A Proposal for OpenEXR Color Management

Florian Kainz, Industrial Light & Magic

Revision 5, 08/05/2004

Abstract

We propose a practical color management scheme for the OpenEXR image file format as used in a film and visual effects production environment. Our scheme produces predictable results when images from film, video and other sources are converted to the OpenEXR format, processed digitally, and then output onto film or video. Predictable results are possible even when activities like film scanning, image processing and film recording are performed at different companies. The scheme also allows accurate previewing, so that artists working with OpenEXR files can see on their computer monitor what their images will look like when they are presented in a theater. A variation of the proposed color management scheme has been used in production at Industrial Light & Magic for several years, demonstrating that the scheme is conceptually sound, efficient, and is not a burden on artists.

Scene-Referred Images

OpenEXR images are usually *scene-referred*: The floating-point red, green and blue values in an image are proportional to the relative amount of light coming from the corresponding objects in the depicted scene. The scene may exist in reality, it may be hypothetical as is the case for 3D computer renderings, or it may be a composite of real and hypothetical elements.

In scene-referred images, the data stored in the pixels should not depend much on how the images were acquired. Ideally, if a scene is photographed twice, once on motion picture film, and once with a digital camera, then the corresponding scene-referred image files should contain the same data. In the real world, we typically don't have enough information to reconstruct absolute scene luminance values from a scanned film negative or from raw digital camera output. We don't know the cameras' exposure settings, or whether color filters were used. We can only recover relative amounts of light. For example, we can tell that one object emits twice as much "blue" light as another, but we cannot assign absolute values. However, relative values are usually good enough; we can make our two images look the same by color timing, that is, by scaling the values in the red, green and blue channels.

Of course there are limits to how closely two scene-referred images captured with different types of cameras will match. The cameras usually have a different dynamic range and color gamut. Pixels outside the range or gamut of one of the cameras will not match. Digital cameras sometimes perform local color and contrast optimization such that the data stored in a particular pixel depend on the colors of nearby pixels. If an image has been processed like this, recovering a scene-referred representation is difficult or impossible. To generate a scene-referred image, we should start with raw image sensor data, or as close to linear-light image data as the camera can provide.

Cameras might also differ in their spectral sensitivities, that is, in the way they reduce the relative proportions of all the wavelengths humans can see to just the relative proportions of what we call red, green, and blue. In practice, however, camera and media combinations are all engineered to approximate the acquisition response of the human visual system. Consequently, scene-referred images of real-world objects under real-world lighting tend to be fairly similar, even if they were recorded with different cameras.

Representing images in scene-referred form is good for image processing. Assembling a composite image from multiple input images is relatively easy because all input images live in the same color space, no matter whether they were originally captured on film or video, or whether they are computer-generated. Also, implementing filters, compositing operations, or anti-aliased drawing routines is easy since no non-linear light encoding has to be taken into account.

Representing images in scene-referred form is bad for viewing the images. An image file contains an approximation of what someone observing the actual scene would see; the pixel data do not specify what the image should look like when it is presented to an audience. For example, photographing a scene on motion-picture negative film, striking a print, and projecting this print in a theater imparts a certain look to the recorded image. What the audience sees in the theater is not a faithful reproduction of the original scene. Typically, the projected image has a higher contrast, and colors are more saturated than in the original scene. The dynamic range and the color gamut of the projected image are

limited. Extreme luminance values and very saturated colors in the original scene cannot be reproduced, and so the overall contrast and saturation of the projected image are enhanced to compensate. The look of the projected image is in part an artistic decision; it can be changed by choosing specific negative and print film stocks, and by modifying how the film is processed (for example by employing the bleach bypass method).

Making an image scene-referred removes the "film look" (or video look) from the image. The pixel data tell you what was in the original scene, not what would be projected on the screen. The film look must be put back into the image before it is shown in a theater. If the image is previewed on a video monitor, an approximation of the film look must be applied to the pixel data, so that the image's appearance on the monitor and in the theater matches within the limits of the monitor's color gamut and dynamic range.

OpenEXR Color

The following proposes a color management scheme for OpenEXR images, which can be summarized as follows:

- RGB data in OpenEXR files are always scene-referred.
- Converting an image from another medium to OpenEXR RGB requires reconstruction of scene luminance values.
- Viewing an image requires applying a transformation to the pixel data that produces the image's intended look.

Note that non-RGB pixel data in OpenEXR files are not necessarily scene-referred. Because OpenEXR supports arbitrary image channels, a film scanner could store raw printing density data in the file. The density values would not be considered scene-referred. (Storing density values would be useful for digital archiving of a film negative.)

The goal of OpenEXR color management as described here is to provide an environment that makes digital image processing easy and allows compositing of images from different sources.

Process and Terminology

Images of real scenes are captured on *input media*, for example film or video.

Converting the images to OpenEXR makes the pixel data scene-referred; the values stored in the pixels are proportional to the luminance of the corresponding objects in the original scenes. The conversion process that reconstructs relative scene luminance values from the data stored on an input medium is called the *input medium to scene-referred transform* or IMSR.

