Gain Maps

Eric Chan (erichan@adobe.com)

Version 1.0 draft 15

February 28, 2024

Note on Standardization Effort

This document was originally written by Adobe in late 2021 as a technical proposal. Concepts and algorithms described in this document are currently undergoing standardization in ISO/TC 42 Photography. Please be aware that this document is not endorsed by ISO and may not be functionally equivalent to the forthcoming ISO standard.

Overview

We would like to display HDR content and take advantage of the additional headroom offered by modern displays.

A major issue with HDR content is the lack of consistency in appearance across different devices and platforms, with respect to both color and tone.

This document proposes a method of extending existing still image file formats to support HDR content, with compatibility, consistency, and predictability as major goals. Video is considered out-of-scope for this proposal.

The primary use case is exporting images to be distributed, shared, and viewed in apps such as the Finder/Explorer, the camera roll, native gallery apps, social media apps, and web browsers.

Goals

Our primary goals are:

- Take advantage of HDR display technology for more exciting and impactful photos
- Adapt dynamically to the current display capabilities & viewing conditions
- Avoid keeping track of separate SDR and HDR renditions (two or more separate files)
- Keep the photographer in control of the HDR-to-SDR tone mapping process, instead of leaving it up to platform-specific behavior
- Support local adaptation in tone mapping
- Provide consistency and predictability for how HDR content is displayed across apps, devices, and platforms
- Provide backward compatibility for legacy software

- Minimize image file storage overhead
- Minimize complexity of additional runtime display logic

Concept

The idea is to store a “Base” rendition and a “Gain Map.” The Base rendition is the main image in the file. The Gain Map is a secondary image and associated metadata.

The Base may be either SDR or HDR. If the Base is SDR, then applying the Gain Map to the Base produces the HDR rendition. The concept also works in the other direction: if the Base is HDR, then applying the (inverse) Gain Map produces the SDR rendition. A single Gain Map can both brighten & darken.

An advantage of choosing the Base as the SDR rendition is built-in compatibility with older image software readers and older displays (i.e., by ignoring the optional Gain Map tag).

Image readers may interpolate the values in the Gain Map dynamically at display time to accommodate current display capabilities and viewing conditions. This runtime interpolation is crucial to the flexibility of the solution.

How to Compute the Gain Map

The following discussion assumes that the authoring app has created SDR and HDR renditions. The method by which these renditions are obtained is unimportant (they can be automatically generated, manually adjusted by the user, etc.). The two renditions should be linear gamma and share the same colorimetry (e.g., same RGB primaries, such as Rec. 2020 or P3).

1. Optionally downsample the renditions to a lower resolution.
2. Optionally convert the renditions to grayscale.
3. Compute the Gain Map G = $\log_2$ ((HDR+k_{hdr}) / (SDR+k_{sdr})). This is either a 1-plane or 3-plane log (base 2) gain map, where 0 means no change, negative values mean darken, and positive values mean brighten. The parameters k_{hdr} and k_{sdr} are intended to ensure a well-defined gain map. They must be chosen so that the numerator and denominator are both positive, since the domain of the log function is positive.
4. Find the minimum, mean, & maximum values of each color plane of G. For robustness, consider excluding a small percentage of the smallest & largest values from this calculation.
5. Remap each color plane of G with an affine transform so that the minimum and maximum values map to 0 and 1, respectively, and the resulting pixel values are clipped to [0, 1].
6. Optionally map each color plane of G with a gamma function; the gamma value may be different for each plane.
7. Choose a bit depth N for encoding (e.g., N = 8 bits). Scale the normalized gain map data to the range 0 to 2^N-1 (e.g., 255 for 8-bit): f(x) = floor(x * (2^N – 1) + 0.5)
8. Compute and store metadata (see next section)
9. Optionally compress the gain map.

Gain Map Metadata

The following metadata should be stored along with the Gain Map image:

1. Per-channel min and max Gain Map values (log base 2)
2. Per-channel gamma values
3. Per-channel offset constants k_{sdr} and k_{hdr}
4. Minimum and maximum HDR capacity values for interpolating the Gain Map
5. Boolean flag indicating whether the Base rendition is SDR or HDR
6. A version tag for compatibility purposes

The minimum and maximum HDR capacity values are specified in log base 2 space and can be set by the authoring app to control how the Gain Map is interpolated; see next two sections for details.

