O1S-VQA

User Manual

Version 1.1

support@o1stream.com

COPYRIGHT

DISCLAIMER

GoMax provides this document “as is”, without warranty of any kind, neither expressed nor implied, including, but not limited to, the particular purpose. GoMax may make improvements and/or changes in this document or in the product described in this document at any time. This document could include technical inaccuracies or typographical errors.

TRADEMARKS

O1Stream® is a trademark of GoMax Electronics Inc. Other names mentioned in this document are trademarks/registered trademarks of their respective owners.

REVISION HISTORY

Revision	Release Date	Summary
V1.1		Enriched edition with introduction, diagrams, and SOPs

1 Introduction

The O1Stream O1S-VQA (VQA) is a desktop application for evaluating how faithfully a video device, codec, or signal path reproduces a known reference. It was built by GoMax Electronics on top of MATLAB and bundles every common quality-assessment metric (PSNR, SSIM, VMAF, ΔE2000) together with a complete capture, alignment, and reporting workflow.

A typical VQA session consists of three stages. First, the operator builds a synchronized reference video using the Preamble function, which inserts three sync frames at the head of the source and stamps every following frame with a four-digit number. Second, the reference video is played through the device under test and recaptured using either a PCIe capture card or a USB capture stick. Third, the captured frames are compared frame-by-frame against the reference, and the analyzer reports per-frame and aggregate quality metrics together with diagnostic plots and PDF reports.

This manual is organized in three layers. Chapters 2 to 5 document the basic operation menus (Viewer, Tools, Measure, Capture, Study, Help) with screenshots of every dialog and a complete worked example of capturing 4K60 video. Chapters 6 to 9 cover the measurement theory: what each metric actually measures, why no single metric is sufficient, and how to read the curves the analyzer produces. Chapters 10 to 16 document the additional features that were added after the first release of the manual (Long PSNR Measurement, Color Vector Analysis, the eleven Advanced Pattern Generators, the Bandwidth Reference tools) and end with a set of standard operating procedures for the most common evaluation scenarios.

1.1 What This Manual Covers

Operation: how to use every menu in the application, with screenshots of every dialog.

Theory: the formulas behind each metric and the perceptual interpretation of the values they produce.

Architecture: the recommended hardware setup for trustworthy measurements, including HDMI splitter requirements, capture-card considerations, and source-player guidance.

Recipes: end-to-end SOPs for evaluating a USB capture card, comparing two encoders, running long stability tests, measuring colour accuracy of a splitter or scaler, and diagnosing frame drops.

1.2 Test Architecture at a Glance

Two architectures are common, and the analyzer supports both. In the compressed-video case the test video with preamble is sent through a video encoder and decoder before being recaptured; this isolates the codec under test. In the capture-device case the test video with preamble is sent directly through the capture device and recorded by OBS or by the bundled OpenCV script; this isolates the capture path.

Compressed video measurement case:

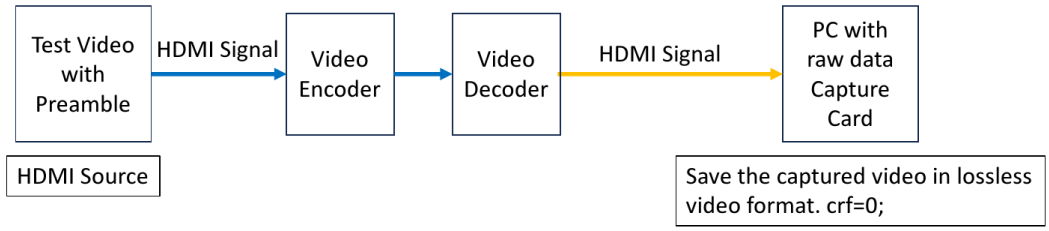


Figure 1.1 — Compressed-video measurement architecture. Source: bundled measurement_process.pdf.

Capture video measurement case:

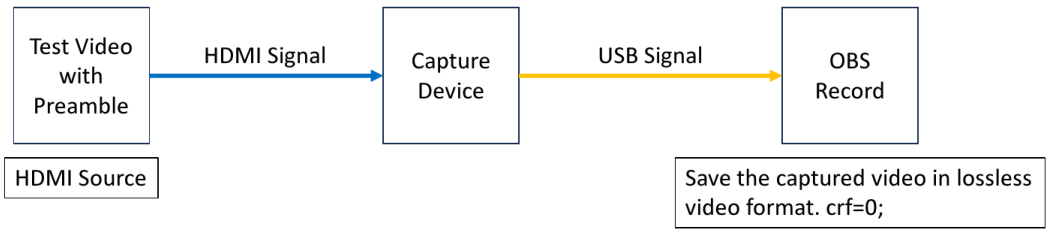


Figure 1.2 — Capture-device measurement architecture. Source: bundled measurement_process.pdf.

1.3 The Preamble and Frame-Number Stamp

Both architectures depend on a synchronized reference. The Preamble function inserts three black "1 / 2 / 3" sync frames at the head of the master video and then stamps every following frame with a four-digit number in the bottom-right corner. The capture pipeline then OCR-reads that number to detect frame drops and to align reference and distorted sequences on a per-frame basis. The illustration below shows the resulting frame sequence as written to disk.

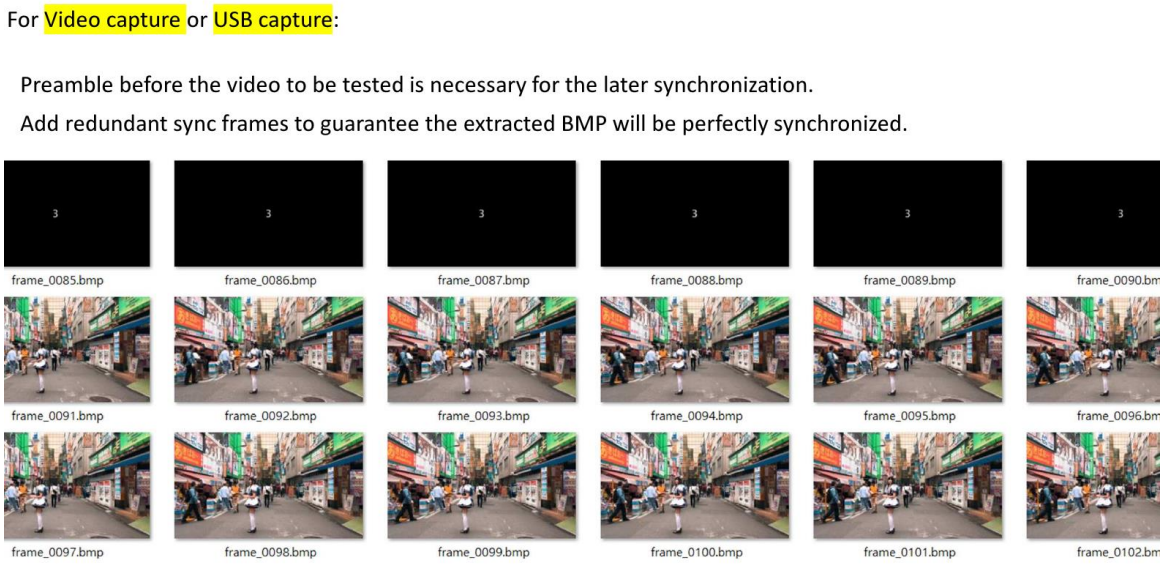
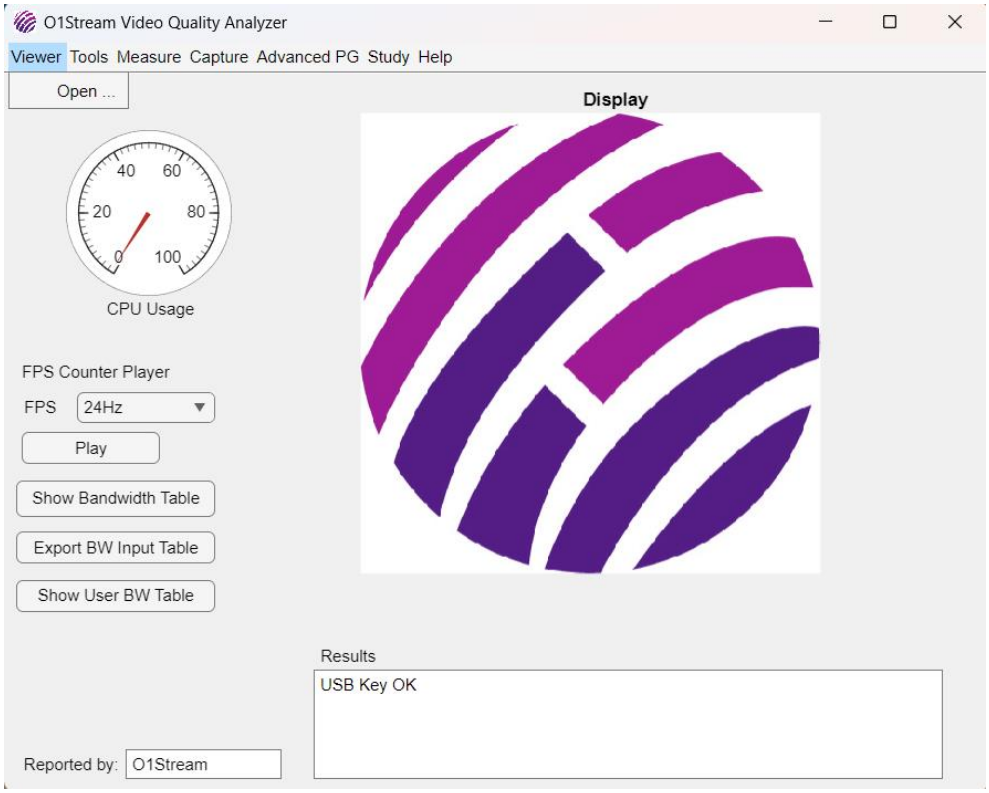


Figure 1.3 — Output of the Preamble function. Three sync frames followed by numbered content frames. Source: bundled measurement_process.pdf.

1.4 License and Versioning

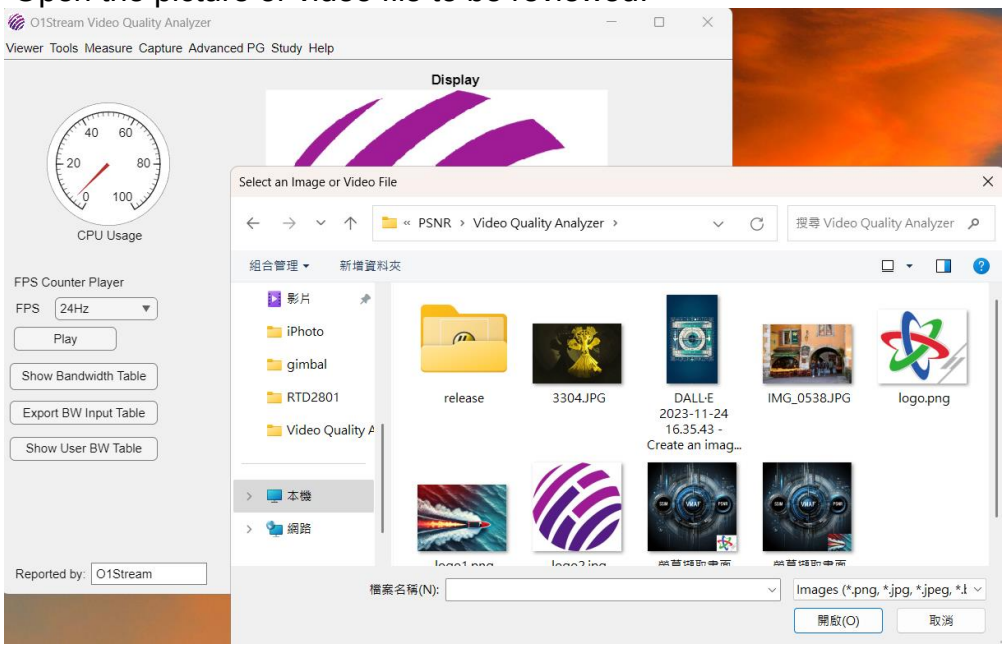
VQA is licensed via a USB hardware key. The application verifies the key at startup and exits gracefully if the key is missing. The current version of this manual is Rev. 1.1; the matching application version is shown in Help > About.

2 Viewer



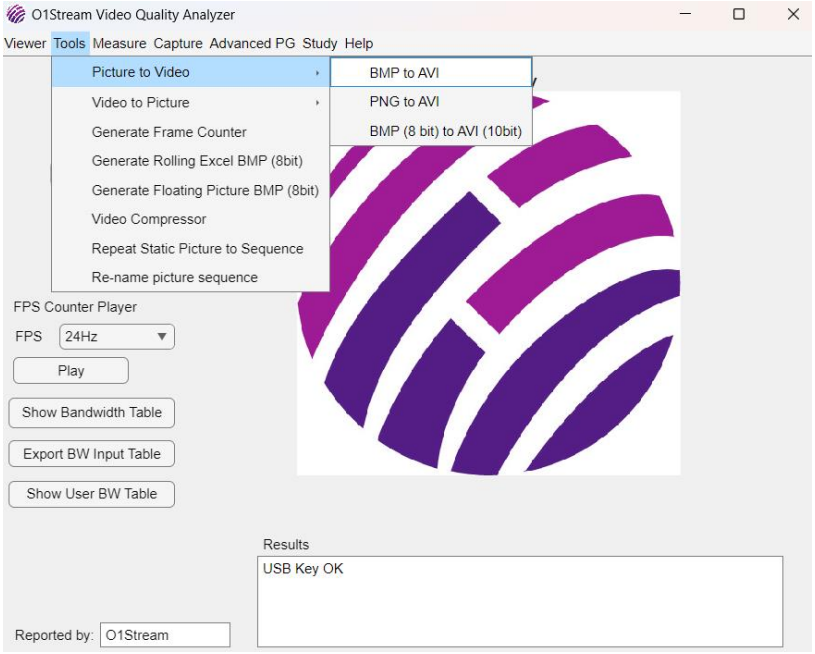
2.1 Open

Open the picture or video file to be reviewed.



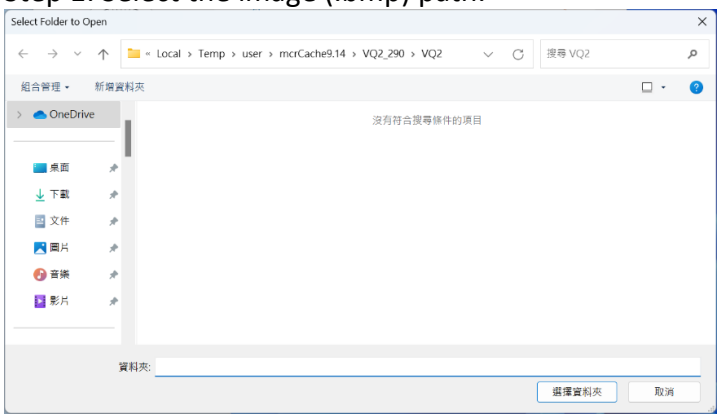
3 Tools

3.1 Picture to Video

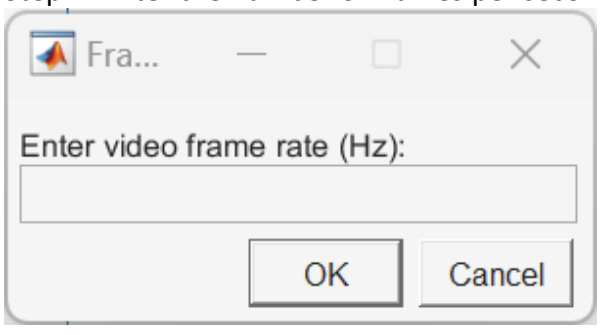


3.1.1BMP to AVI

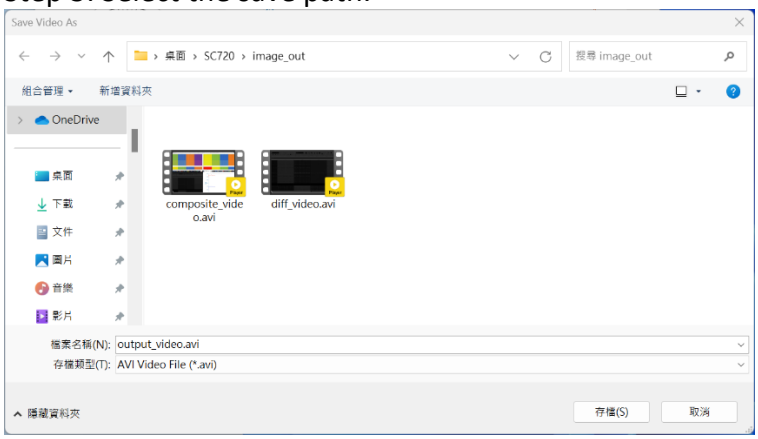
Step 1: Select the image (.bmp) path.



Step 2: Enter the number of frames per second for creating the video.

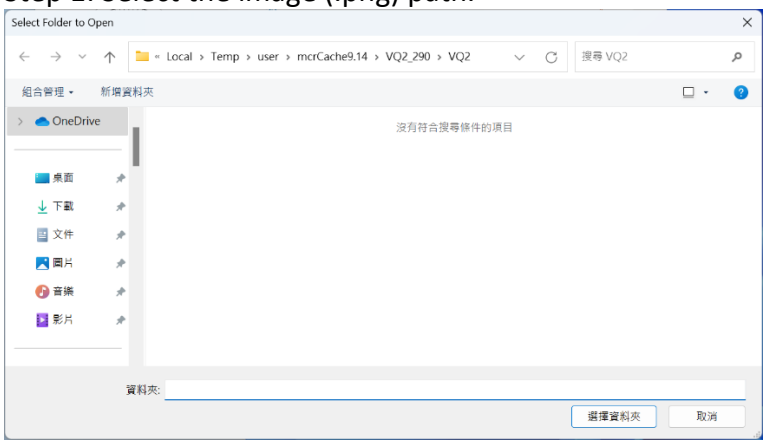


Step 3: Select the save path.

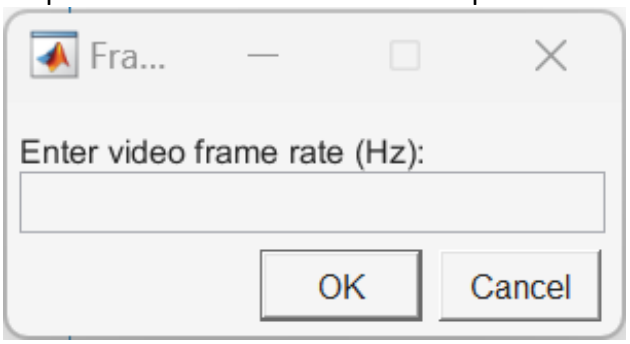


3.1.2PNG to AVI

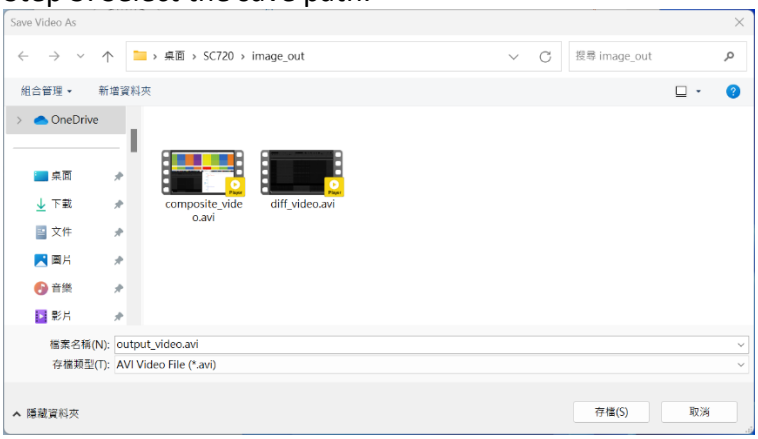
Step 1: Select the image (.png) path.