Image processing takes place in the scene-referred color space.

After all image processing has been performed, OpenEXR images are not typically delivered directly to a paying client. Instead, the images are recorded on an *output medium*, usually film or video. Recording converts the scene-referred pixel data to the output medium's native representation. The conversion process is called the *scene-referred to output medium transform* or SROM.

When the output medium is viewed on a *reference display*, that is, on a display that is typical for the output medium, the recorded pixel data are converted to luminance values on the display surface. This conversion is called the *output medium to reference display transform* or OMRD.

It is often necessary to *preview* an OpenEXR image, that is, show an image on a *preview display* other than the reference display, while making the image look the same as on the reference display. For example, images ultimately intended for film projection are often previewed on a computer monitor. For any given preview display / reference display pair, there is a mapping from reference display pixel values to preview display pixel values such that colors on the preview display appear to be (more or less) the same as the corresponding colors on the reference display. This mapping is called the *reference display to preview display transform* or RDPD. To preview an OpenEXR image, the SROM, OMRD and RDPD must be applied to the pixel data.

Examples

Scenario 1 - Producing Film

We are working on a scene that is a composite of several elements that are shot separately. Our deliverable is a film negative (see Figure 1).

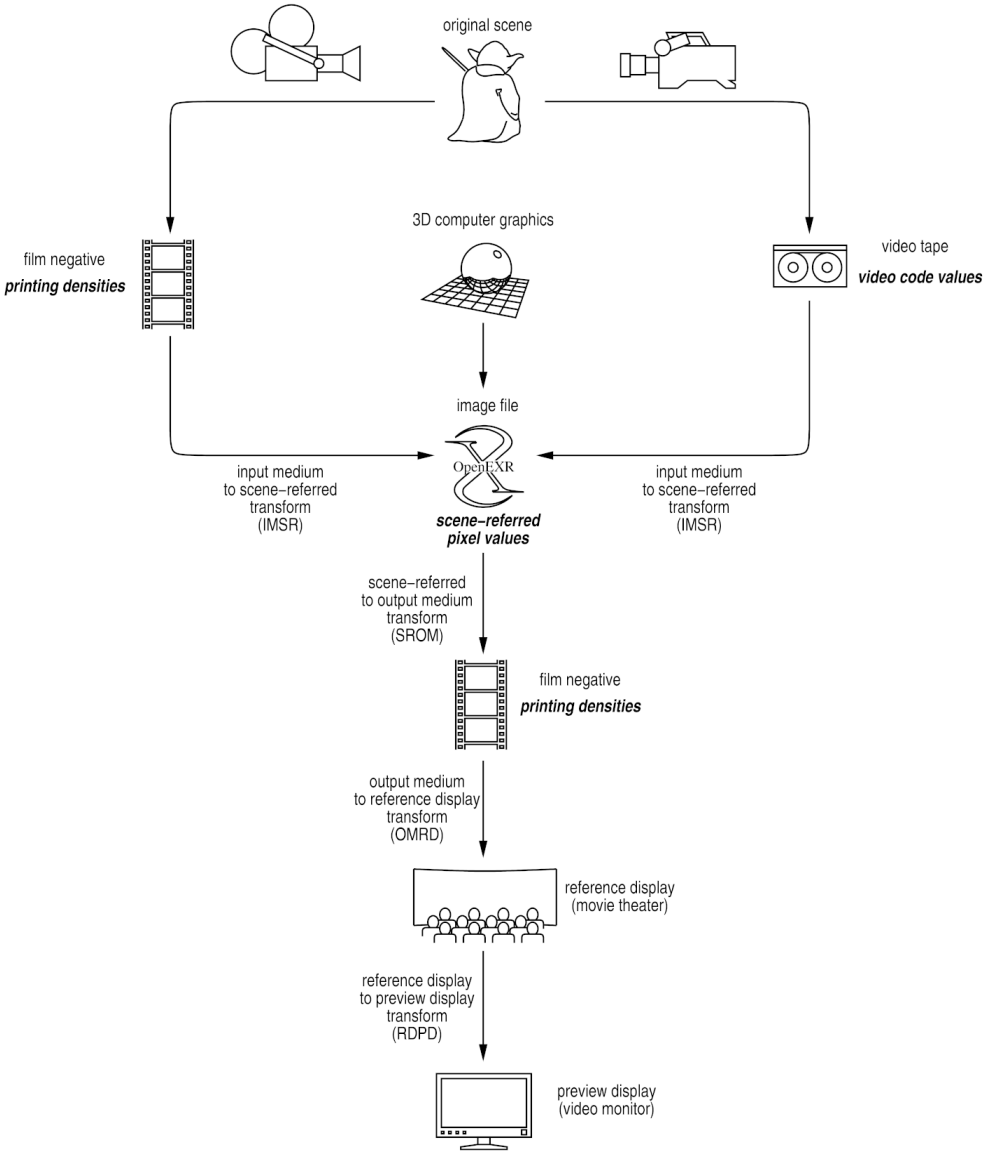

Figure1: producing film

The cameras used to shoot the elements may include film cameras loaded with one or more types of camera original film stock, video cameras, and digital still cameras. The film negatives, video tapes and raw camera image files are our input media.