How to Apply the Gain Map

1. Start with the Base (e.g., SDR) rendition in linear gamma. For example, if the Base rendition is in Display P3, then undo the 2.2 gamma encoding.
2. Read the Gain Map and corresponding metadata. For N-bit integer encoded gain maps, obtain the floating-point value as follows: $g(x) = x / (2^N - 1)$
3. Resample the Gain Map if needed to match the Base image dimensions. Implementations can use simple methods (e.g., bilinear) or more advanced methods (e.g, edge-preserving). It is recommended to avoid nearest-neighbor (point sampling) for quality reasons.
4. Use the gamma, minimum & maximum values and k_{sdr} and k_{hdr} offsets per color plane to invert the affine transform and produce the $\log_2$ Gain Map.
5. Scale the Gain Map by some weight W based on the metadata and the current display conditions. More details in the next section.
6. Apply 2^x to produce a linear space Gain Map.
7. Multiply Base image by the Gain Map from the previous step. If the Gain Map is grayscale (1 plane), multiply each of the R, G, and B pixels of the Base image by the same factor.

$$HDR = (SDR + k_{sdr}) \cdot 2^G - k_{hdr}$$

$$SDR = (HDR + k_{hdr}) \cdot 2^G - k_{sdr}$$

The result is a linear gamma rendition that can be composited with other SDR and HDR linear content; note that this image may have negative values. It can also be encoded with a standard HDR transfer curve (e.g., Perceptual Quantizer, Hybrid Log-Gamma) for compatibility with downstream APIs.

How to Scale the Gain Map

The purpose of scaling the Gain Map is to interpolate between SDR and HDR renditions. This should be done dynamically at display time to account for the display's current HDR capacity.

To scale the Gain Map, multiply each pixel in the Gain Map by a weight W.

The scaling should be done while the Gain Map values are in log space to provide a perceptually meaningful interpolation between the two renditions.

Implementations should choose the weight W based on:

- H, the display's current **HDR capacity** (ratio of HDR to SDR white) expressed as a log base 2 value. A SDR display has $H = 0$.
- M_{lo} and M_{hi}, the minimum and maximum HDR capacity values specified in the Gain Map metadata.

Here is a recommended implementation for the case where the Base rendition is SDR.

Compute the weight W (a constant over all pixels in the image):

$$F = clamp\left(\frac{H - M_{lo}}{M_{hi} - M_{lo}}, 0, 1\right)$$

If the Base rendition is SDR, then set the weight W = F. This causes the gain map to be applied in the "forward" direction to map SDR to HDR.

If the Base rendition is HDR, then set the weight W = F – 1. This causes the gain map to be applied in the "inverse" direction to map HDR to SDR.

Notes on the above method:

- When $H \leq M_{lo}$, then the result is the SDR rendition.
- When $H \geq M_{hi}$, then the result is the HDR rendition.
- When $M_{lo} < H < M_{hi}$, then the result is somewhere in between the SDR and HDR renditions. This is the case where the display currently does not have enough highlight headroom to display the full HDR rendition without clipping. The resulting interpolated Gain Map should keep the pixel values in the derived rendition approximately within the currently displayable range.

Recommended Usage of Minimum and Maximum HDR Capacity

The minimum and maximum HDR capacity values in the gain map metadata provide authoring apps with a way to control the range of HDR capacity values over which tone mapping is applied.

For example, suppose a photographer is editing an image that has a small number of extremely bright pixels (e.g., specular highlights). The brightest pixel in the HDR rendition (e.g., +6 stops) might exceed the authoring screen's HDR capacity (e.g., +3 stops). The photographer or authoring app may prefer to set the maximum HDR capacity tag value to 3 instead of 6 to avoid darkening the image too much on less capable displays. By setting the max HDR capacity value to 3, the author indicates that tone mapping should not be applied when the viewing display's current HDR capacity is at least 3.

Interpolation vs Extrapolation

Implementations should avoid using weight values outside the range [-1,+1], i.e., avoid extrapolating the gain map to fill the displayable range.