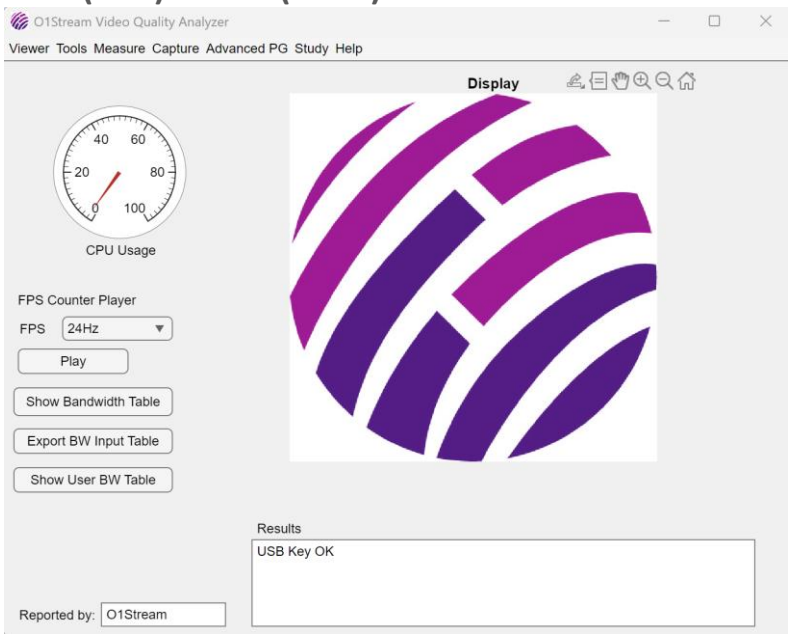
Step 2: Enter the number of frames per second for creating the video.



Step 3: Select the save path.

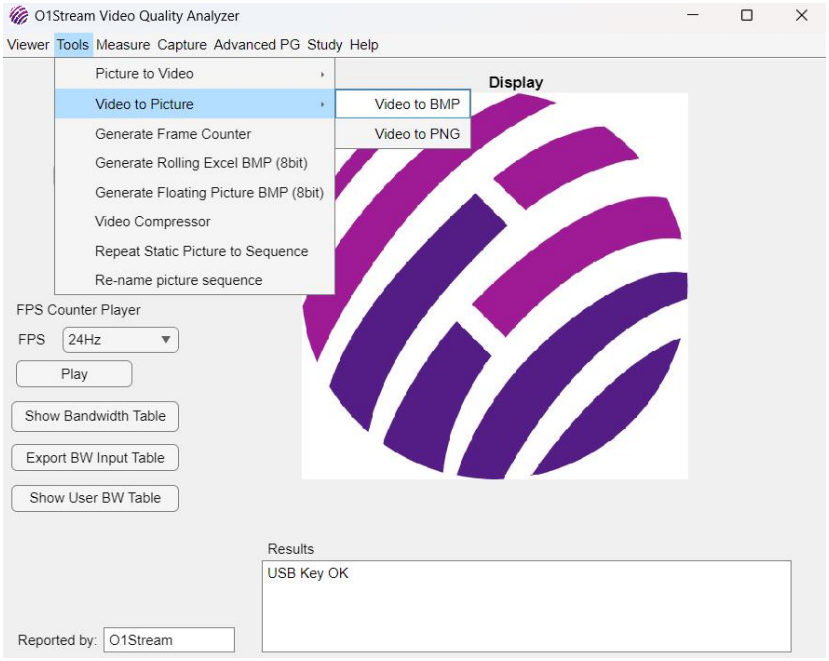


3.1.3BMP (8bit) to AVI (10bit)



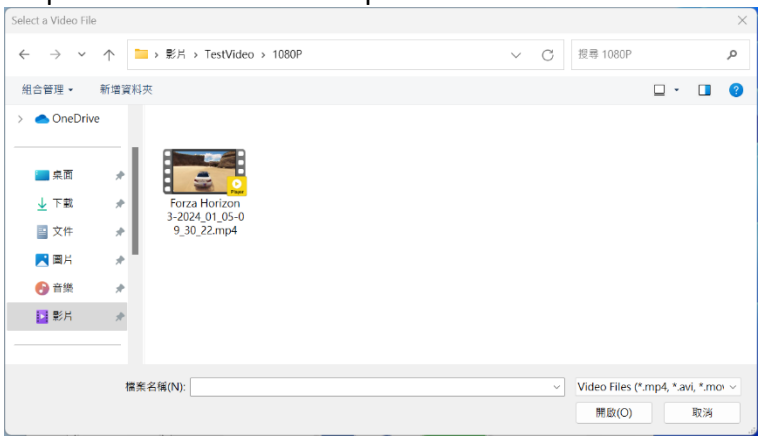
The function in this app is designed to streamline the process of converting BMP images into a 10-bit AVI video file. It guides users through a series of interactive steps, starting with folder selection for BMP files, followed by setting the desired frame rate and specifying the output video file name. The function then processes each BMP image by scaling its bit depth from 8-bit to 10-bit and saving the resulting images as intermediate PNG files in a designated subfolder. Utilizing FFmpeg, the function compiles these 10-bit PNG frames into a high-quality AVI video in the YUV444 format. Throughout the process, the app provides real-time updates and handles errors gracefully.

3.2 Video to Picture

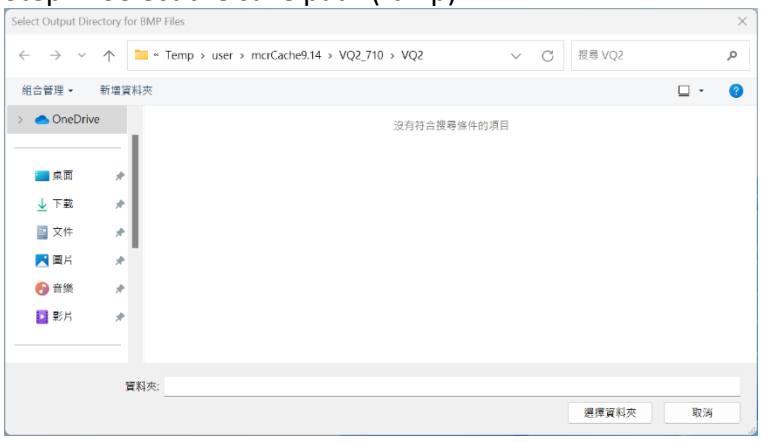


3.2.1AVI to BMP

Step 1: Select the video file path.

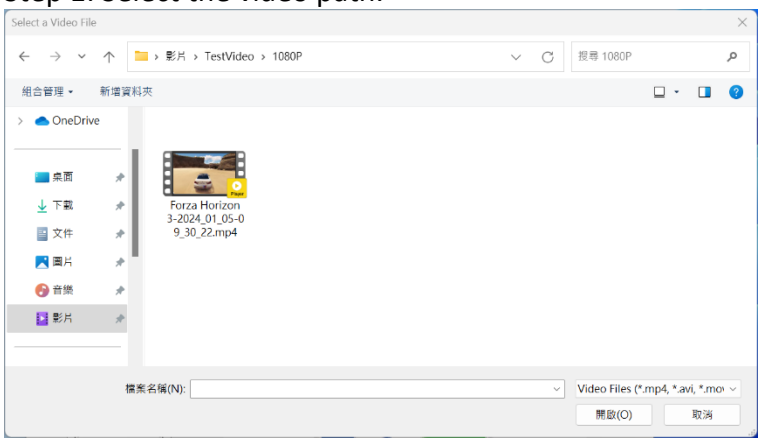


Step 2: Select the save path (.bmp).

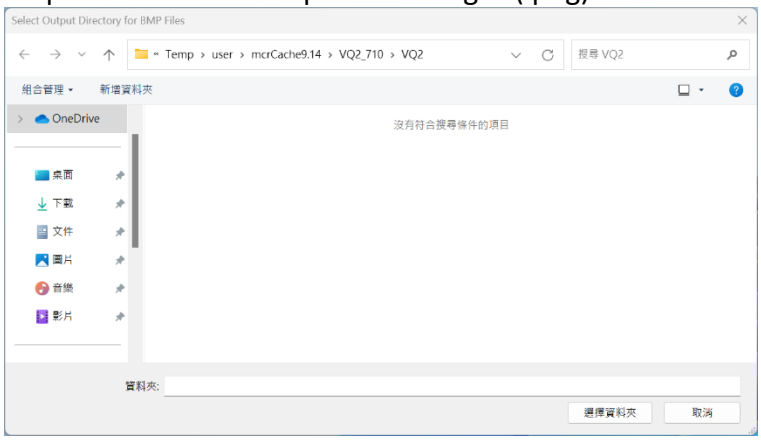


3.2.2AVI to PNG

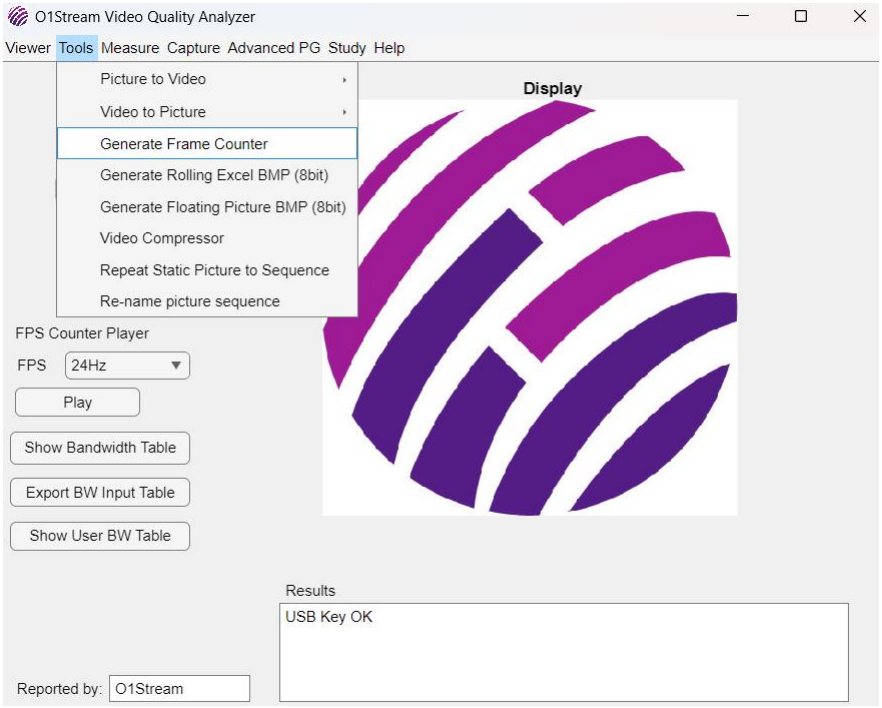
Step 1: Select the video path.



Step 2: Select the save path for images (.png).



3.3 Generate Frame Counter

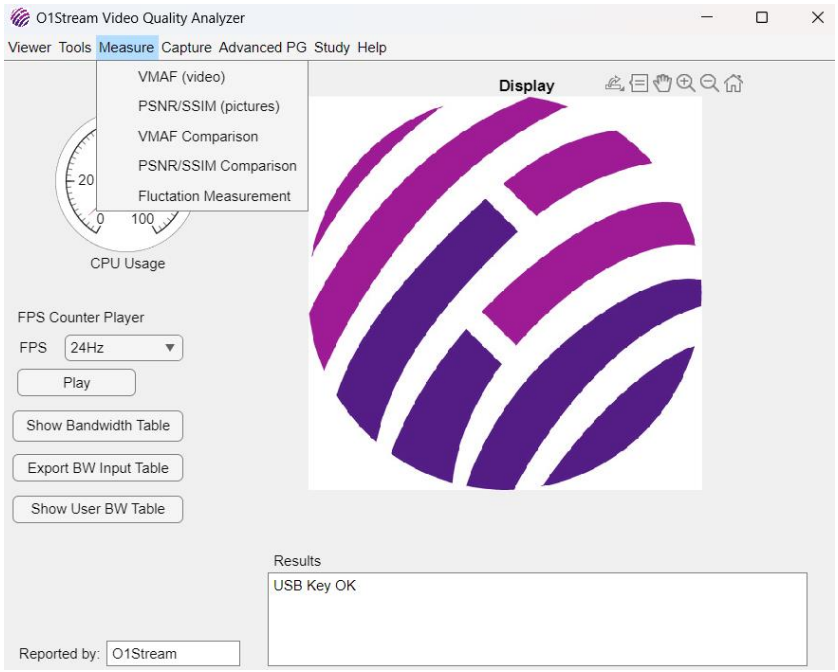


This function is designed to produce a visually dynamic **frame counter video** for testing purposes. It begins by prompting the user for critical parameters, such as the desired frame rate, video duration, and output file location and name. Once these details are provided, the function dynamically generates video frames featuring a prominently displayed frame counter timer in the center of the screen.

The video also incorporates user-defined text overlays, including a primary string entered by the user and a secondary string that adapts dynamically based on the content of the first. Additionally, animated elements such as bouncing squares and overlapping triangles move across the screen, creating a visually engaging output. To enhance visual appeal, the background color changes at regular frame intervals, adding another layer of dynamism to the video. Using MATLAB's VideoWriter class, the function compiles these frames into a high-quality MPEG-4 video file while validating user inputs, providing real-time status updates, and handling errors gracefully.

The primary purpose of this function is to test USB capture devices by generating a precise frame counter video to measure latency. The prominently displayed frame counter allows users to easily identify the number of frames delayed between the input and output of the capture device. The dynamic visual elements, such as moving shapes and changing backgrounds, further test the capture device's ability to handle complex visuals and maintain frame synchronization. With its customizable frame rate and duration, the function is adaptable to various testing scenarios. The resulting video provides a reliable and repeatable tool for evaluating the real-time performance of USB capture devices under different conditions.

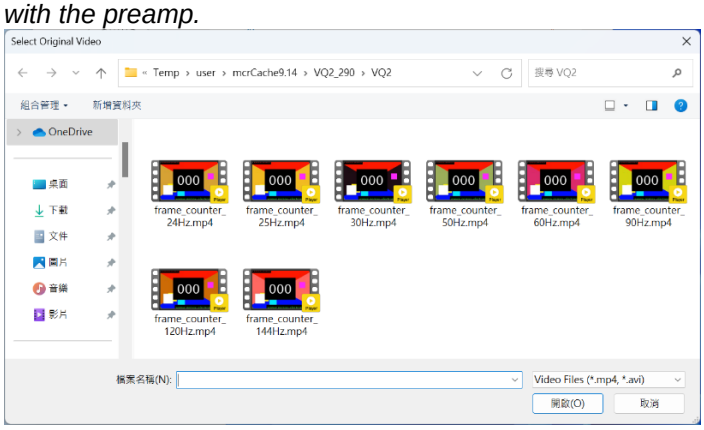
4 Measure



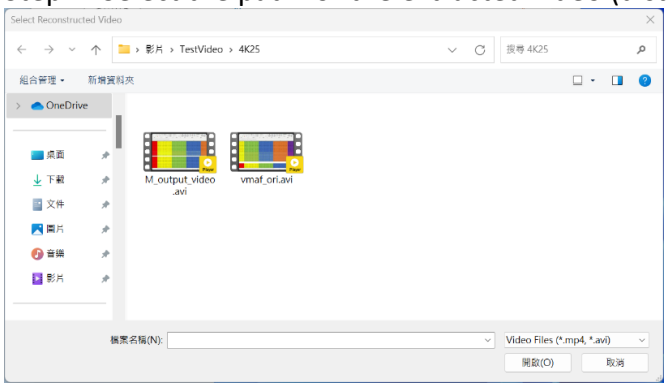
4.1 VMAF (video)

Step 1: Select the original video path.

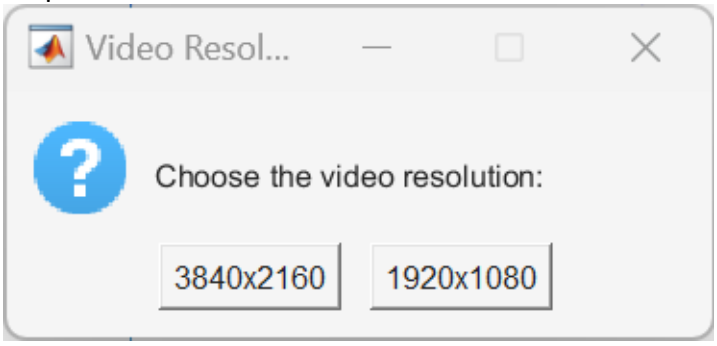
**Note: To generate a video with the correct frame count displayed in the bottom right corner, use images created with the preamp.*



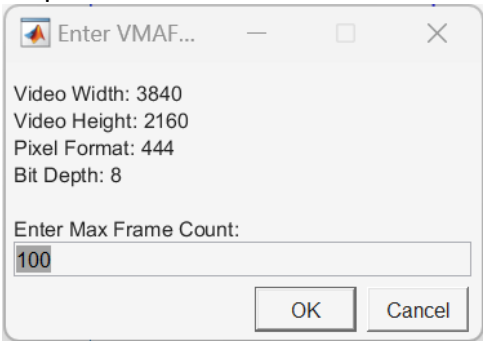
Step 2: Select the path for the extracted video (disturb).



Step 3: Select the video resolution.



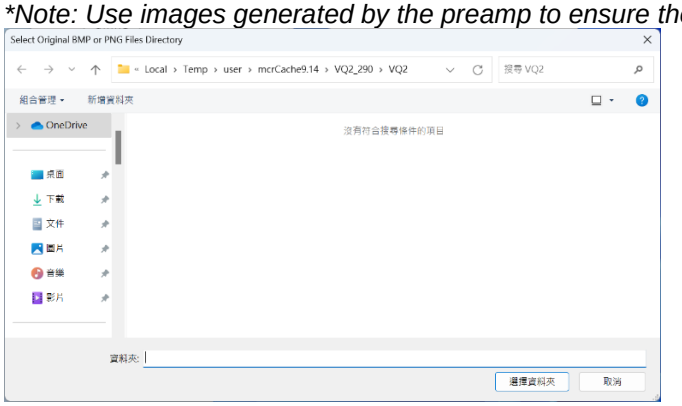
Step 4: Enter the max number of frames for the video to be analyzed.



