0 SD_ref.yuv -i1 SD_test.yuv -o results.txt -w 2048 -h 1088
```

3. IV-PSNR of SC_ref.yuv and SC_test.yuv. Sequence resolution is 4096×2048, YUV420, 10 bits per sample. Sequence format is ERP, with lateral range equal to 90°. Mean IV-PSNR calculated for 5 frames (frames 0-4 of reference video and 10-14 of test video) will be written into o.txt:

```
IV-PSNR -i0 SC_ref.yuv -i1 SC_test.yuv -w 4096 -h 2048 -erp -lar 90 -l 5 -s1 10 -o o.txt
```

4. Using config file:

```
IV-PSNR -c "config.cfg"
```

5. Using external config file with some parameters added/overridden:

```
IV-PSNR -c "config.cfg" -v 1 -t 4
```

## 5. Software

| | |
|---|---|
| MPEG Git Repository: | http://mpegx.int-evry.fr/software/MPEG/MIV/RS/IVPSNR |
| Public read-only access: | https://gitlab.com/mpeg-i-visual/ivpsnr |
| Software coordinator: | Adrian Dziembowski, adrian.dziembowski@put.poznan.pl |

## 6. Usage and citation

Please cite reference [Dziembo22] when using IV-PSNR.

## 7. References

[Dziembo22] A. Dziembowski, D. Mieloch, J. Stankowski, and A. Grzelka, "IV-PSNR – the objective quality metric for immersive video applications," IEEE Transactions on Circuits and Systems for Video Technology, 2022, DOI: 10.1109/TCSVT.2022.3179575.

[M48093] A. Dziembowski, M. Domański, "[MPEG-I Visual] Objective quality metric for immersive video," ISO/IEC JTC1/SC29/WG11 MPEG/M48093, July 2019, Göteborg, Sweden.

[M54279] J. Stankowski, A. Dziembowski,
"[MPEG-I Visual] Fast implementation of IV-PSNR software,"
ISO/IEC JTC1/SC29/WG11 MPEG/M54279, July 2020, Online.

[M54896] J. Stankowski, A. Dziembowski,
"Even faster implementation of IV-PSNR software,"
ISO/IEC JTC1/SC29/WG04 MPEG VC/M54896, October 2020, Online.

[M55752] A. Dziembowski, J. Stankowski,
"Slightly faster IVPSNR,"
ISO/IEC JTC1/SC29/WG04 MPEG VC/M55752, January 2021, Online.

[M59974] J. Stankowski, A. Dziembowski,
"Improved IV-PSNR software,"
ISO/IEC JTC1/SC29/WG04 MPEG VC/M59974, July 2022, Online.

[M64727] J. Stankowski, A. Dziembowski,
"Optimized IV-PSNR software with invalid pixel detection,"
ISO/IEC JTC1/SC29/WG04 MPEG VC/M64727, Oct. 2023, Hannover, DE.

[Sun17] Y. Sun, A. Lu, L. Yu,
"Weighted-to-Spherically-Uniform Quality Evaluation for Omnidirectional Video," IEEE Signal Processing Letters 24.9(2017):1408-1412.

# 8. Changelog

**v5.0** [M64727]:

- general overhaul of entire software structure,
- new cMake-based build system with simultaneous build of four variants of x86-64 Microarchitecture Feature Level and runtime dynamic dispatch,
- added unit tests for basic data processing routines,
- added detection invalid pel values (higher than (1<<BitDepth) - 1) and possibility to choose taken action (see InvalidPelAction parameter),
- added warning for settings influencing performance or breaking conformance with IV-PSNR metric defined in [M54279],
- added detection of mismatch between file name and provided parameters (resolution, bit depth and chroma format),
- added usage of hugepages on Linux-based systems (using madvise),
- added support for chroma format 4:2:2,
- more data processing functions implemented using AVX2,
- wider SIMD (AVX512) implementation for some data processing functions.

**v4.0** [M59974]:

- SIMD (SSE 4.1) implementation of IV-PSNR calculation (for interleaved picture buffers),
- wider SIMD (AVX2) implementation for most data processing functions,
- runtime adjustable component weights for IV-PSNR metric,
- adjustable search range for IV-PSNR metric,
- adjustable unnoticeable color difference threshold coeff for IV-PSNR metric,
- reading parameters from config file,
- protection against StartFrame >= DetectedFrames,
- writing error messages to stdout and stderr,
- non-performance critical parameters moved from compile-time to run-time selection,
- added mask file option.

**v3.0** [M55752]:

- enabled INTERPROCEDURAL_OPTIMIZATION and assumed x86-64 Microarchitecture Feature Level >= x86-64-v2,
- new implementation picture I/O,
- reduced filesystem burden (avoid repetitive open-seek-read-close cycles),
- use of interleaved picture layout for IVPSNR calculation,
- SIMD (SSE 4.1) implementation for most data processing functions,
- dedicated thread pool instead of OpenMP directives (due to high OpenMP overhead).

**v2.1.1** (no reference):

- bug fixes.

**v2.1** [M54896]:

- support for parallel processing (using OpenMP),
- addition of PSNR and WS-PSNR [Sun17] values outputting,
- fixed WS-weight calculation for ERP sequences with non-180 lateral range,
- changed commandline arguments formatting,
- addition of detection of corrupted YUV files,
- change in compile-time parameters:
  - VERBOSE_LEVEL is now a commandline parameter,
  - WSPSNR_PEAK_VALUE_8BIT flag added (default: enabled), when enabled, the signal peak value is set to 255 << (BitDepth – 8). Otherwise, it is equal to 2^BitDepth – 1.

**v2.0** [M54279]:

- addition of (rOff) and (tOff) commandline parameters,
- removal of redundant GCD calculations,

- usage of uint16_t data type and 4:4:4 chroma format for internal picture storage,
- new implementation of pixel-level processing steps,
- reduction of filesystem burden by coalescing read,
- detection of read errors – causes application to exit returning EXIT_FAILURE,
- implementation of Kahanand-Babuska-Neumaier accumulation,
- improved conversion of 8bps input sequences,
- improved interpolation for input sequences with 4:2:0 chroma format,
- addition of 3 compile-time parameters:
  - VERBOSE_LEVEL – controls number of per-frame printing; default = 0,
  - USE_KBNS – enables the Kahan-Babuska-Neumaier accumulation; default: enabled,
  - USE_FIXED_WEIGHTS – enables faster 5×5 block search with fixed component weight (equal to 4:1:1); default = enabled,
- fixed possibility of reading from unallocated memory region during 5×5 block search,
- fixed GCD values rounding and clipping.

**v1.0** [M48093].