Pseudocode for Calculating Gain Map

The following is GPU shader pseudocode for computing a gain map. Note that it is the authoring app's responsibility to ensure that the resulting gain map image is well-behaved.

```
vec3 CalculateGainMap (image2d sdrImage,
                       image2d hdrImage,
                       vec2 position)
    {

    // Read SDR image. This should be gamma 1.0.

    vec3 sdr = ReadImage (sdrRendition, position);

    // Read HDR image. This should be gamma 1.0 and
    // have the same RGB primaries as the SDR rendition.

    vec3 hdr = ReadImage (hdrRendition, position);

    // Offset parameters, which need to be written to the
    // Gain Map metadata so that the transform can be reversed later.

    constexpr vec3 k_sdr = vec3 (1.0-5f);
    constexpr vec3 k_hdr = vec3 (1.0-5f);

    // Calculate gain map in log2 space.

    return log2 ((hdr + k_hdr) / (sdr + k_sdr));

    }
```

Pseudocode for Applying Gain Map

```
vec3 ApplyGainMap (image2d baseImage,
                   image2d gainMap,
                   vec2 position,
                   float w, // weight for applying the gain map
                   vec3 gainMapMin,     // from gain map metadata
                   vec3 gainMapMax,     // from gain map metadata
                   vec3 gainMapGamma,   // from gain map metadata
                   vec3 offsetBase,     // from gain map metadata
                   vec3 offsetOther)    // from gain map metadata
    {

    // Read base image (e.g., SDR). This should be gamma 1.0.

    vec3 base = ReadImage (baseImage, position);

    // Read gain map in normalized [0,1] space.

    vec3 gainMapEncoded = ReadImage (gainMap, position);

    // Undo gamma & affine transform; result is in log 2 space.

    vec3 gainMapLog2 = lerp (gainMapMin,
                             gainMapMax,
                             pow (gainMapEncoded,
                                  vec3 (1.0) / gainMapGamma));

    // Apply weighted gain map to image.
    //
    // If Base = SDR, then offsetBase is k_sdr and offsetOther is k_hdr.
    // If Base = HDR, then offsetBase is k_hdr and offsetOther is k_sdr.

    return ((base + offsetBase) * exp2 (gainMapLog2 * w)) - offsetOther;

    }
```

Encoding Notes

Log base 2 is a convenient representation because it can be scaled meaningfully based on current display conditions. Also, flipping the sign of the Gain Map (i.e., -G) allows it to be applied in the opposite direction (e.g., from HDR to SDR, instead of from SDR to HDR). In fact, the same display logic works in both directions by allowing the weight parameter W to be signed (from -1 to +1).

Color vs Gray Notes

The Gain Map can be either color (RGB, 3 planes) or grayscale (1 plane). A grayscale Gain Map requires less storage but cannot represent advanced color mapping techniques often used in HDR-to-SDR tone mapping. For example, it is common when fine-tuning a SDR rendition to adjust both contrast and saturation; capturing both requires a color Gain Map.

Resolution Notes and Resampling

A full-resolution Gain Map maximizes accuracy but requires more storage; therefore, it doesn't offer much advantage over storing two full renditions. A lower resolution Gain Map requires less storage but may result in missing high-frequency details (e.g., small bright points of light). A reasonable starting point for experimentation is ¼ resolution (½ width and ½ height).

Lower-resolution Gain Maps can be resampled to match the Base resolution using a variety of methods. It is recommended to avoid using nearest-neighbor (point sampling) for quality reasons.

Bit Depth Notes

Ideally store the Base rendition using at least 10 bits per component.

However, in many cases very good visual image quality can still be achieved by storing a Base SDR rendition using only 8 bits per component. Doing so enables backward compatibility in the short term with existing, widely adopted 8-bit file formats and codecs such as JPEG.

The bit depth of the Gain Map should be at least 8 bits per component. The bit depth of the Gain Map does not need to match the bit depth of the Base.

Normally, using HDR transfer functions to represent an integer-coded rendition (such as Hybrid Log-Gamma or Perceptual Quantizer) means that at least 10 bits per component are needed for that rendition; otherwise, artifacts such as banding may occur. In a HLG or PQ rendition, roughly half the encoding space is used to represent the SDR content and the other half represents the HDR or overrange content (if any). If that is an 8-bit rendition, then only 128 levels represents the SDR content; in other words, each of the SDR and HDR content effectively gets only a 7-bit representation.