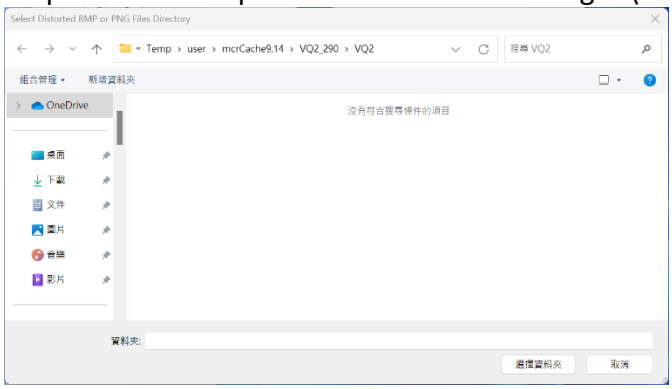
4.2 PSNR / SSIM (pictures)

Step 1: Select the original image path.

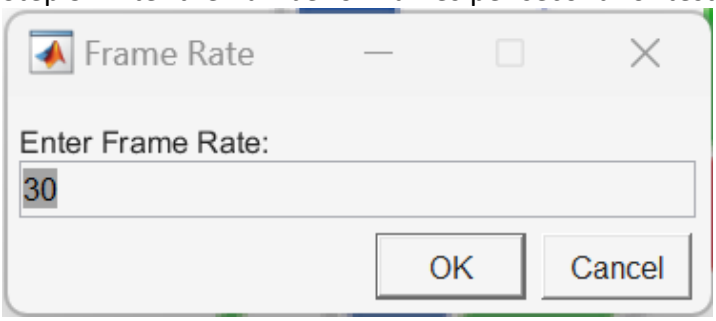
**Note: Use images generated by the preamp to ensure the frame count is displayed in the bottom right corner.*



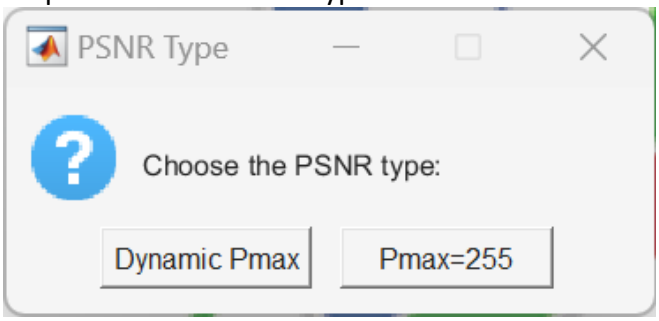
Step 2: Select the path for the extracted images (disturb).



Step 3: Enter the number of frames per second for testing.

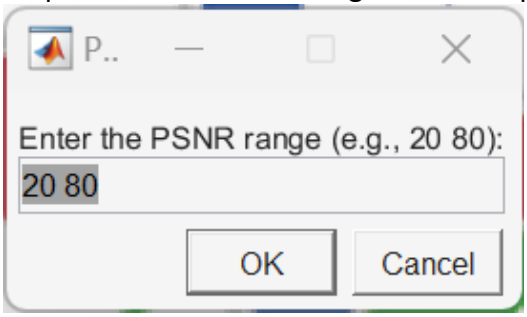


Step 4: Select the PSNR type.



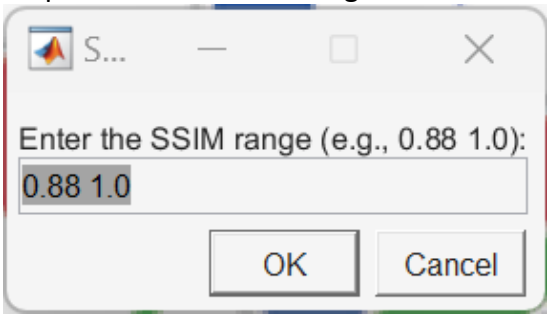
**Note: Select Pmax = 255 for most cases.*

Step 5: Enter the PSNR range to use for plotting the PSNR figure.



**Note: The test results may be higher or lower than the set value.*

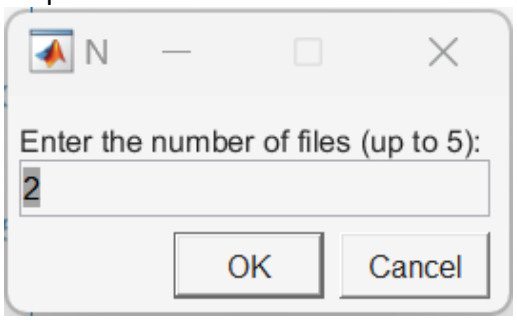
Step 6: Enter the SSIM range.



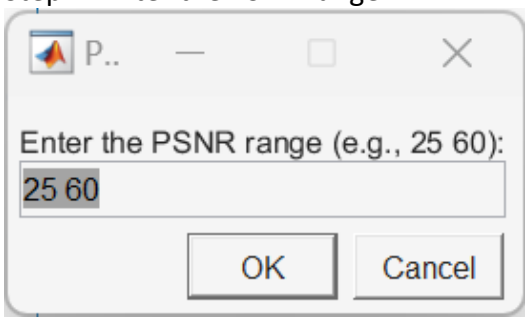
*Note: The test results may exceed or fall below the set value.

4.3 VMAF Comparison

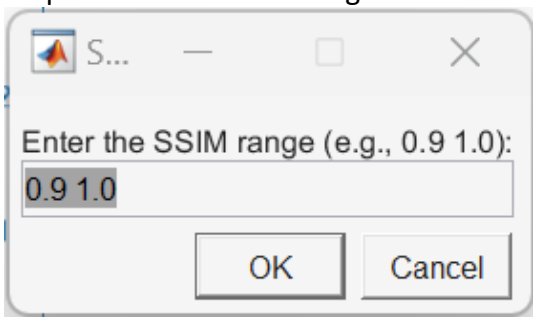
Step 1: Select the number of files to compare.



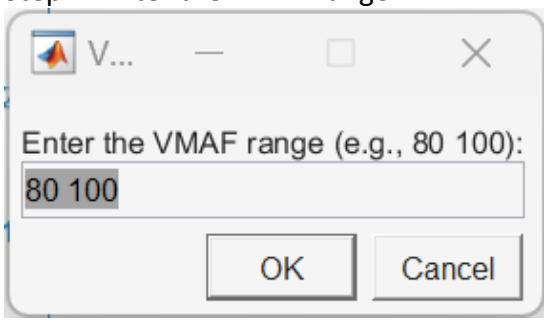
Step 2: Enter the PSNR range.



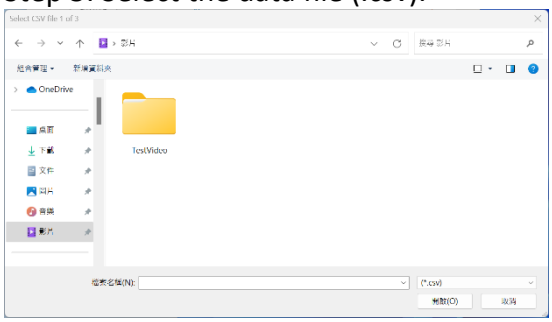
Step 3: Enter the SSIM range.



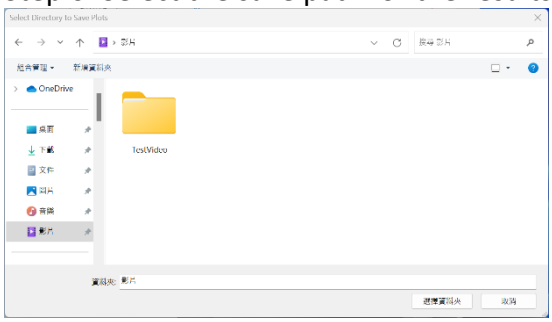
Step 4: Enter the VMAF range.



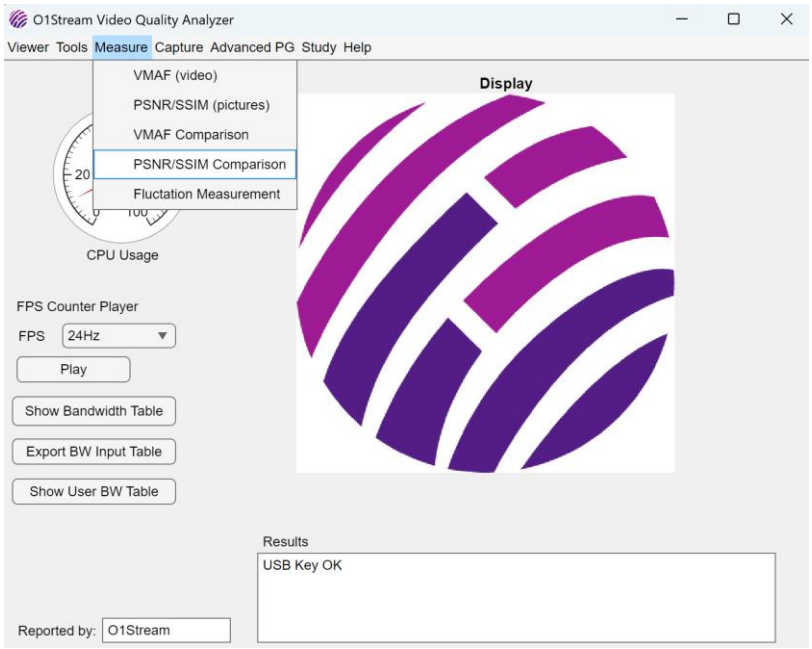
Step 5: Select the data file (.csv).



Step 6: Select the save path for the results.



4.4 PSNR / SSIM Comparison



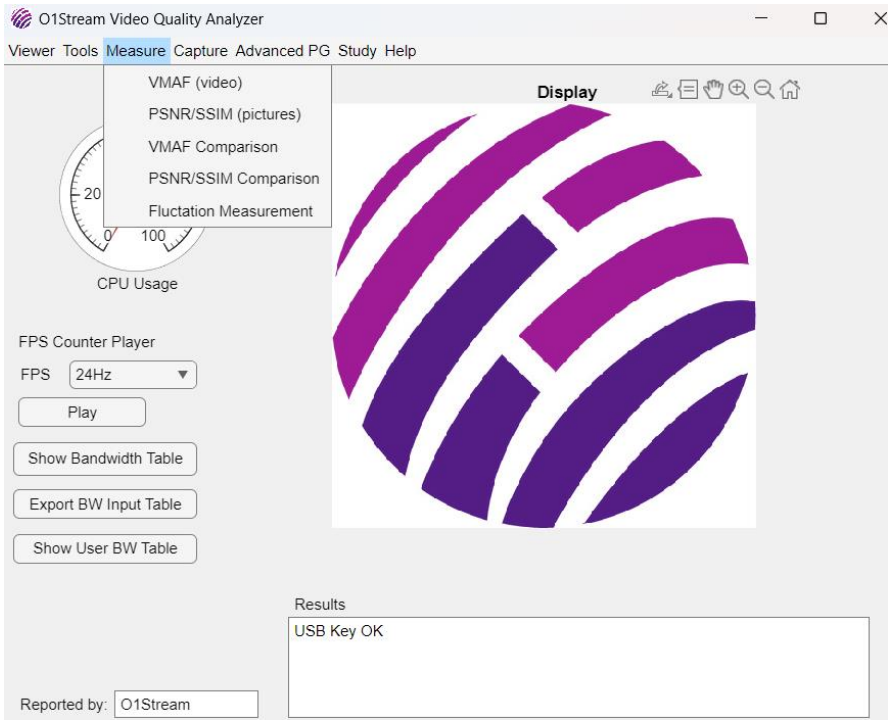
The feature is designed to facilitate the comparison of PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index) metrics across multiple datasets stored in Excel files. It offers a structured, user-friendly process that includes file selection, metric extraction, visualization, and automated report generation. The detailed workflow ensures precise and customizable analysis for video quality comparisons.

Detailed Process:

- **Input Collection:**
  - Prompts the user to specify the number of Excel files to compare (up to 5).
  - Asks for custom PSNR and SSIM ranges, allowing tailored visualization to fit the data's scale or analysis needs.
- **File Selection:**
  - Guides the user through selecting individual Excel files containing PSNR and SSIM metrics.
  - Validates file selection to ensure all required files are provided before proceeding.
- **Save Location:**
  - Prompts the user to select a directory for saving output plots and the final PDF report.
  - Defaults to the last selected file's directory if no new location is specified.
- **Data Extraction:**
  - Reads PSNR and SSIM data from the selected Excel files.
  - Stores the data in structured arrays for plotting and analysis.
- **Visualization and Plotting:**
  - Generates individual comparative plots for PSNR and SSIM metrics across all files.
  - Saves these plots as high-quality PNG images for future reference.
- **Report Generation:**
  - Combines the plots into a single PDF report annotated with user-provided details such as an author name.
  - Includes clear labels and comparisons to ensure the report is professional and easy to interpret.
- **Final Output:**
  - A well-organized report containing PSNR and SSIM comparison plots, saved as a PDF in the user-specified directory.
  - Clear feedback in the app confirming the process completion and output location.

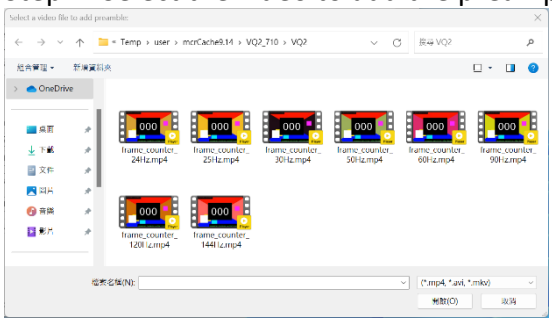
By automating the comparison process and providing customizable options, this function ensures efficient and accurate video quality analysis, making it an essential tool for evaluating and comparing video datasets in a structured manner.

5 Capture

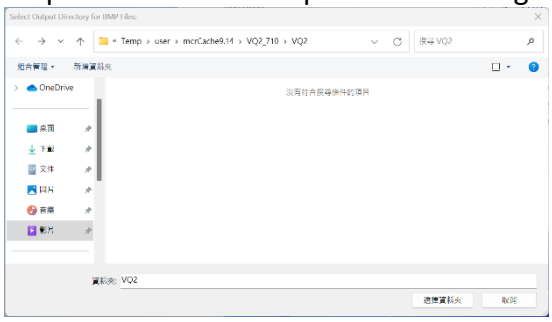


5.1 Preamble

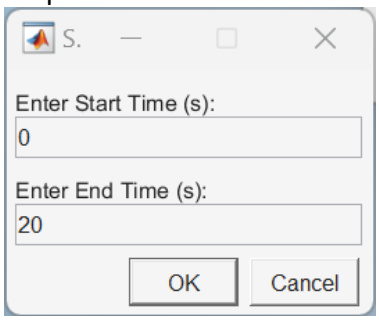
Step 1: Select the video to add the preamp.



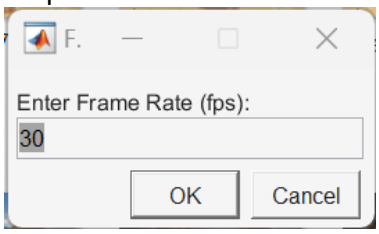
Step 2: Select the save path for the images after adding the preamp.



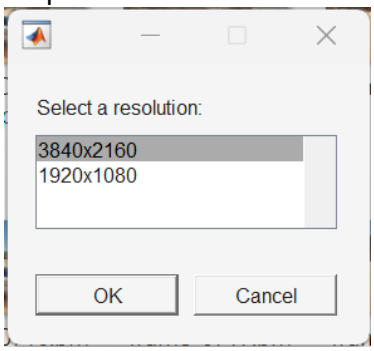
Step 3: Set the start and end times of the original video.



Step 4: Set the number of frames per second for the video.

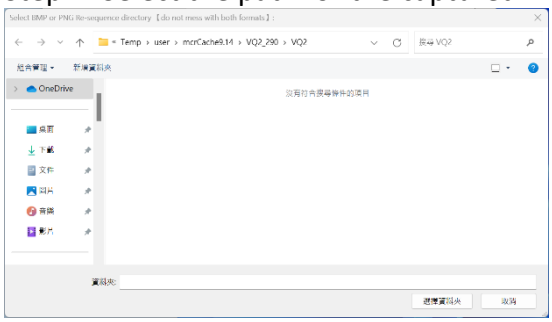


Step 5: Set the video resolution.

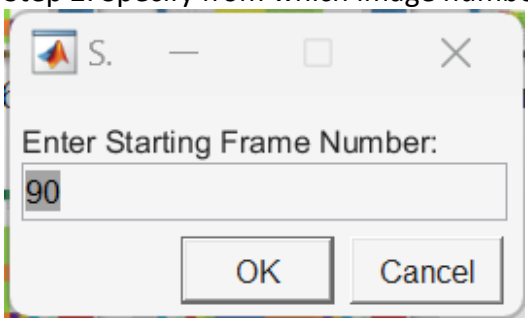


5.2 Re-Seq

Step 1: Select the path for the captured images.

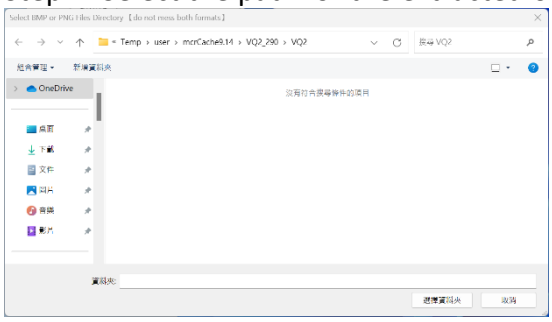


Step 2: Specify from which image number the extracted or captured images should be retained.

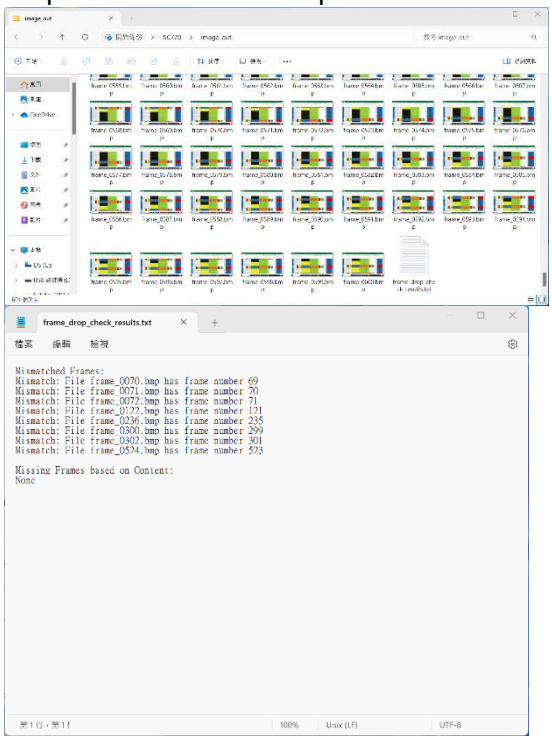


5.3 Frame drop check

Step 1: Select the path for the extracted or captured images.