The Gain Map approach with 8-bit Base rendition = SDR is a little better, because the SDR content can make full use of the 8-bit encoding space; the HDR content is stored in another 8-bit image (the Gain Map).

When storing the normalized gain map image using N-bit integers, use the following rounding convention:

$$f(x) = \text{floor}(x * (2^N - 1) + 0.5)$$

When reading N-bit integer gain map images, use the following equation to obtain the normalized gain map:

$$g(x) = x / (2^N - 1)$$

Color Space, Gamut, and Overrange Values

The SDR and HDR renditions are defined using the same linear color space (the same color primaries with gamma 1.0), but their contents may have different color gamuts. For example, the Base rendition

may be SDR in (linear) sRGB, but the HDR rendition may have a wider gamut such as Rec 2020. Expressing this wide-gamut HDR rendition using the narrow-gamut primaries (sRGB) means that the HDR rendition may have pixel values less than zero and/or pixel values greater than 1. Note that overrange pixel values (finite floating-point values < 0 or > 1) are permitted in either rendition.

Color Profiles

Gain maps represent the log-ratio of two images and therefore do not have intrinsic color spaces. Any colorimetry or color profile tag (e.g., ICC profile, colr/nclx box, CICP coefficients, etc.) associated with the gain map itself is ignored.

Orientation

The gain map must have the same orientation as the base image. Any orientation tag associated with the gain map itself is ignored.

Codec & Container Notes

The Gain Map method described above is largely independent of the choice of image compression method and image file container. In principle, it can work with (for instance) JPEG, PNG, HEIC, TIFF, AVIF, and so on. See this section for details on how Gain Maps can be stored in some existing file formats.

Gain Maps can be uncompressed, lossless compressed, or lossy compressed. When this method is used with lossy compression methods, extra care must be taken for both the Base and Gain Map images to ensure that the Derived rendition looks acceptable.

Discussion of Goals

This section reviews the goals stated at the beginning of the document and briefly evaluates the proposal against each item.

Take advantage of HDR display technology

The proposal inherently supports the display of HDR photos.

If the Base rendition is SDR, the derived HDR rendition (after applying the gain map) will be approximate; the quality will depend on the properties of the Gain Map (resolution, number of color planes, compression, codec, …) and Base rendition.

If the Base rendition is HDR, the HDR rendition itself will be exact (assuming a suitable display and viewing environment), and the SDR rendition will be approximate.

Adapt dynamically to the current display capabilities & viewing conditions

The Gain Map supports dynamic scaling (between [0,1]) to interpolate between SDR and HDR renditions. This ability, coupled with the metadata, enables the app to make runtime decisions based on the current display conditions.

Avoid keeping track of separate SDR and HDR renditions (two or more separate files)

The Gain Map can be stored as an optional tag in an image file, thereby avoiding having to track multiple files.

Keep the photographer in control of the HDR-to-SDR tone mapping process

Apps can optionally provide authors with the ability to produce both HDR and SDR renditions, leaving the author mostly in control of the output.

There is much flexibility here. Some options:

- An app can automatically generate both HDR and SDR renditions and derive the Gain Map from the two. This is a fully automatic workflow. Example: import a photo, press the Auto button, then Export.
- An app can let the user author a HDR rendition, then automatically compute a SDR rendition and derive a Gain Map from the two. Example: edit a photo in HDR mode then Export.
- An app can let the user author both SDR and HDR renditions, then derive the Gain Map from the two. Example: edit a photo in HDR mode, preview the photo in SDR mode, make additional adjustments to ensure the SDR rendition looks good, then Export. This is an “advanced” workflow that enables photographers to fine-tune image appearance across a range of devices.

Support local adaptation in tone mapping

The Gain Map is a 2D spatial map and therefore supports local adaptation. It may include, but is not limited to, a 1D curve. Local adaptation is critical when tone mapping HDR content for SDR displays.

Provide (some degree of) consistency and predictability

The proposal handles this by defining the space of possible renditions as a linear interpolation (in log space) between the SDR and HDR renditions. Therefore, if the authoring app provides a way to preview both SDR and HDR renditions (e.g., soft proofing) then the author should have a reasonable sense of how the image might look on a different system.

Provide backward compatibility for legacy software

If the Base rendition is SDR, then legacy apps will ignore the Gain Map tag and just display the SDR rendition. Similarly, older devices with SDR displays will just display the SDR rendition.

Minimize image file storage overhead

The Gain Map is effectively an additional image that is stored in the file as an optional tag. Authoring apps can trade off quality vs size by adjusting the Gain Map’s resolution, number of planes (RGB vs grayscale), and compression.

Minimize complexity of additional runtime display logic

Applying the Gain Map to the Base rendition at display time involves a per-plane (1D) affine transform, image resampling, an 2^x exponential, and multiplication. This is just a few lines of shader code.

Pros and Cons

The advantage of the proposal is that it meets all of the goals, as discussed in the previous section. In particular, if used with an existing file format such as JPEG, HEIC, or PNG, then different apps and platforms (both within Adobe and outside of Adobe) can adopt the proposal at different times without fundamentally breaking compatibility.

The main disadvantages of the proposal are the increase in storage and the additional runtime display logic. However, the proposal (Base rendition + Gain Map) should still be smaller in size than two separate renditions (SDR + HDR), and the additional runtime logic should be relatively fast and straightforward to implement on most GPUs.

Comparison to Other Methods

Hybrid Log-Gamma

Hybrid Log-Gamma (HLG) is a standard and transfer curve for representing HDR data, typically in an integer coding (most commonly 10-bit or 12-bit). Due to the construction of the curve, it provides some measure of built-in tone mapping control for both HDR and SDR displays.

Disadvantages (vs the proposed method):

- HLG's tone mapping is limited to a curve and therefore SDR renditions may be suboptimal, especially compared to local adaptation techniques.
- If the base image is 10-bit or 12-bit HLG, it will not be compatible with older software.
- HLG is limited to a maximum HDR value of 12.0 (+3.6 stops of highlight headroom).

Direct HDR Formats (10-bit HEIC, AVIF, JPEG XL, …)

Using new file formats and codecs (instead of JPEG, PNG, etc.) and higher bit-depth encoding provides the highest quality at the smallest file sizes. However, this direction has two major disadvantages:

- System- and app-dependent behavior when viewing HDR content on different devices across viewing conditions
- Incompatible with older software

Note that the Gain Map proposal is actually complementary to HDR-compatible file formats. For example, it is reasonable in the long term to have the Base rendition be a 10-bit AV1-compressed file with the Perceptual Quantizer or HLG curves, with an optional Gain Map to provide a HDR-to-SDR tone mapping. This approach would give up compatibility for older readers (since the Base image uses a new format) and potentially inconsistent display of HDR content for readers that ignore the Gain Map.

Recommendations and Intended Usage

The intended usage of the proposal is for HDR & SDR renditions in end-user viewing and distribution applications – in other words, any place a JPEG might normally be used. Examples include photographs viewed on social media, a web page, slideshow app, and so on.

It is **not** recommended to use this proposal for intermediate storage in an editing workflow. Derived HDR renditions are approximate because of gain map resolution, compression, and other factors. Therefore, repeated round-trip editing may lead to gradual data loss, similar to re-editing JPEG or other lossy compressed files multiple times.

For capture and editing workflows involving HDR content, we recommend using a high-fidelity image representation, such as 16-bit or 32-bit floating-point in a linear encoding (where 1.0 represents SDR white) and lossless compression.

Example Use Case #1:

The Base rendition is SDR and is stored as an 8-bit JPEG in the Display P3 color space. The Gain Map and associated metadata are stored as an optional tag in the JPEG. The Gain Map is also stored as an 8-bit JPEG compressed image; it can be either color or grayscale, and it can be stored at a lower resolution.

- Older image readers will ignore the Gain Map tag and just display the SDR rendition.
- Newer image readers on SDR displays can read the Gain Map tag but choose to ignore it and just display the SDR rendition.
- Newer image readers on HDR displays can read the Gain Map tag and choose how much of it to apply at display time.
- Since the Base rendition is stored using only 8 bits per component and is JPEG compressed, the reconstructed HDR rendition (using the Gain Map) may exhibit banding and other color artifacts in some images.