Step 2: Generate the inspection results.



note:

- **Mismatched Frames**: Indicates that the file name of the image does not match the frame number within the image.
- **Missing Frames Based on Content**: Indicates frame numbers that cannot be found among all the images.

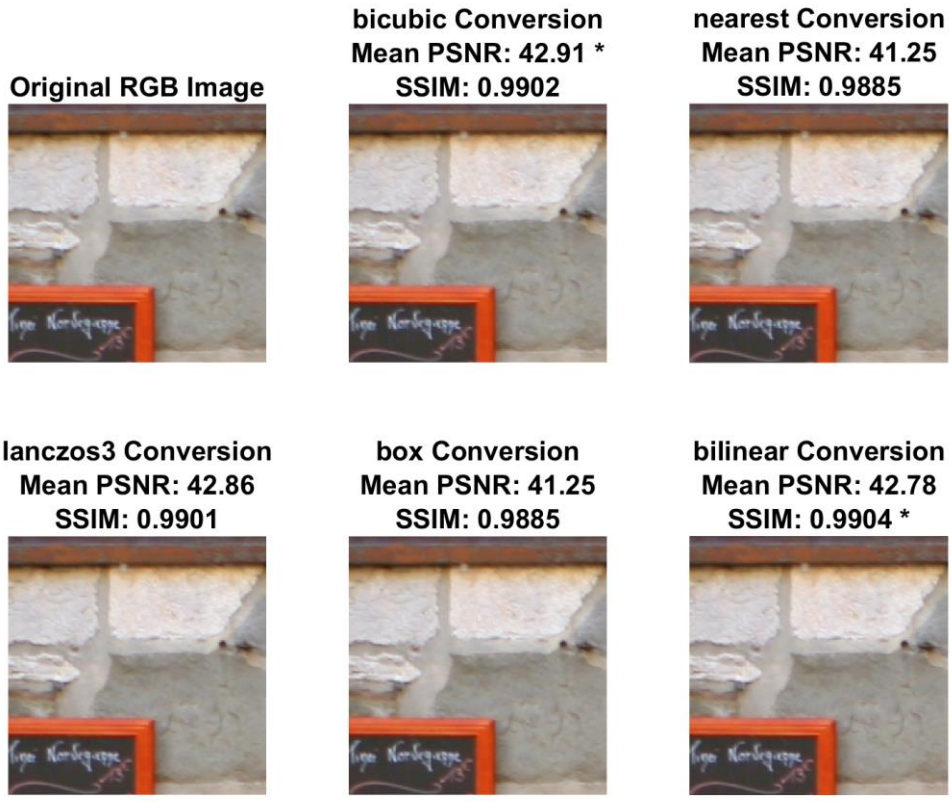
6 Study

In this section, we present several examples to illustrate the underlying meaning of PSNR and SSIM, two commonly used metrics for assessing video quality. These measurements are widely recognized and generally provide a reliable indication of quality differences between original and compressed videos or images. For most scenarios, they serve as useful benchmarks for evaluating video fidelity. However, it is important to acknowledge that PSNR and SSIM do not always capture the nuances of perceived video quality and can occasionally produce misleading results, particularly in cases where visual perception differs from mathematical accuracy.

To address these limitations, we encourage users to explore these intuitive examples, which demonstrate scenarios where PSNR and SSIM may not align with human visual judgment. By doing so, users can develop a better understanding of these metrics' strengths and weaknesses. For a more comprehensive evaluation of video quality, it is recommended to use multiple measurement methods in combination, incorporating subjective assessments or other advanced metrics to ensure a well-rounded analysis.

6.1 Simple RGB to NV12 Conversion

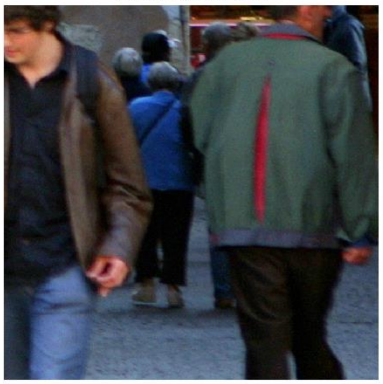
Example of RGB to NV12 conversion shows that the best PSNR approach doesn't always yield the best SSIM, highlighting differences in picture quality.



6.2 PSNR paradox

Example of images with comparable PSNR values but distinctly different in visual perception.

Noisy Original
PSNR: 34.36



Blurred Image
PSNR: 34.31

Noisy Original
PSNR: 39.02

Blurred Image
PSNR: 39.10

Noisy Original
PSNR: 32.33

Blurred Image
PSNR: 32.42

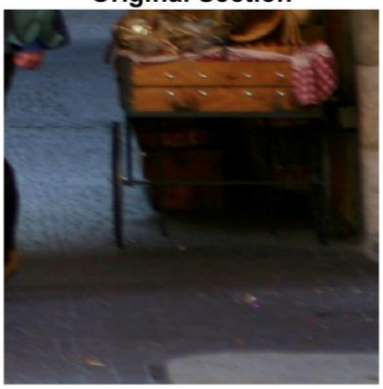
6.3 SSIM paradox

Example of images with comparable SSIM values but distinctly different in visual perception.

Original Section

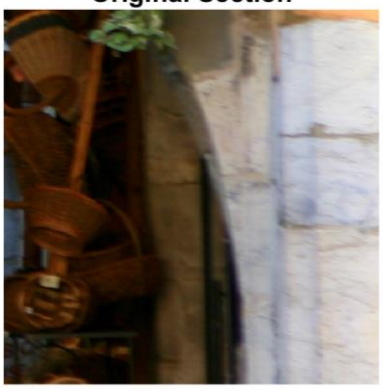
Contrast Adjusted
SSIM: 0.9347

Original Section

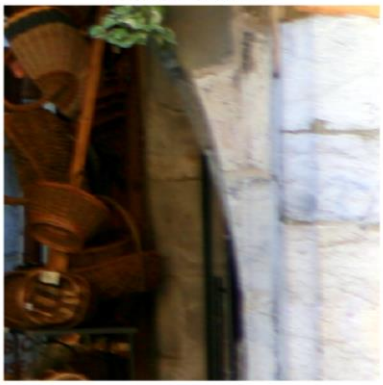


Contrast Adjusted
SSIM: 0.6509

Original Section



Contrast Adjusted
SSIM: 0.9791

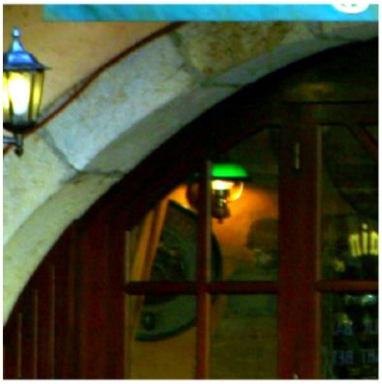


Original Section

Contrast Adjusted
SSIM: 0.8221

Original Section

Contrast Adjusted
SSIM: 0.6169

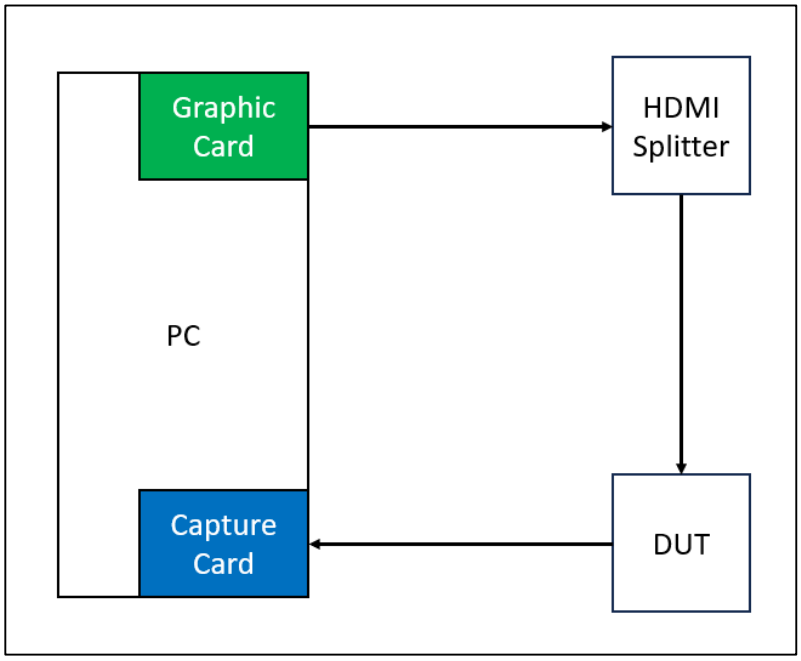


7 Help

The help section includes several PDF files, such as the video evaluation flow, O1S-VQA SOP, video metrics analysis guide, and the advanced PG guide. These resources provide valuable insights into how to effectively use the O1S-VQA and explore the key concepts behind this powerful toolkit.

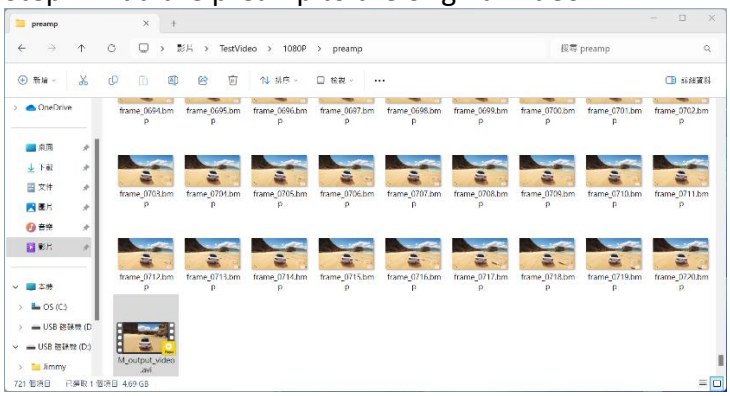
8 Video Capture Setting Example

A 1920x1080P60 video is available and processed through the device under test (DUT). Frames are extracted and compared with the original video to evaluate PSNR, SSIM, and VMAF values. The testing framework is as follows:



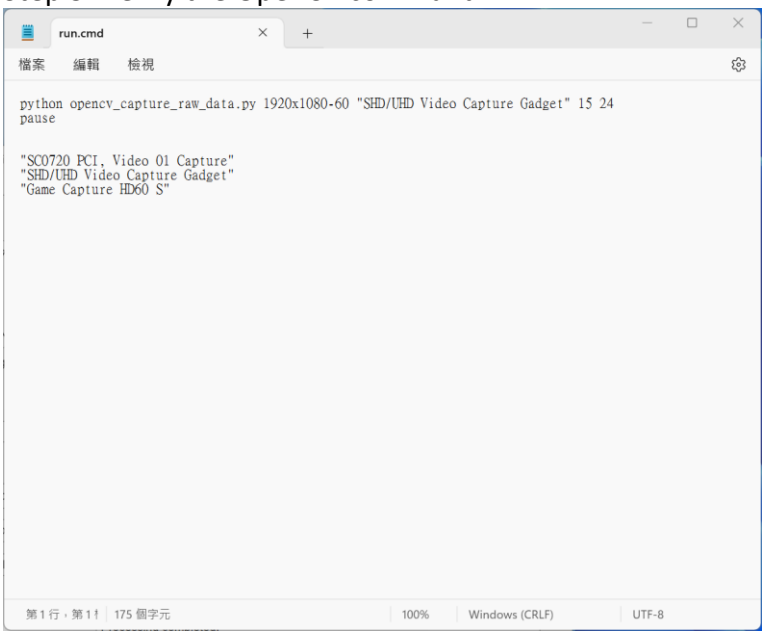
In our example setup, we use a 4K60-capable PCIe capture card and utilize Python with OpenCV to capture images from the PCIe card.

Step 1: Add the preamp to the original video.



Step 2: Verify that the resolution of the graphics card output matches the video resolution.

Step 3: Verify the OpenCV command.



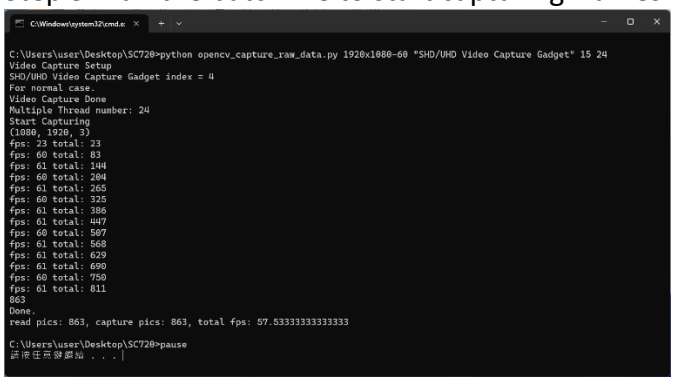
**Note: Explanation of the OpenCV command is as follows:*

- **Resolution-Frame Rate**: 1920x1080-60
- **Capture Card Name**: "SHD/UHD Video Capture Gadget"
- **Capture Duration**: 15 seconds
- **Number of Threads**: 24

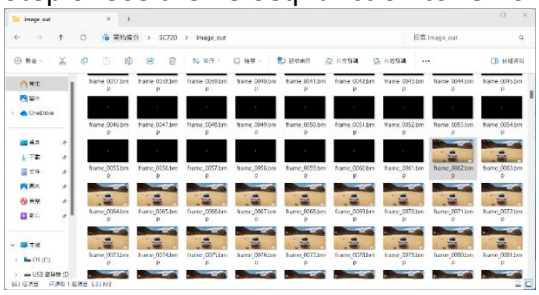
Step 4: Use VLC Player to play the preamp video and switch to full-screen mode.

**Note: Other media players may experience frame drops; it is recommended to use VLC Player.*

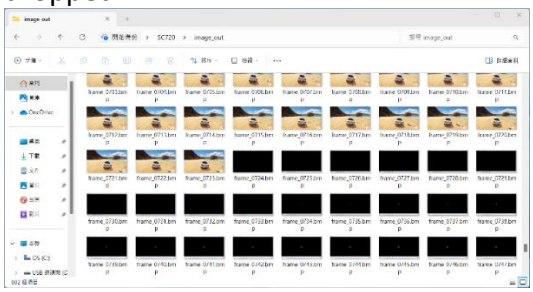
Step 5: Run the batch file to start capturing frames and wait for the process to complete.



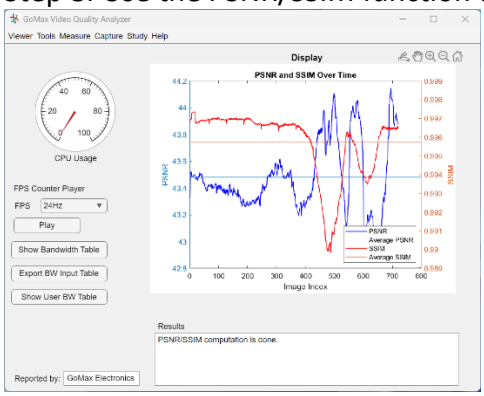
Step 6: Use the Re-Seq function to remove the preamp frames.



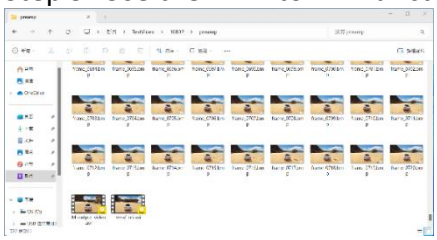
Step 7: Delete the extra images and use the Frame Drop Check function to ensure no frames were dropped.



Step 8: Use the PSNR/SSIM function to evaluate the quality of the frames.

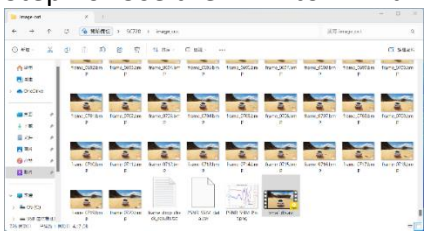


Step 9: Use the BMP to AVI function to generate the VMAF_ori video.



Note: The **VMAF_ori video is created from images generated by the preamp (with frame numbers displayed in the bottom right corner).*

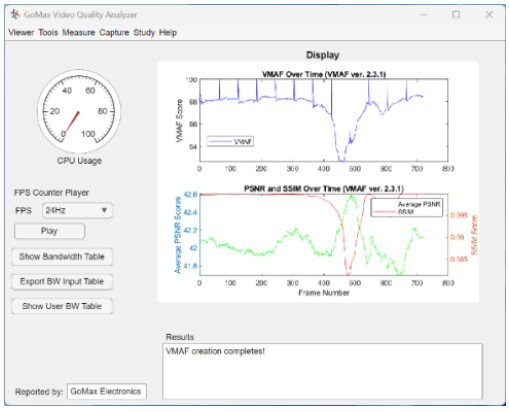
Step 10: Use the BMP to AVI function to generate the VMAF_dis video.



Note:

1. If the captured images are in PNG format, use the **PNG to AVI** function.
2. Ensure there are no extra PNG files in the directory (e.g., PSNR_SSIM_Plot.png) when processing the PNG images.

Step 11: Use the VMAF function to evaluate video quality.



Notes:

- Ensure the graphics card and PCIe capture card do not drop frames. It is especially possible that a not-qualified PCIe capture card will constantly and frequently loses pictures.
- Use an qualified HDMI 2.0 Splitter with a fixed EDID to ensure consistent output from the graphics card across all devices under test.
- Confirm that all testing paths maintain the same resolution, frame rate, color space, and color depth.

9 Concepts and Metrics

This chapter explains the measurement theory behind the analyzer. The tool implements full-reference video quality assessment, meaning every metric compares an extracted (distorted) frame against the corresponding original (reference) frame. Without an aligned reference there is no measurement to make; correct synchronization is therefore the single most important prerequisite for trustworthy results.

9.1 Reference and Distorted Streams

Throughout this manual the original video is referred to as the reference, and the version that has passed through the device under test (DUT) is referred to as the distorted version. The Preamble function in chapter 5.1 makes per-frame alignment between the two reliable: it inserts three sync frames at the head of the reference and stamps every subsequent frame with a four-digit number in the bottom-right corner. The capture pipeline then OCR-reads that number to detect frame drops and to align reference and distorted sequences on a per-frame basis.

9.2 PSNR (Peak Signal-to-Noise Ratio)

PSNR is computed from the mean squared error (MSE) between corresponding pixels of the reference and distorted images, expressed in decibels relative to a peak signal level. The formula used is:

$$PSNR = 20 \cdot \log_{10}(MAX_I / \sqrt{MSE})$$

where MAX_I is the maximum possible pixel value (255 for 8-bit data, 1023 for 10-bit) and MSE is the mean squared error between the original and the distorted image. Higher PSNR means a closer match.

9.2.1Strengths

PSNR is fast, exact, and trivially reproducible. It does not depend on training data or on the parameters of any model, which makes it the natural metric for regression tests and for cross-checking other metrics.