This use case is intended for the “short term” because JPEG is widely supported and new HDR still image file formats are not yet widely supported. This solution provides excellent backward compatibility at the expense of some image quality.

Example Use Case #2:

The Base rendition is SDR and is stored as lossy compressed 10-bit JPEG XL in the Rec 2020 color space. The Gain Map and associated metadata are stored as an optional tag. The Gain Map is also stored as a lossy compressed 10-bit JPEG XL image; it can be either color or grayscale, and it can be stored at a lower resolution.

- Legacy image readers that do not support JPEG XL will not be able to read the image.
- Newer image readers that support JPEG XL but do not support the Gain Map will ignore the Gain Map tag and just display the SDR rendition.
- Newer image readers that support both JPEG XL and the Gain Map can read the Gain Map and choose how much of it to apply at display time.

- The higher bit depth and wider color space should provide higher visual quality than Example Use Case #1 for some images.
- This example is also applicable to other codecs and formats (e.g., H.265/HEIF, AV1/AVIF instead of JPEG XL) and other higher bit depths (e.g., 12-bit instead of 10-bit).

This use case is intended for the "longer term" once new HDR still image file formats become more widely supported. This solution provides better image quality at the expense of backward compatibility.

Example Use Case #3:

The Base rendition is HDR and is stored as lossy compressed 10-bit JPEG XL in the Rec 2020 color space with the Hybrid Log-Gamma transfer curve. The Gain Map and associated metadata are stored as an optional tag. The Gain Map is also stored as a lossy compressed 10-bit JPEG XL image; it can be either color or grayscale, and it can be stored at a lower resolution.

- Legacy image readers that do not support JPEG XL will not be able to read the image.
- Newer image readers that support JPEG XL but do not support the Gain Map will ignore the Gain Map tag and just display the HDR rendition. The tone mapping of the HDR rendition (if any) may be app- and system-dependent.
- Newer image readers that support both JPEG XL and the Gain Map can read the Gain Map and choose how much of it to apply at display time.
- Since the Base rendition is HDR, the Gain Map is used to tone map the image (i.e., produce the SDR rendition). This is the opposite of Examples #1 and #2.
- This example is also applicable to other codecs and formats (e.g., H.265/HEIF, AV1/AVIF instead of JPEG XL) and other higher bit depths (e.g., 12-bit instead of 10-bit).
- This example is also applicable to other transfer curves for the Base HDR rendition, such as Perceptual Quantizer.

This use case is similar to Example #2 except that the Base rendition is HDR instead of SDR. A downside of this approach (compared to Example #2) is that different apps and platforms may treat the Base HDR rendition very differently on SDR displays or HDR displays that do not have sufficient headroom to display the HDR content. Still, this may be a good option if tone map consistency is not important.

Example Sizes

The following numbers are based on a preliminary study of 50 raw images from digital cameras of natural content (landscape photographs) converted to 2000-pixel (long edge) renditions, RGB, Display P3 color space. The numbers represent the average size (per photo).

- 8-bit uncompressed SDR rendition: 12842 KB
- 8-bit JPEG compressed SDR rendition (Photoshop quality 90): 1747 KB
- 16-bit JXL compressed SDR rendition (cjxl -d 1): 662 KB
- 16-bit JXL compressed Gain Map, ¼ resolution, RGB quality 1 (cjxl -d 1): 101 KB (15% of SDR JXL)
- 16-bit JXL compressed Gain Map, ¼ resolution, RGB quality 2 (cjxl -d 2): 60 KB (9% of SDR JXL)

- 16-bit JXL compressed Gain Map, ¼ resolution, RGB quality 3 (cjxl -d 3): 43 KB (7% of SDR JXL)

How to Store Gain Maps in Various File Formats

Earlier drafts of this document described methods to store Gain Map images and metadata in various file formats including JPEG, JPEG XL, HEIF/AVIF, TIFF, and DNG.

Gain Maps are currently undergoing standardization in ISO/TC 42 Photography. Furthermore, various technical committees have ongoing efforts to standardize Gain Map storage in different file formats.

Consequently, the original Gain Map storage proposals have been removed from this document to avoid confusion and conflicting information.