9.2.2Weaknesses

PSNR treats every pixel independently and weights all errors equally. It does not account for the spatial and temporal masking of the human visual system, so it correlates only moderately with perceived quality. PSNR also fluctuates strongly with scene content: complex frames produce lower PSNR even when the codec is functioning correctly, while easily compressible content produces high PSNR that overstates the quality of weak codecs.

9.2.3Reading PSNR Values

As a rough rule of thumb, PSNR above 40 dB usually corresponds to a visually transparent reproduction, 30 – 40 dB to a noticeable but acceptable degradation, and below 25 dB to clearly degraded video. Do not interpret short dips as catastrophic failures without cross-checking SSIM or VMAF, especially when the dips coincide with scene cuts or with GOP boundaries in compressed streams.

The figure below shows what truncating a 10-bit luminance gradient down to fewer bits does to the visible image and to PSNR. Banding becomes obvious below ~6 bits; the PSNR floor is approximately 6 dB per lost bit. This is also what the Study › Pixel Bit Depth menu produces.

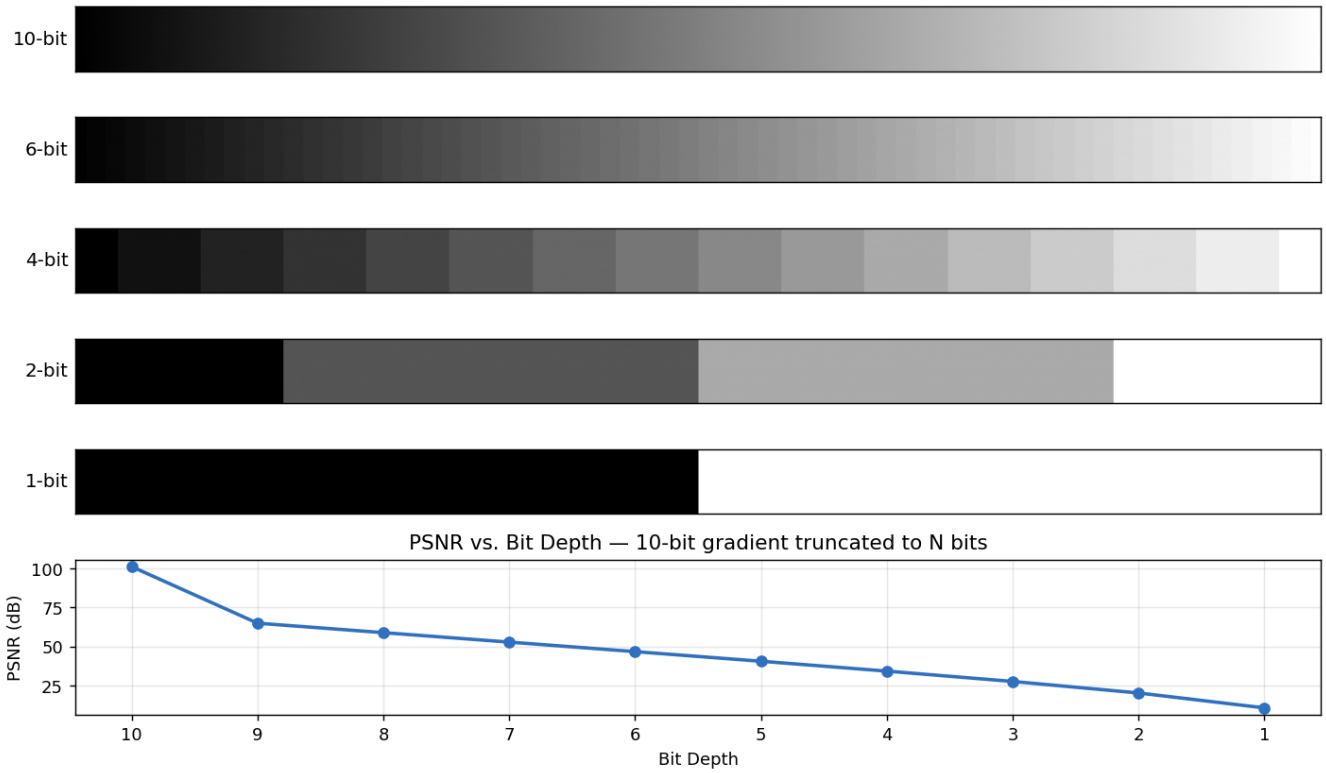


Figure 9.1 — PSNR vs. bit depth (10-bit gradient truncated to N bits).

9.3 SSIM (Structural Similarity Index)

SSIM compares local windows of luminance, contrast, and structure between reference and distorted images. The published formula is:

$$SSIM(x, y) = [(2\cdot \mu_x \cdot \mu_y + C1) \cdot (2\cdot \sigma_{xy} + C2)] / [(\mu_x^2 + \mu_y^2 + C1) \cdot (\sigma_x^2 + \sigma_y^2 + C2)]$$

where μ_x and μ_y are the local means of the two windows, σ_x^2 and σ_y^2 are their variances, σ_{xy} is the cross-covariance, and C1 and C2 are small stabilizing constants. The score ranges from 0 to 1, where 1 is identical.

9.3.1Strengths

SSIM is much closer to perceived quality than PSNR for typical compression artifacts (blocking, blurring, ringing, and quantization noise). Because it compares local structure, it tends to match human judgements about sharpness and detail preservation.

9.3.2Weaknesses

SSIM is insensitive to several global distortions that humans notice strongly. Most notably, hue rotations and uniform colour shifts have little effect on SSIM: a frame can be visibly recoloured and still score 0.93. SSIM also reacts to geometric distortions like rotations and stretches that may or may not be visually objectionable depending on context.

SSIM is therefore best read alongside PSNR and $\triangle$E2000 rather than in isolation.

9.4 VMAF (Video Multimethod Assessment Fusion)

VMAF is a learned full-reference metric originally trained by Netflix on subjective ratings collected from human viewers. It fuses several elementary features into a single 0 – 100 score, where higher is better.

9.4.1Feature Extraction

The VMAF model extracts the following features from each pair of reference and distorted frames: Detail Loss Metric (DLM), Motion^2, Visual Information Fidelity (VIF), and an additive distortions metric.

9.4.2Feature Integration

These features are then combined using a Support Vector Machine trained on a dataset rated by human viewers. The model outputs a quality score per frame that is averaged across the clip to produce the headline VMAF figure.

9.4.3How the Analyzer Computes VMAF

VMAF is computed only on video files. The analyzer converts both reference and distorted clips to YUV (4:4:4, 8-bit by default) using FFmpeg, then invokes vmaf.exe with both PSNR and float-SSIM features enabled. Per-frame results are written to vmaf_output.csv next to the distorted file, and a summary plot is rendered in the main display area.

9.4.4When to Use VMAF

Because VMAF accounts for some temporal context, it is the recommended single-number quality figure when compressed codecs are involved. It is particularly useful for comparing competing encoders and bitrate ladders. VMAF is not infallible — its accuracy depends on whether the content type and distortion type were represented in its training set — so cross-checking with PSNR and SSIM remains good practice.

9.5 △E2000 (Colour Difference)

PSNR and SSIM operate in the RGB or luminance domain and do not explicitly characterize colour shift. The Color Vector Analysis (△E2000) function fills that gap.

9.5.1From CIE76 to CIEDE2000

The classic △E76 formula is the Euclidean distance between two colours in CIE Lab space:

△E76 = sqrt(△L^2 + △a^2 + △b^2)

△E76 assumes a uniform perceptual space, which is not the case: human eyes perceive colour differences non-linearly, and equal numerical changes can look very different depending on hue, chroma, and lightness. CIEDE2000 was introduced to correct for this. The figure below shows the structure of CIEDE2000 — the major addition over △E76 is the hue-rotation term R_T and the perceptual-uniformity scaling factors S_L, S_C, and S_H.

hues differently.

✅ **Lightness Weighting** – Accounts for the fact that dark colors appear to change more than light ones.

✅ **Neutral Area Sensitivity** – Improves accuracy for near-grayscale colors.

CIEDE2000 Formula (Simplified)

The ΔE2000 formula refines **ΔE76** by adding corrections based on:

- **ΔL** (Lightness Difference)
- **ΔC** (Chroma Difference)
- **ΔH** (Hue Difference)
- **G** (Correction Factor for Chroma)
- **S_L, S_C, S_H** (Scaling Factors for Perceptual Uniformity)

The general structure:

$$\Delta E_{2000} = \sqrt{\left(\frac{\Delta L'}{S_L}\right)^2 + \left(\frac{\Delta C'}{S_C}\right)^2 + \left(\frac{\Delta H'}{S_H}\right)^2 + R_T \cdot \frac{\Delta C'}{S_C} \cdot \frac{\Delta H'}{S_H}}$$

where **R_T** accounts for hue rotations.

Figure 6.1 — CIEDE2000 formula structure. Source: bundled ΔE.pdf.

The analyzer always uses CIEDE2000, never the older △E76.

9.5.2Interpreting △E2000 Values

Because CIEDE2000 is normalized to perceptual difference, a small set of thresholds applies across all colours. The pairs below illustrate the perceptual jump at each threshold:

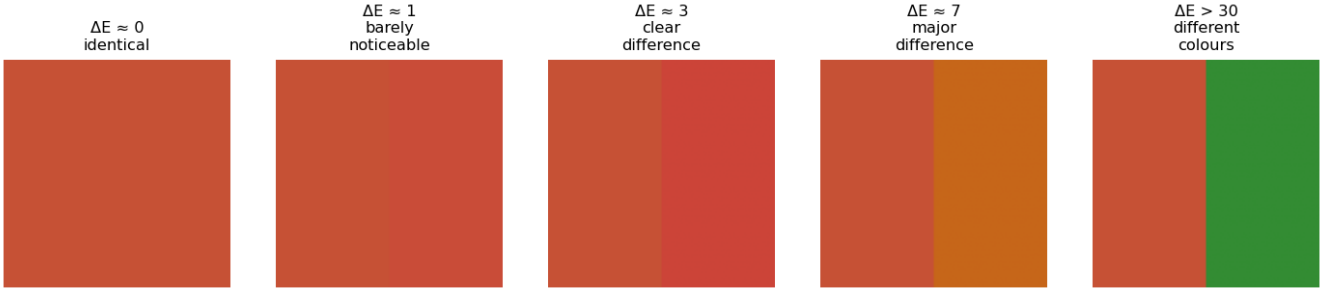


Figure 6.2 — △E2000 perceptual scale. Each pair shows two adjacent colour swatches at the indicated △E2000 distance.

△E2000 0 to 1: barely noticeable. The two colours look identical to most observers.
△E2000 1 to 2: slight difference. Visible only on careful comparison.
△E2000 2 to 3: noticeable. Visible side by side.
△E2000 3 to 5: clear difference. Visible without side-by-side comparison.
△E2000 5 to 10: major difference. The colours are obviously different.
△E2000 above 10: the colours should be considered different colours.

9.6 Choosing the Right Metric

Use PSNR for fast end-to-end smoke tests and for regression detection in known-good pipelines.
Use SSIM whenever a codec or scaler is in the path and you care about perceived sharpness or texture preservation.
Use VMAF when comparing competing encoders or bitrate ladders and you want a single bottom-line score.
Use △E2000 whenever the test exercises colour processing, including chroma subsampling, gamma adjustment, or wide-gamut to standard-gamut conversion.
In practice we recommend running PSNR/SSIM and △E2000 on every capture, and adding VMAF only when the path includes a video encoder.

10 Why Multiple Metrics Matter

PSNR alone can be misleading when used to evaluate video codecs that compress each frame independently. The fluctuations of PSNR across frames in such codecs do not necessarily reflect a decline in perceived quality. Two short studies bundled with the analyzer illustrate this point.

10.1 Frame-by-Frame Codecs vs. Temporal Codecs

Picture codecs such as JPEG and MJPEG compress each frame independently, with no reference to surrounding frames. PSNR measured on such streams therefore tracks the per-frame complexity of the source: easy frames score high, busy frames score low. The result is a PSNR curve that fluctuates much more frequently than the GOP frequency of a typical video codec.

Modern video codecs such as AV1 use I, P, and B frames within a Group of Pictures (GOP). The first I-frame of a GOP is encoded independently and may show a small PSNR dip relative to the surrounding P and B frames; the subsequent frames recover quality by referring back to that I-frame. These dips are usually imperceptible to viewers because of spatial and temporal masking effects in the human visual system.

10.2 A Bundled Comparison Experiment

The bundled VQ.pdf documents an experiment that compares JPEG-based codec output to AV1 codec output on the same 4K30 source at matched effective bitrates. The JPEG path was configured at up to 800 Mbps with frame-by-frame compression, and the AV1 path at up to 400 Mbps with adaptive bitrate streaming and a GOP size of 120 frames. Both clips were measured with the analyzer.

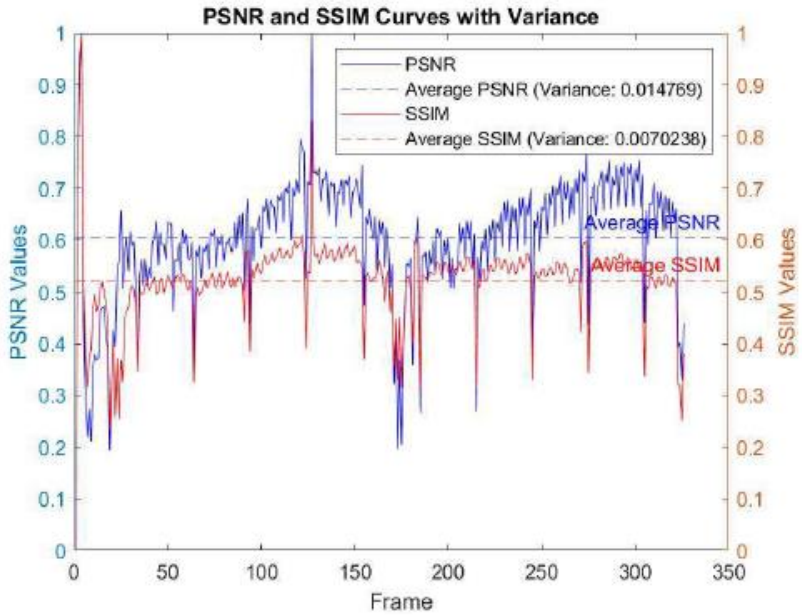
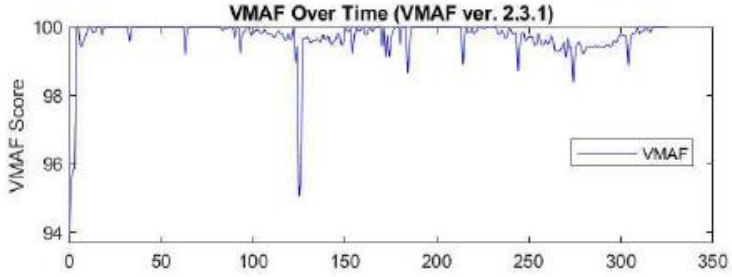


Fig 4. Normalized PSNR & SSIM Distribution of JPEG based Video Codec



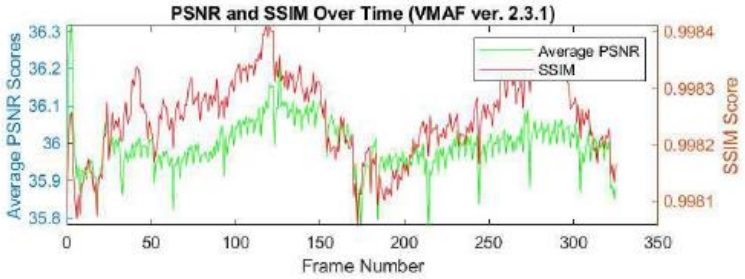


Fig 5. PSNR & SSIM & VMAF Distribution of JPEG based Video Codec

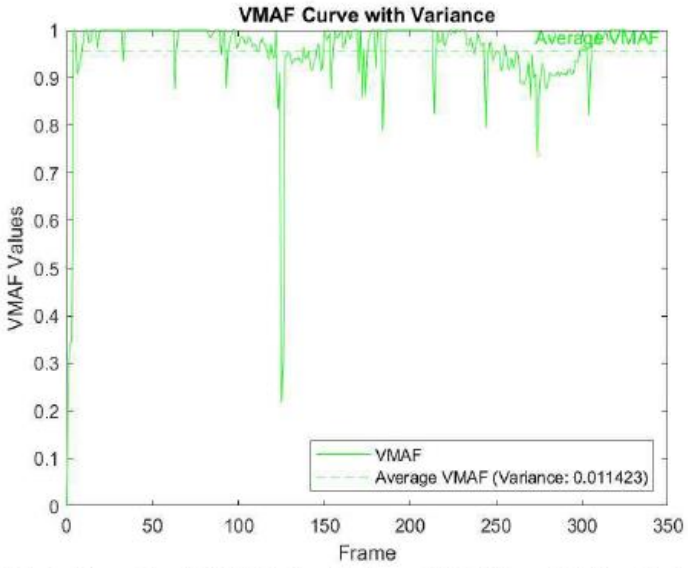


Fig 6. Normalized VMAF Distribution of JPEG based Video Codec

Figure 7.1 — PSNR / SSIM / VMAF distributions for the JPEG-based and AV1 codecs. Source: bundled VQ.pdf.

Even though AV1 routinely shows GOP boundary dips, its overall PSNR / SSIM / VMAF curves are flatter than the JPEG-based curves on the same content. The normalized fluctuation ratios reported in the study were approximately:
JPEG path: PSNR 1.47 percent, SSIM 0.70 percent, VMAF 1.14 percent.
AV1 path: PSNR 1.43 percent, SSIM 0.84 percent, VMAF 1.02 percent.
The conclusion is not that one codec is universally better than the other. The point is that comparing only peak PSNR or only one metric is insufficient. SSIM and VMAF capture spatial and temporal effects that PSNR misses, and a stable curve across all three metrics is the strongest evidence of consistent quality.

10.3 Recommended Practice

Report every measurement together with the resolution, frame rate, colour space, colour depth, and the version of the analyzer. Do not rely on a single metric. Whenever possible, run the same content through two devices or codecs and compare the curves rather than the absolute scores.

11 Measurement Process Overview

This chapter summarizes the end-to-end measurement workflow. It applies to both compressed-video and capture-device evaluations.

11.1 Step 1 — Build a Synchronized Reference

Use Capture > Preamble to add a synchronization preamble to the source video. The preamble consists of three labelled black frames (1, 2, 3) followed by every original frame stamped with a four-digit number in the bottom-right corner. The preamble guarantees that later capture passes can be aligned at frame zero, and the per-frame number lets the analyzer detect dropped or repeated frames during measurement.
Always include a few redundant sync frames at the head of the master so that a capture device that drops one or two leading frames still produces a usable measurement.

11.2 Step 2 — Choose the Test Architecture

Two architectures are typical, depicted in figures 1.1 and 1.2 of the Introduction:

11.2.1 Compressed-Video Measurement

A test video with preamble is sent to a video encoder, then through any HDMI signal path, and finally captured by a capture card running on a PC with raw-data capture. The captured frames are saved in a lossless format (CRF = 0 if a codec must be used). This test isolates the encoder.

11.2.2 Capture-Device Measurement

A test video with preamble is sent through the capture device under test (commonly a USB capture stick) and recorded by OBS or by the bundled OpenCV script. Frames are saved in a lossless format. This test isolates the capture path itself.

11.3 Step 3 — Convert and Trim

Use Tools > Video to Picture (AVI to BMP or AVI to PNG) to convert the captured video into individual frames, then run Capture > Re-Seq to remove the preamble frames. After Re-Seq, frame numbering starts at one and aligns one-to-one with the reference set.

11.4 Step 4 — Frame Check Before Metrics

Run Capture > Frame drop check on the captured frames. The function OCR-reads the four-digit counter in each frame and compares it to the filename. It reports two diagnostics: Mismatched Frames (filename does not match the OCR-read number) and Missing Frames Based on Content (numbers absent from the OCR set). If either list is non-empty, fix the capture before computing metrics — quality numbers measured on a dropping path are not informative.

11.5 Step 5 — Compute Metrics

11.5.1 Approach 1 — Use vmaf.exe

Convert the reference and captured frame folders back to uncompressed AVI using Tools > BMP to AVI (or PNG to AVI), then run Measure > VMAF (video). This approach measures VMAF, PSNR, and SSIM in a single pass.

11.5.2 Approach 2 — Compute on still frames

Use Measure > PSNR/SSIM (pictures) directly on the BMP or PNG folders. This approach cannot produce a VMAF score, but it generates a composite video that lets you watch how PSNR and SSIM evolve over time.

11.5.3 Approach 3 — Colour-difference analysis

Use Measure > Color Vector Analysis (△E2000) to characterize colour shift. This is the only approach that exposes hue rotations, gamma drift, and chroma-subsampling losses.

12 Recommended Test Setup

The accuracy of every measurement in this manual depends on a clean test path. The following requirements are mandatory before trusting any number.

12.1 Source Equipment

The graphics card, the source PC, and the source player must not drop frames at the target frame rate. Use the bundled FPS Counter Player (Play button on the main panel) to verify that the rendering chain delivers each numbered frame exactly once. VLC Media Player is the recommended source player; other players have been observed to drop frames during fullscreen playback.

12.2 HDMI Path

Always insert an HDMI 2.0 splitter with a fixed EDID upstream of the device under test. The splitter ensures that the source negotiates the same resolution, colour space, and bit depth regardless of which DUT is connected, which removes a major source of cross-device variance.

Resolution, frame rate, colour space, and colour depth must remain identical end-to-end. If any link in the chain rescales or converts the signal, PSNR will collapse to meaningless values driven by the resampling rather than the device under test.

12.3 Capture Hardware

Confirm with Capture > Frame drop check that the capture device does not drop frames during a representative run before trusting any quality number. Underperforming PCIe capture cards can drop frames continuously; this is the single most common reason for unexplained low PSNR.

Ensure that the image_out folder used by the OpenCV capture script is on a fast SSD. Uncompressed 4K frames at 60 Hz produce roughly 1.5 GB per second of data; a slow disk will become the bottleneck and indirectly cause drops.

12.4 License

The application enforces a hardware licence check at startup using the bundled USB key. If the splash screen reports a missing licence, plug in the dongle before launching VQ.exe; the app will exit gracefully if the key is not detected.

13 Additional Tools

The application contains additional tools beyond those documented in the basic chapters. Each tool starts on its own page below.

13.1 Video Compressor

Located under Tools > Video PostProcessing. Re-encodes an input video using H.264 or H.265.
Quality presets map to FFmpeg CRF values: Low = CRF 51, Medium = CRF 30, High = CRF 18.
Output container can be MP4, MKV, or AVI. AVI is unavailable for H.265.
Pixel format is forced to yuv420p for downstream compatibility, so do not use this tool when 4:4:4 or 10-bit data must be preserved — use Tools > BMP to AVI (10-bit) instead.

Figure 13.1 — Same source frame at the three quality presets. CRF 18 is visually transparent, CRF 30 is the streaming default, CRF 51 introduces visible artefacts.

13.2 Video Composer

Concatenates or layers multiple input videos into a single output. Useful for building golden test reels that combine reference clips with stress-test patterns from the Advanced PG menu.

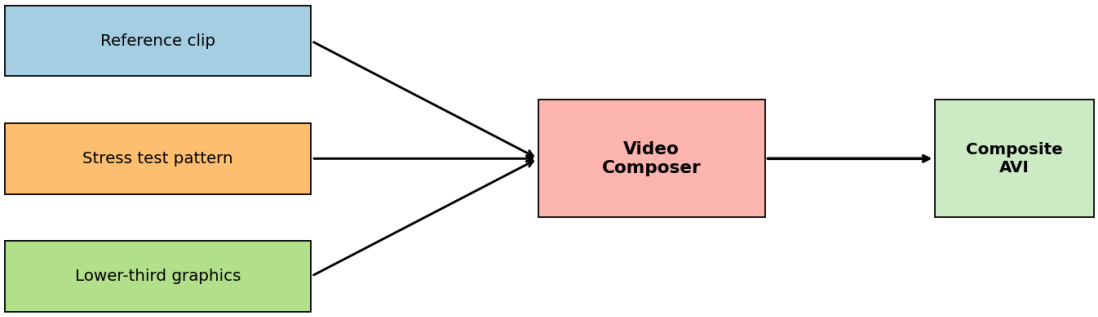


Figure 13.2 — Multiple input clips combined by the Composer into a single test reel.

13.3 Video up-to-4K Scaler

Upscales any input video to 3840 x 2160 using a Lanczos 3 kernel. Output is written as Uncompressed AVI to preserve every pixel, so downstream measurement tools observe only the rescale, not codec losses.
Use this when the device under test expects a 4K input and you only have a 1080p source available.

Figure 13.3 — Upscaling result with magnified detail crops, showing the Lanczos 3 reconstruction.

13.4 Video down-to-2K Scaler

Symmetrical to the up-to-4K Scaler. Downscales any input to 1920 x 1080 using Lanczos 3 and writes the result as Uncompressed AVI.

Figure 13.4 — Downscale result with magnified detail crops.

13.5 Add Distortion to Pictures

Synthesizes controlled distortions on a folder of BMP or PNG frames so the rest of the toolchain can be exercised without real hardware in the loop. Three distortion types are supported. The figure below shows what each distortion looks like on the same source frame.

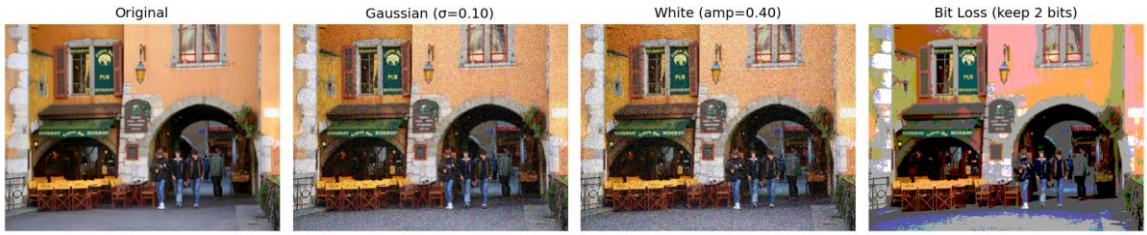


Figure 13.5 — Original frame compared to the three distortion modes (Gaussian noise, White noise, Bit Loss).

13.5.1 Gaussian Noise

Configurable mean and variance. Mean defaults to 0 and variance to 0.01. Use this to mimic sensor read noise.

13.5.2 White Noise

Configurable amplitude in 0 to 1. Default amplitude is 0.05. Use this to mimic uniformly distributed measurement noise.

13.5.3 Bit Loss

Truncates each pixel to N bits, where N is between 1 and 8. Default keeps 1 bit. Use this to model aggressive bit-depth reduction or quantization.

13.6 Sharpen

Applies a configurable sharpening pipeline to a sequence of frames. Combine with Add Distortion to model post-processing chains that some capture devices apply implicitly.

Figure 13.6 — Original vs Sharpen output (Amount = 2.0).

13.7 Generate Calibration Pattern

Creates a 24-bit RGB image populated with concentric squares whose side length increases by a configurable separation (default 300 pixels).
Resolution can be 1920 x 1080 or 3840 x 2160.
The user can shift the centre of the pattern up, down, left, or right by a configurable number of pixels. This is useful when a downstream device is known to introduce a fixed offset that must be cancelled.
Output format is BMP or PNG.

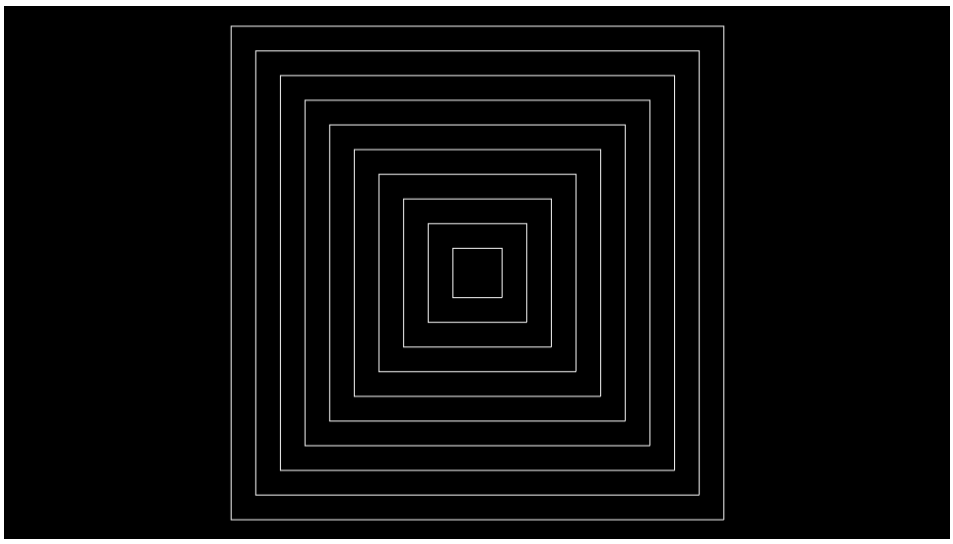


Figure 13.7 — Sample calibration pattern at 1920 × 1080 with 100-pixel separation, centred.

13.8 Verify Pattern Shift

Compares an original calibration pattern with a captured copy and reports the pixel-accurate horizontal and vertical shift between the two.
This is the recommended way to detect the small geometric offsets introduced by some HDMI splitters, scalers, and inexpensive USB capture cards before running quality metrics. A few pixels of shift will reduce PSNR and SSIM dramatically without indicating any real codec or quality fault.

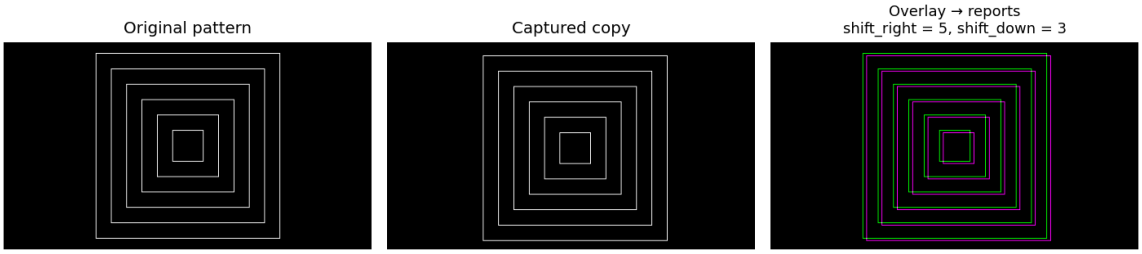


Figure 13.8 — Original calibration pattern vs a captured copy that has shifted; the green/magenta overlay reveals the offset, which the tool reports numerically.

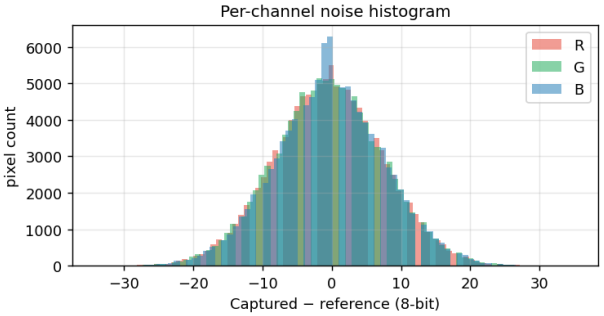
13.9 Add Noise

Injects a configurable noise floor onto a sequence of frames. Use to characterize how much extra noise the capture path is adding on top of the source.

Figure 13.9 — Clean reference frame vs the same frame with Gaussian noise (σ = 12) added.

13.10 Noise Statistics

Analyzes a captured sequence and reports per-channel mean, variance, and signal-to-noise ratio. Pair Noise Statistics on a captured run with Add Noise on a synthetic reference to tease apart the noise contribution of the capture path versus the source.



Noise Statistics output

Channel	Mean	Std σ	SNR (dB)
R	-0.511	7.854	19.58
G	-0.441	7.895	17.89
B	-0.272	7.801	17.85

Figure 13.10 — Per-channel noise histogram and the summary statistics table the tool produces.

13.11 Generate Frame Counter

Builds a 1280 x 720 MPEG-4 video with a prominent centre-screen frame counter, two user-configurable text strings, animated bouncing squares, overlapping triangles, and a background colour that cycles every 20 frames. The bundled clips frame_counter_24Hz.mp4 through frame_counter_144Hz.mp4 in the application directory are products of this tool.

Figure 13.11 — One frame from frame_counter_60Hz.mp4 showing the centre-screen counter, the bouncing squares, and the colour-cycling background.

Use these clips with the FPS Counter Player on the main panel to confirm that a display, a capture device, or a refresh-rate-conversion box renders each numbered frame exactly once.

13.12 Generate Rolling Excel BMP

Produces a stress-test pattern that overlays test content on a continuously scrolling Excel-style worksheet background. Useful for exercising motion estimation and small-text legibility on capture devices that target office and education markets.

Frame	PSNR_Y	PSNR_Cb	PSNR_Cr	SSIM	VMAF	CRF	Bitrate	Notes	Δ	Σ	σ	…
69.38	90.88	81.28	37.79	43.71	89.01	20.42	84.88	82.97	56.97	43.94	42.00	40.13
55.16	59.86	63.73	98.64	82.62	69.15	98.13	37.01	32.66	68.39	23.47	22.82	60.68
56.83	92.46	69.71	60.62	59.25	39.55	20.93	35.20	74.67	35.85	49.19	20.30	85.57
32.20	41.14	89.55	60.27	86.92	70.54	78.60	27.23	62.75	60.11	88.84	48.54	67.26
24.68	50.62	45.52	31.87	84.49	49.98	97.32	66.61	67.80	70.40	73.44	31.91	54.78
38.93	51.80	27.64	96.46	36.99	73.07	43.73	89.05	72.31	30.40	86.76	94.65	91.41
65.01	31.49	35.20	93.30	63.63	34.26	89.84	70.68	65.01	49.73	52.47	38.92	23.01
89.22	56.95	63.26	45.45	79.35	21.99	49.40	22.40	29.71	96.40	71.96	53.83	61.38
88.95	47.19	66.63	74.01	48.08	61.01	80.45	91.83	31.93	93.74	20.41	79.49	84.03
30.80	53.09	84.41	21.13	69.65	82.65	60.53	77.34	37.89	35.68	48.69	34.17	47.34
94.90	65.29	46.87	41.45	95.21	55.11	97.45	60.73	61.17	90.83	78.68	65.87	53.71
89.38	52.52	92.90	25.43	53.97	61.04	95.12	39.83	83.68	73.44	76.65	69.74	96.75
46.28	51.46	36.03	24.01	36.82	92.32	86.37	28.88	67.70	57.86	66.98	72.08	44.23
95.95	56.80	69.62	70.18	34.53	24.89	52.51	80.36	84.40	77.67	28.94	92.16	83.36
89.34	61.34	92.34	23.69	22.39	21.60	39.97	39.64	34.81	64.80	23.08	66.64	33.11
73.55	21.66	44.54	94.13	62.53	84.12	71.98	68.25	35.11	65.38	23.14	83.33	95.85
87.47	24.01	46.75	45.12	28.90	69.50	83.00	44.78	88.16	82.97	30.20	80.58	89.73
35.59	65.32	70.46	68.14	27.60	72.23	69.92	85.09	83.48	45.85	77.04	88.51	90.54
32.76	22.11	71.41	36.96	64.53	94.64	49.97	39.97	56.06	71.92	27.99	50.07	30.56
72.33	85.61	49.77	49.37	62.62	36.99	39.55	46.06	56.14	26.44	79.47	65.75	43.68
26.13	80.29	30.36	30.52	30.32	26.42	91.60	41.27	44.21	85.79	68.97	34.78	54.35

pattern overlay

Figure 13.12 — One frame of a Rolling Excel BMP sequence: a scrolling spreadsheet background with the test pattern overlaid (red highlight box).

13.13 Generate Floating Picture BMP

Animates a still image across a configurable resolution and frame count. Useful for capture-device evaluations where natural content does not exercise enough of the codec's motion-estimation logic.

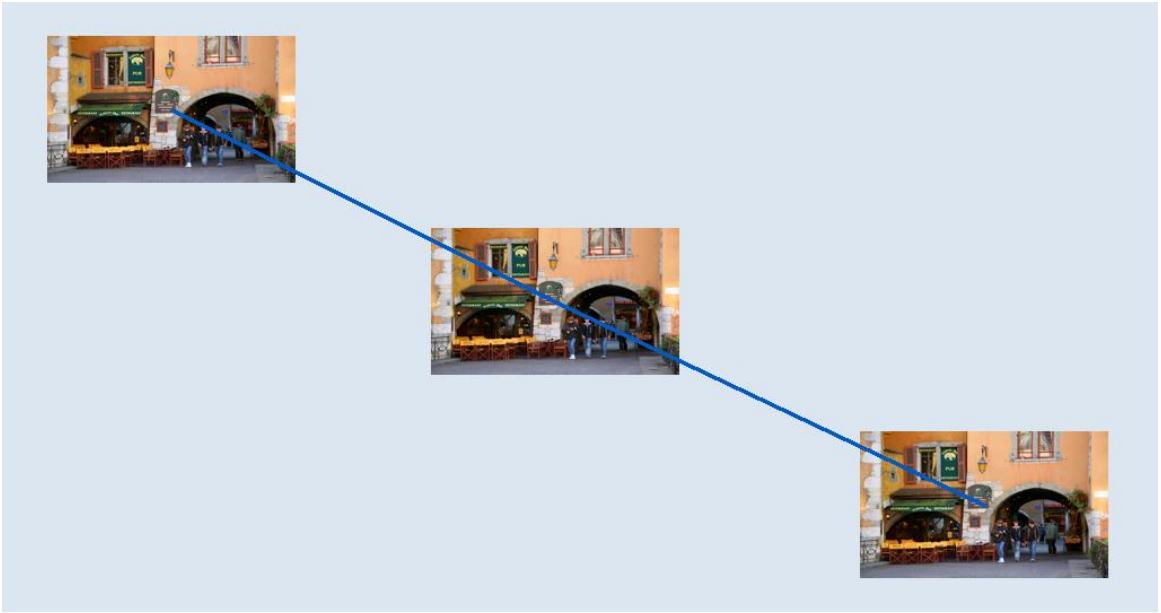


Figure 13.13 — Floating Picture motion path: the same image sampled at three frames as it traverses the canvas.