Notes on Perceptual Quantizer (PQ) and Hybrid Log-Gamma (HLG)

The PQ and HLG transfer curves are commonly used to represent encoded HDR signals, especially in image formats with integer pixel types (see ISO 22028-5 and ITU-R BT.2100). This section describes how to combine PQ/HLG-encoded Base HDR image data with gain maps for portable tone mapping.

When the Base image is encoded using PQ or HLG, the image must first be transformed to a linear (gamma 1.0) display-referred signal before the gain map is applied.

For PQ:

1. Apply the EOTF to map the non-linear PQ signal to display-referred light (0 to 10000 nits).
2. Divide by the SDR white level (default: 203 nits).

For HLG:

1. Apply the Inverse OETF.
2. Apply the OOTF with peak luminance $L_w = 1000$ nits and system gamma $\gamma = 1.2$.
3. Divide by the SDR white level (default: 203 nits).

In both cases, the result is a normalized display-referred signal where 1.0 means SDR white. Per ISO 22028-5, the default SDR white level is 203 nits, but a custom SDR white level may be specified via metadata.

The default SDR white level of 203 nits means that PQ- and HLG-encoded images have maximum HDR capacities of 5.6 and 2.3 stops, respectively:

- PQ: $log_2(10000/203) = log_2(49.26) = 5.6$ stops
- HLG: $log_2(1000/203) = log_2(4.926) = 2.3$ stops

History

- Draft 1, 2021-Nov-23

- Draft 2, 2021-Dec-10
    - Add constants k1 and k2 to avoid divides by zero
    - Add pseudocode for calculating and applying gain map
    - Add section on bit depth for renditions
    - Add section with three example use cases
- Draft 3, 2022-Jan-23
    - Add gamma parameter for better use of gain map encoding space
    - Clarify that gain map can be stored in different pixel formats
    - Add example size data
- Draft 4, 2022-Mar-8
    - Add section on XMP metadata representation
    - Add section on embedding Gain Maps in JPEG and JPEG XL files
    - Add details for baseline interpolation method
    - Fix bugs in CalculateGainMap and ApplyGainMap pseudocode
- Draft 5, 2022-Dec-20
    - Add section on HEIF and AVIF
    - Add version tag
    - Several minor language revisions
    - Rename k1 and k2 constants to k_{hdr} and k_{sdr} for clarity
- Draft 6, 2023-Jan-13
    - Revise JPEG proposal to use 2-byte part index and count
- Draft 7, 2023-Jan-20
    - Add section Color Space and Gamut Notes
    - Revise section on computing the gain map to clarify the role of the *k* parameters
- Draft 8, 2023-Feb-3
    - Clarify multi-valued field specification in XMP
    - Clarify placement of gain map metadata in primary vs auxiliary image XMP
    - Revise JPEG section with alternate embedding method (Multi-Picture Format)
- Draft 9, 2023-Feb-11
    - Numerous minor language revisions
- Draft 10, 2023-April-3
    - Clarify floating-point to/from integer conversion
    - Clarify orientation & color profile tags with respect to gain maps
    - Clarify meaning of “default” and “count” in XMP metadata table
    - Clarify XMP metadata data types (Text and Real instead of String and Float32)
    - Revise multi-element fields to use standard XMP sequential list
    - Add inequality requirements for various metadata fields
- Draft 11, 2023-April-27
    - Remove app markers proposal from JPEG section (recommend MPF only)
    - Clarify XMP metadata example with XML <rdf:Description /> syntax
- Draft 12, 2023-May-2

    - Add section on Perceptual Quantizer and Hybrid Log-Gamma
- Draft 13, 2023-June-8
    - Make Version, GainMapMax, and HDRCapacityMax required tags.
    - Explain behavior when required tags are missing/invalid (ignore gain map).
    - Add proposal for storing Gain Maps in TIFF and DNG.
- Draft 14, 2023-October-7
    - Fix section "How to Apply the Gain Map" so that resampling step happens first. This matches the GPU shader pseudocode in the following section.
    - Update section on HEIF and AVIF with more recent proposal for storing Gain Maps using an alternate image group.
- Draft 15, 2024-February-28
    - Add information about ongoing standardization efforts
    - Remove concrete proposals describing how to store Gain Maps in various file formats