13.14 Generate Floating Colorbar

Animates a SMPTE-style colour bar across the frame. Useful for testing colour-handling consistency and chroma-subsampling losses without the variability of natural content.

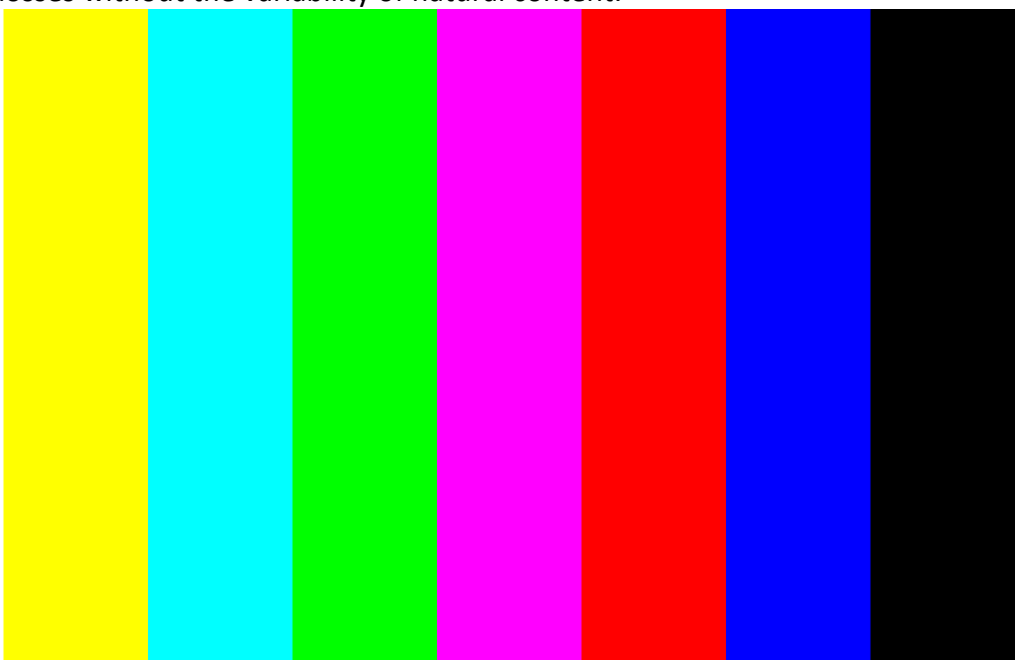


Figure 13.14 — One frame of a Floating Colorbar sequence (100 % colour bars at 1920 × 1080).

13.15 Repeat Static Picture to Sequence

Replicates a single still image into a numbered sequence (frame_0001.bmp through frame_NNNN.bmp). Use this to build a constant-image reference when you only need to characterize a device's noise or colour stability, not its temporal handling.

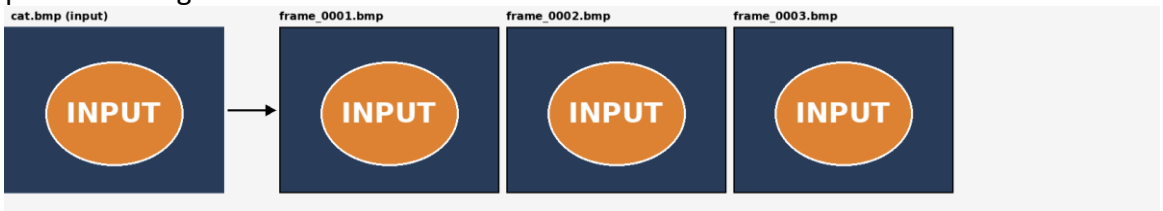


Figure 13.15 — Repeat Static Picture to Sequence: a single input file is replicated into a numbered output sequence.

13.16 Re-name Picture Sequence

Renumbers an arbitrary collection of BMP or PNG files into a clean frame_NNNN.* sequence. Often needed after manually editing or filtering captures, and required before Frame drop check can run reliably.

14 Long PSNR Measurement

Long PSNR Measurement (Measure > LongPSNRMeasurement) is designed for stability testing over thousands of frames. Most capture-device problems are intermittent: a card may pass a 600-frame run perfectly, then drop a frame two minutes later. This tool automates the long-duration loop end-to-end so the operator does not need to babysit the run.

14.1 Inputs

Reference directory containing 600 numbered frames named frame_0001.bmp (or .png) through frame_0600.*.
Capture output directory, commonly named image_out, that must be empty at the start.
Number of successful batches N to measure.
Expected capture time in seconds (used only for the progress display).
Mode selection: 1 to 600 (full block) or 51 to 550 (skip first 50 and last 50; useful when the first and last few frames of every capture are routinely unreliable).

14.2 How the Tool Runs

The tool repeatedly invokes run.cmd in the parent folder of the capture directory. Each invocation produces approximately 1199 captured frames.
After each capture batch the tool OCR-scans the captured frames to find a strictly consecutive block of either frames 1 to 600 or frames 51 to 550, depending on the selected mode.
When a clean block is found, PSNR is computed for that block versus the reference and the per-frame results are written to a timestamped LongPSNR_yyyymmdd_HHMMSS.xlsx file inside the capture output directory.
When a clean block is not found (for example because the capture device dropped a frame), the failure is recorded on the FailureBatches sheet and the tool keeps capturing until N successful batches have been collected. Failures do not consume the N counter.

14.3 Output Workbook

Long PSNR — BatchSummary sheet (sample run)

Batch	CapturedTota	SeqStartIndex	NumInf	MeanPSNR	StdPSNR	CapSec	OCRSec	PSNRSec	Note
1	1199	90	12	47.21	0.18	23.4	11.7	9.6	OK
2	1199	92	11	47.18	0.21	23.5	11.9	9.6	OK
3	1198	91	13	47.25	0.16	23.4	11.8	9.7	OK
4	1199	88	8	47.05	0.34	23.6	12.1	9.5	OK (1 outlier)
5	1199	90	12	47.2	0.19	23.4	11.7	9.6	OK

Figure 14.1 — BatchSummary sheet from a sample five-batch run.

The resulting Excel workbook contains four sheets:
PerFrame — every measured frame with its PSNR and the source filenames.
BatchSummary — one row per batch with mean PSNR, standard deviation of PSNR, and total infinity count (frames that matched the reference exactly).
OverallSummary — totals across all batches.
FailureBatches — every attempt that failed to produce a clean block, with the reason and a preview of the first eighty OCR results.

14.4 Reading the Results

Use the BatchSummary mean PSNR to track average device performance over the run. Use the BatchSummary standard deviation to detect devices that pass on average but produce occasional bad frames; those will show low mean drift but unusually large standard deviation.

A healthy capture device shows tight standard deviation (typically below 0.5 dB) across all batches. A device that is borderline acceptable will show occasional spikes in standard deviation that correspond to frame mis-alignments or one-off corruption.

15 Color Vector Analysis

Two complementary colour-space tools are available under the Measure menu. Both compare a folder of reference frames against a folder of captured frames sharing the same filenames.

15.1 Color Vector Analysis (VScope)

The VScope variant produces a 4K composite video that lets the operator visually inspect the colour difference between reference and capture, plus a CSV with per-frame metrics. The composite video uses the four-quadrant layout shown below.

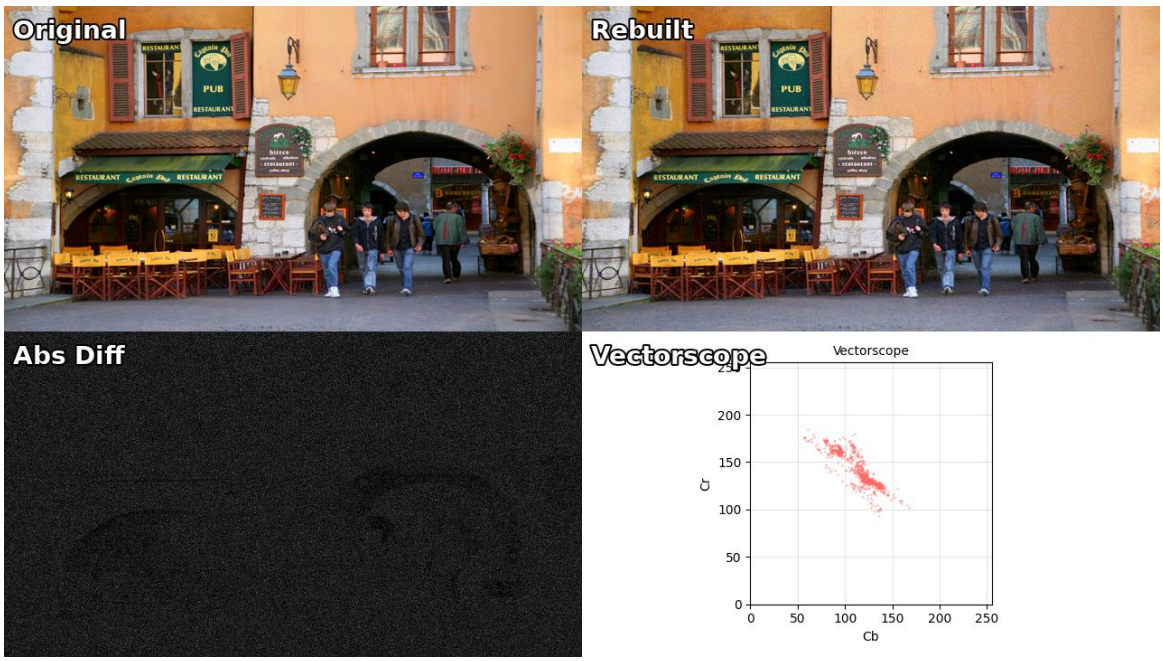


Figure 15.1 — VScope composite layout: Original (top-left), Rebuilt (top-right), Absolute Difference (bottom-left), Vectorscope of the rebuilt frame (bottom-right).

15.1.1 Output Files

diff_video.avi — absolute luminance difference per frame, normalized so that the largest difference of the run becomes white.

composite_video.avi — a 3840 x 2160 canvas split into four 1920 x 1080 quadrants showing the Original frame (top-left), the Rebuilt (captured) frame (top-right), the Absolute Difference image (bottom-left), and a Vectorscope view of the rebuilt frame (bottom-right).

ColorBarDifferences.csv — per-frame mean RGB values, ΔE2000 mean and maximum, and raw RGB pixel differences.

MeanColorDifference.png — a plot of mean ΔE2000 over the run.

15.1.2 Corner Exclusion

An option exists to exclude the bottom-right corner from colour measurement, since that area carries the preamble frame-number stamp and would otherwise pollute the score.

15.2 Color Vector Analysis (△E2000)

The △E2000 variant is similar to VScope but reports CIEDE2000 instead of CIE76 △E.

CIEDE2000 corrects for the fact that human eyes perceive colour differences non-linearly, applying hue, chroma, and lightness weighting. It is the recommended colour-difference metric for any modern display-oriented test, as discussed in the Concepts and Metrics chapter.

The output CSV includes both the mean and the maximum △E2000 per frame. The maximum is useful for detecting localized colour faults that would average out at the frame level.

Figure 15.2 — Original frame, captured frame (with hue shift), and per-pixel ΔE2000 heatmap.

16 Advanced Pattern Generators

The Advanced PG menu contains specialized pattern-generation tools intended to stress codecs and capture devices in ways that natural content rarely does. Each tool prompts for resolution, frame rate, and duration, and writes an uncompressed AVI file. Each pattern is documented on its own page below.

16.1 CoCo Database Synthesizer

16.1.1 Description

Composes random tiles drawn from the COCO image dataset into a configurable grid: 2, 3, 4, 6, 8, or 16 images per frame.

16.1.2 Purpose

Acts as the closest substitute for natural content with high spatial and temporal complexity. Because the tiles change every few frames, the encoder is forced to handle frequent scene transitions inside the same frame.

16.1.3 Example

A 4K60 video where each frame contains four randomly selected COCO photographs at 1920 x 1080 each, with the photographs rotating every 20 frames.

Sample CoCo Synthesizer frame: four random photographs tiled into a single 4K canvas.

16.2 Dynamic Range Stress Patterns

16.2.1 Description

Patterns with a wide dynamic range, including both very bright and very dark areas.

16.2.2 Purpose

To test the codec's ability to handle high contrast and detail in shadows and highlights.

16.2.3 Example

A video sequence featuring alternating bright and dark bars that gradually shift positions, combined with scenes that have both bright highlights and deep shadows.

Sample Dynamic Range Stress frame: alternating bright/dark bars with extreme highlight and shadow regions.

16.3Complex Motion Sequences

16.3.1 Description

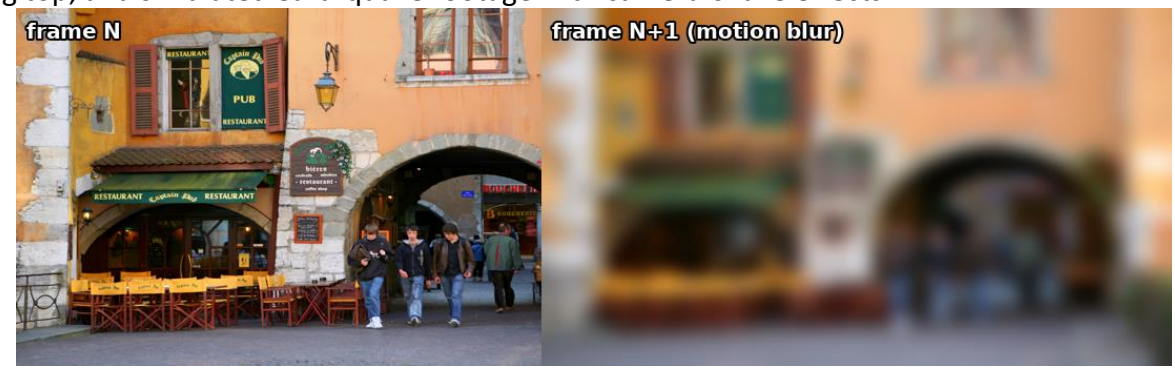
Video sequences with fast and irregular motion, such as rotating objects, shaking-camera effects, or scenes with many moving elements.

16.3.2 Purpose

To stress-test motion estimation and compensation algorithms in codecs.

16.3.3 Example

Footage of a crowded street scene with people moving in various directions, a rotating object such as a spinning top, and simulated earthquake footage with camera-shake effects.



Sample Complex Motion frame pair: sharp frame N vs heavily motion-blurred frame N+1.

16.4 High-Frequency Detail Patterns

16.4.1 Description

Sequences with intricate detail and texture, such as finely detailed grass, leaves, or noise patterns.

16.4.2 Purpose

To test the codec's ability to retain detail without introducing artifacts.

16.4.3 Example

A field of grass blowing in the wind, close-up footage of leaves rustling, or a sequence showing static white noise.

Sample High-Frequency Detail frame: dense grass-like vertical-line texture with per-pixel chrominance variation.

16.5 Rapid Scene Changes

16.5.1 Description

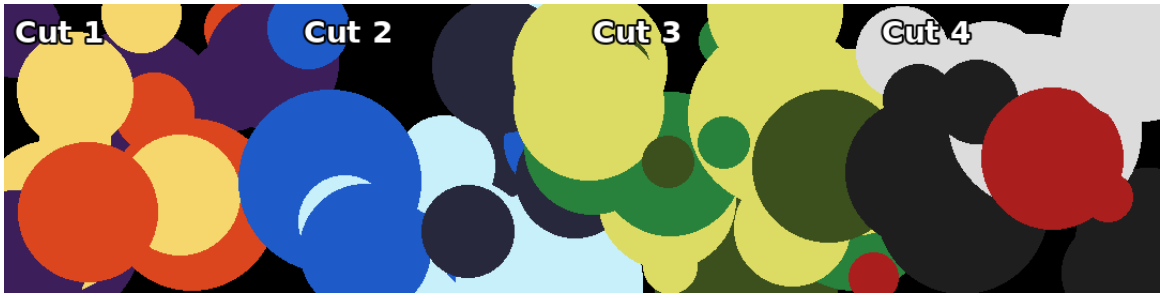
Sequences with rapid cuts between vastly different scenes.

16.5.2 Purpose

To challenge the codec's ability to handle abrupt changes in content. Particularly relevant for codecs that rely on temporal prediction across GOP boundaries.

16.5.3 Example

An action-movie trailer with quick cuts between scenes, or a video montage switching rapidly between different environments and activities.



Sample Rapid Scene Changes filmstrip: four cuts in succession with dramatically different palette and content.

16.6 Artificial Noise Patterns

16.6.1 Description

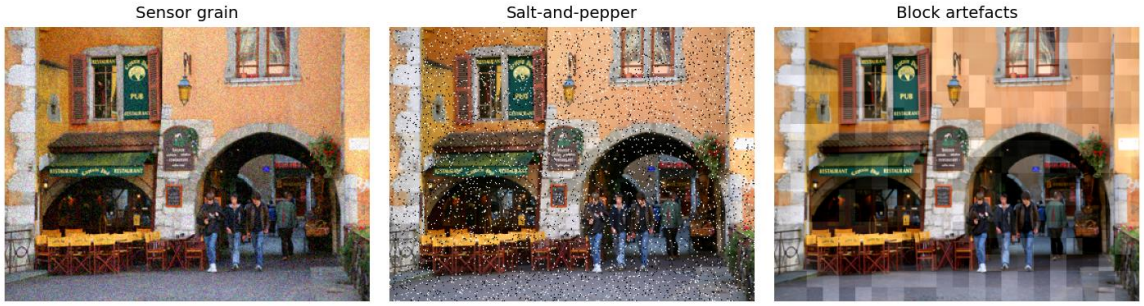
Sequences with various types of noise, including grain, compression artifacts, or other synthetic noise patterns.

16.6.2 Purpose

To test the codec's robustness to noise and its ability to avoid amplifying it.

16.6.3 Example

Simulated camera-sensor noise, artificial grain added to a video sequence, or a sequence designed with compression artifacts to test noise handling.



Sample Artificial Noise Patterns: sensor grain, salt-and-pepper, and block-artefact variants on the same source.

16.7 Mixed Content

16.7.1 Description

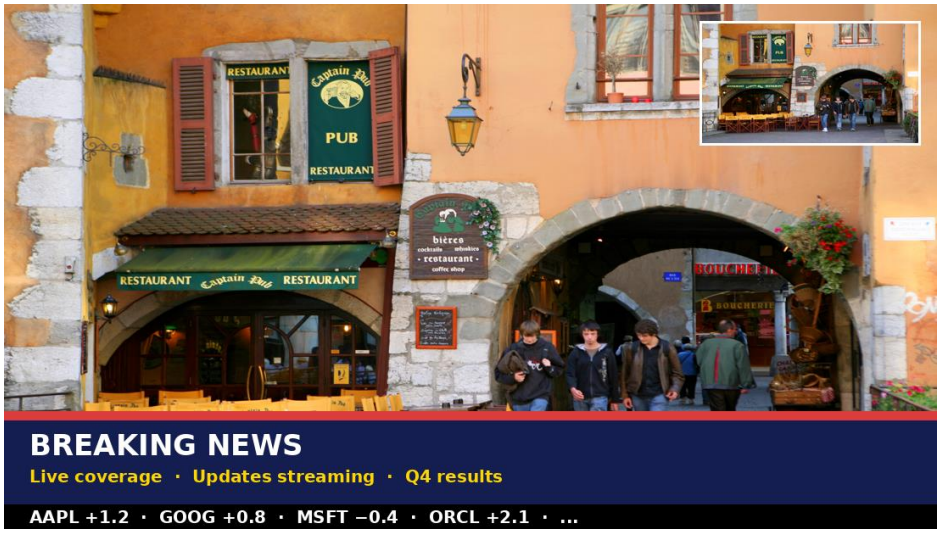
Sequences that mix different types of content, such as combining live-action footage with animated graphics.

16.7.2 Purpose

To test the codec's flexibility and adaptability to different types of content. This is the typical broadcast workflow case.

16.7.3 Example

A news broadcast with live video, animated lower thirds, scrolling text, and picture-in-picture elements.



Sample Mixed Content frame: live background with PIP, lower-third graphics, headline bar, and stock ticker.

16.8 High Bitrate / Resolution Patterns

16.8.1 Description

4K or higher-resolution videos with high bitrates.

16.8.2 Purpose

To test the codec's ability to handle high-resolution content efficiently when bandwidth is not the limiting factor.

16.8.3 Example

Detailed panoramic shots of cityscapes or natural landscapes in 4K or 8K, or a high-detail sports event recorded in 4K resolution.

Sample High Bitrate / Resolution frame: dense cityscape with thousands of small bright windows.

16.9 Low Bitrate / Resolution Patterns

16.9.1 Description

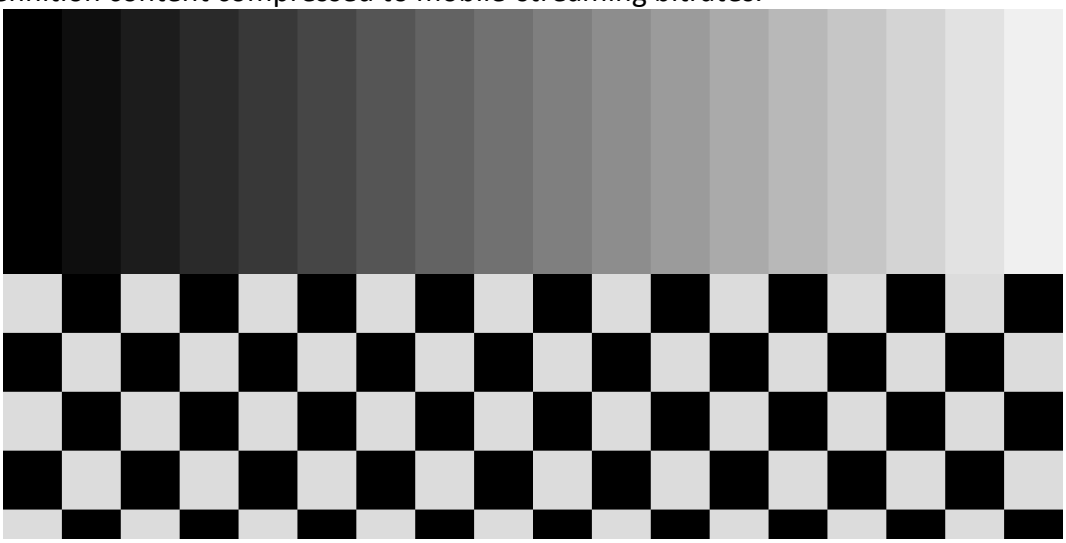
Videos compressed at very low bitrates, designed to expose how the codec degrades when the bitrate budget is tight. Sub-patterns include Low Detail Gradients, Static Noisy Backgrounds, Slowly Moving Checkerboards, and Sparse Details.

16.9.2 Purpose

To test the codec's performance in preserving quality under constrained bitrate conditions.

16.9.3 Example

Standard-definition content compressed to mobile-streaming bitrates.



Sample Low Bitrate / Resolution frame: heavy banding gradient (top) plus a slowly-moving checkerboard (bottom).

16.10 Synthetic Test Sequences

16.10.1 Description

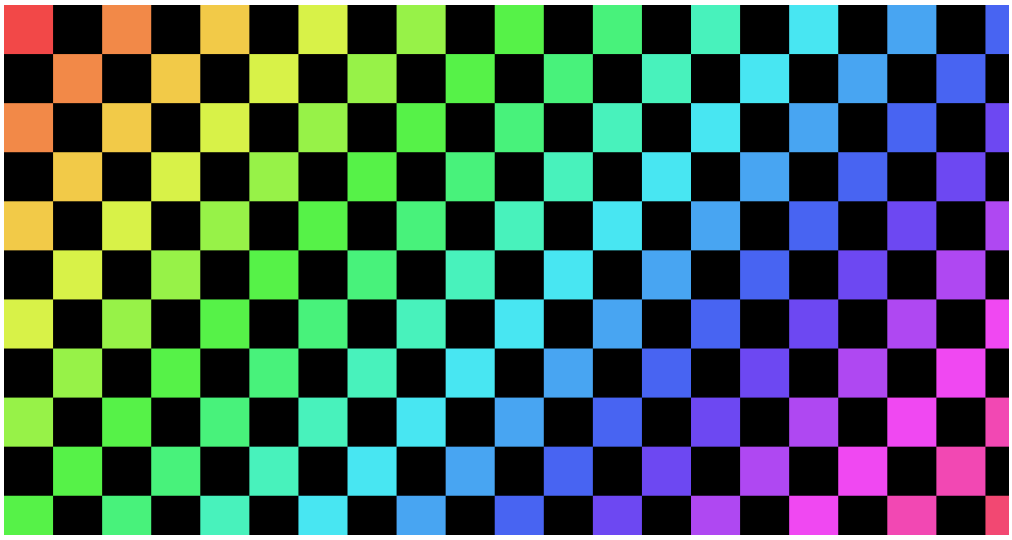
Sequences generated specifically for testing purposes, including sequences designed to break typical assumptions made by codecs.

16.10.2 Purpose

To identify weaknesses in codec algorithms.

16.10.3 Example

Chessboard patterns with shifting colours, rotating spirals, or sequences with fast-moving geometric shapes of varying sizes and colours.



Sample Synthetic Test frame: chessboard with hue shifting along the diagonal — designed to break codec assumptions about local correlation.

16.11 Natural Scenes with Unpredictable Motion

16.11.1 Description

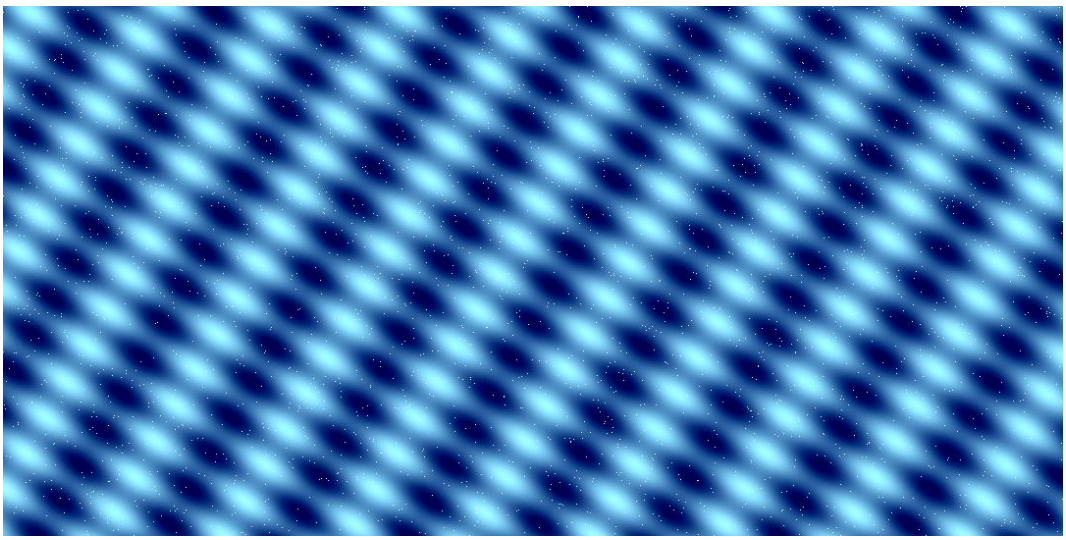
Real-life scenes with unpredictable and non-repetitive motion. Sub-patterns include Ocean Waves, Busy Marketplace, Flocks of Birds, and Dynamic Patterns with Multiple Elements.

16.11.2 Purpose

To provide a real-world challenge that is difficult for codecs to predict and compress.

16.11.3 Example

A video containing fast-moving geometric shapes of varying sizes and colours on a high-detail textured background, changing every few frames. This is designed to overload the codec's prediction and transformation mechanisms.



Sample Natural Scenes frame: ocean-wave pattern with white-foam specks — high-frequency motion with no temporal predictability.

16.12 Rainbow Diamond Pattern

16.12.1 Description

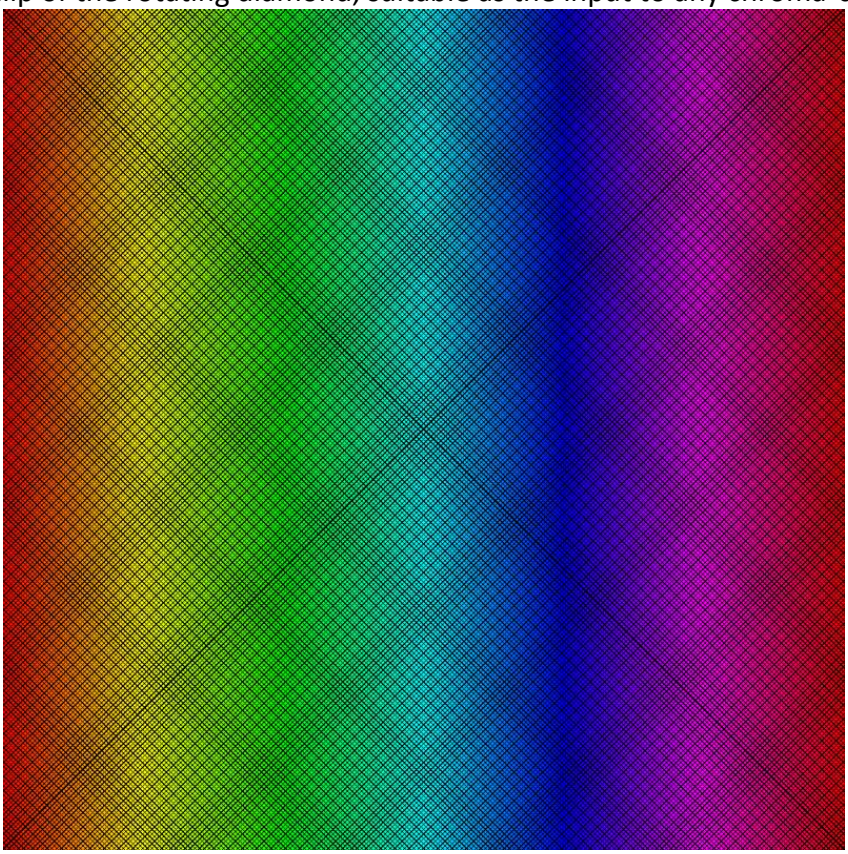
A rotating, chroma-rich diamond pattern.

16.12.2 Purpose

To stress chroma-subsampled paths. The pattern is particularly aggressive because it forces fine-grained chroma changes at every pixel boundary, which is precisely where 4:2:0 and 4:2:2 subsampling lose information.

16.12.3 Example

A 4K60 ten-second clip of the rotating diamond, suitable as the input to any chroma-conversion test.



Sample Rainbow Diamond frame (rainbow_diamond_pattern.png from the application directory).

17 Bandwidth Reference Tools

Three buttons on the main panel give quick access to bandwidth tables. They are intended to help operators sanity-check whether a planned test fits within the bandwidth budget of the chosen capture device and host system.

17.1 Show Bandwidth Table

Opens a built-in reference table, generated from video_params.txt, that lists video-link and USB-link bitrates for common combinations of resolution, frame rate, colour depth, and pixel format.

Bandwidth Table — sample entries

Sampling	Resolution	FPS	Depth	Colors	PixFmt	Video Gbps	USB Gbps
444	3840x2160	120	8	3	NV12	23.89	11.94
444	3840x2160	60	10	3	YUY2	14.93	7.96
444	3840x2160	60	10	3	NV12	14.93	5.97
444	3840x2160	60	8	3	YUY2	11.94	7.96
444	3840x2160	60	8	3	NV12	11.94	5.97
422	3840x2160	60	8	3	YUY2	7.96	7.96
422	3840x2160	60	10	3	P010	9.95	7.46
422	3840x2160	60	10	3	YUY2	9.95	7.96
420	3840x2160	60	10	3	YUY2	7.46	7.96
420	3840x2160	60	8	3	NV12	5.97	5.97
420	3840x2160	60	10	3	P010	7.46	7.46
420	3840x2160	60	10	3	YUY2	7.46	7.96
420	3840x2160	60	10	3	NV12	7.46	5.97
444	3840x2160	50	10	3	YUY2	12.44	6.64

Figure 17.1 — Bandwidth Table rendered from the bundled video_params.txt / bitrate_results.txt.

Cross-check the predicted USB bitrate against the negotiated bandwidth of the capture device before suspecting the codec or the host system.

17.2 Export BW Input Table

Saves the raw input cases (the video_params.txt content) as a CSV so that the operator can build their own bitrate calculator or feed the data into a procurement spreadsheet.

17.3 Show User BW Table

Renders the same kind of table using a user-supplied parameter file. Useful when evaluating non-standard formats that are not in the bundled list.

18 Practical SOP Recipes

This chapter presents end-to-end recipes for the most common evaluation scenarios. Each recipe assumes the recommended test setup has been verified.

18.1 Recipe A — Evaluating a USB Capture Card

18.1.1 Step 1 — Build the master

Begin with a 1920 x 1080P60 reference clip. Run Capture > Preamble to insert the 1/2/3 sync frames and stamp every frame with a four-digit number. This writes a BMP sequence plus an M_<name>.avi master video.

18.1.2 Step 2 — Rebuild a clean reference

Use Tools > BMP to AVI to rebuild a clean reference video from the BMPs produced in step 1. The rebuilt video is the one you will play through the test path.

18.1.3 Step 3 — Capture

Connect the source through a fixed-EDID HDMI 2.0 splitter to the USB capture device. Play the master with VLC in fullscreen, then run run.cmd. Wait for the capture to complete.

18.1.4 Step 4 — Trim and verify

Run Capture > Re-Seq starting from frame 90 (or wherever the preamble ends in your specific master) to drop the preamp frames and renumber. Then run Capture > Frame drop check. If frame drops are reported, fix them before proceeding.

18.1.5 Step 5 — Measure

Run Measure > PSNR/SSIM (pictures) on the reference and captured directories with Pmax = 255. For codec-based devices, additionally use Tools > BMP to AVI on both directories to produce VMAF_ori.avi and VMAF_dis.avi, then run Measure > VMAF (video). Save the resulting CSV; you can compare against future captures with Measure > VMAF Comparison.

18.2 Recipe B — Comparing Two Encoders

18.2.1 Step 1 — Encode both candidates

Encode the same reference clip through both encoders at matching bitrates. Decode each output back to a video file.

18.2.2 Step 2 — Measure each candidate

Run the standard PSNR/SSIM and VMAF measurement against the reference for each decoded output. Save both CSVs into a single working directory.

18.2.3 Step 3 — Compare side by side

Run Measure > VMAF Comparison and select 2 as the file count. The tool produces a side-by-side plot and a PDF report annotated with the value of the Reported by field on the main panel.

18.2.4 Note on interpretation

Be cautious about absolute scores. The score that matters is the difference between the two curves on the same content; absolute VMAF and PSNR figures depend strongly on the content type.

18.3 Recipe C — Long-Run Stability Test

18.3.1 Step 1 — Prepare directories

Prepare a 600-frame reference directory and an empty image_out directory next to a working run.cmd script.

18.3.2 Step 2 — Choose mode and N

Choose the 51 to 550 mode if the first 50 and last 50 frames of your captures are routinely unreliable, otherwise use the default 1 to 600 mode. Set N to the number of clean batches you want; ten batches at 600 frames each is a good starting point.

18.3.3 Step 3 — Run and review

Launch Measure > LongPSNRMeasurement and let it run unattended. After completion, open the LongPSNR_<timestamp>.xlsx file and check both the BatchSummary mean and standard deviation columns.

18.4 Recipe D — Color Accuracy of an HDMI Splitter or Scaler

18.4.1 Step 1 — Verify geometry

Run Tools > Generate Calibration Pattern at the target resolution, capture the pattern through the splitter, and then run Tools > Verify Pattern Shift to confirm geometric alignment. Fix any non-trivial shift before proceeding.

18.4.2 Step 2 — Run a real-content reference

Run a real-content reference clip through the splitter and capture as usual.

18.4.3 Step 3 — Compute △E2000

Run Measure > Color Vector Analysis (△E2000) on the reference and captured directories. Enable corner exclusion if the captures contain the preamble frame-number stamp.

18.4.4 Deliverables

The MeanColorDifference.png plot and the per-frame CSV are the deliverables. Refer to the △E2000 interpretation table in the Concepts chapter to translate the score into a perceptual statement.

18.5 Recipe E — Frame Drop Diagnostics

18.5.1 Symptom

Frame drop check reports many false positives, typically because a capture is dirty enough that OCR fails on individual frames.

18.5.2 Step 1 — Manual cleanup

Remove obvious black or torn frames from the capture directory.

18.5.3 Step 2 — Renumber

Run Tools > Re-name picture sequence to renumber what remains.

18.5.4 Step 3 — Re-run the check

Re-run Frame drop check. Compare the Mismatched Frames list against the Missing Frames Based on Content list.

18.5.5 Interpretation

A long Mismatched list with an empty Missing list usually indicates a stable capture path with file-rename issues. A long Missing list indicates real frame drops in the path.

19 Final Notes

All numerical results in this manual depend on a clean reference, a correctly aligned distorted stream, and a stable capture path. Use Capture > Frame drop check and Tools > Verify Pattern Shift before trusting any quality score.

When publishing a result, always include the resolution, frame rate, colour space, colour depth, and the version of the analyzer (visible in Help > About). The Reported by field on the main panel propagates to every generated PDF and CSV header.

No single metric is sufficient. Run PSNR, SSIM, and ΔE2000 on every capture, and add VMAF whenever the test path includes a video encoder.