The images recorded on the input media are converted into OpenEXR files. Negative density values, video code values and raw digital camera data are converted into scene-referred red, green and blue pixel data by applying an

input medium to scene-referred transform, or IMSR. The IMSR is specific to each input medium; potentially there is a separate IMSR for each film stock, video system, and digital camera. For film, the IMSR may have to be specific to the combination of film stock and film scanner. Some scanners measure status M density, some measure SMPTE RP 180 printing density, others may use their own unique definition of density. For digital still cameras, the IMSR may vary from manufacturer to manufacturer.

As much as possible, the IMSR for a specific input medium tries to recover relative luminance values in the original scene, as seen by the camera. The RGB pixel values are proportional to the amount of red, green, and blue light received by the camera. Usually, the IMSR cannot recover absolute luminance values. However, if pixel values are proportional to luminance, images converted from two different input media can be made to match by simple color timing. After color timing, two images of the same scene, recorded by two different cameras, for example, film and video, should contain the same pixel data.

Any 3D computer-graphics elements we may need for our scene are rendered directly into scene-referred OpenEXR files. Internally, most 3D renderers already work in a scene-referred color space, and writing scene-referred image files is particularly easy to do.

Image processing operations, for example, applying filters, compositing, or blue-screen extraction, are performed in the scene-referred color space. The resulting final images are also scene-referred.

The final images are recorded on intermediate stock; this is our output medium. A scene-referred to output medium transform, or SROM, converts RGB pixel values back to film densities. The SROM affects what the images will look like in the theater. If we want to make the final images look as if they had been photographed directly on one of the camera stocks, then we use an SROM that simulates that stock. The SROM is essentially the inverse of the IMSR for the camera original stock, but it also depends on the film recorder's density definition (status M, RP 180, etc.).

The images recorded on the intermediate film are usually shown to an audience by making a print from the negative and projecting the print in a movie theater. The movie theater is the reference display for our images. Making a print and projecting it converts the negative's density values to luminance values on the theater screen. We call this conversion the output medium to reference display transform, or OMRD. The OMRD depends on the print stock used, how the print film is processed (for example, with or without bleach bypass), and to some extent also on the specific theater that is chosen as the reference display.

Usually artists want to preview their final images on a video monitor, before the images are recorded on film. In order to give an artist a good idea of what the audience will see, previewing software must predict the luminance values on the theater screen by concatenating the SROM with a model of the OMRD. The preview software must then convert the theater screen luminance values to video monitor RGB values such that the appearance of the image on the monitor matches the appearance on the theater screen. This conversion is the reference display to preview display transform, or RDPD, and the video monitor is the preview display. The RDPD depends on the characteristics of the reference display and on the preview display. Usually the RDPD does not attempt to reproduce exactly the same luminance values as those on the reference display. Typical video monitors are smaller but brighter than theater screens, and video monitors are often viewed in an office environment rather than in a dark theater. Those factors significantly alter how colors appear to a viewer, and a really good RDPD must take color appearance into account.

Scenario 2 - Producing Video

Most of the workflow for producing images on film also applies if the final images are meant to be recorded on video (see Figure 2). As before, elements are recorded on various kinds of input media, and the elements are converted to scene-referred OpenEXR files, using an appropriate IMSR for each input medium. Image processing is performed in scene-referred color space, producing scene-referred final images.

The final images are recorded on video tape, using an SROM that is specific to the video system used.

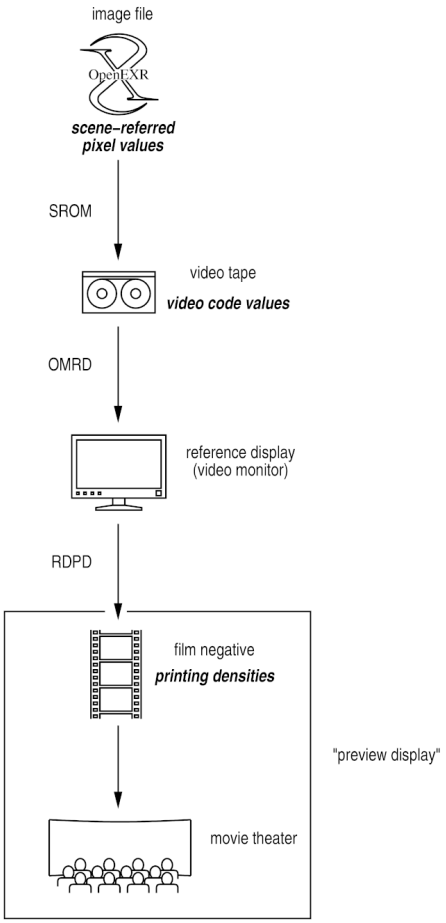

Figure 2: producing video

The images are displayed on a video monitor by playing the tape. For video, the monitor is the reference display. The OMRD, which converts the data on the video tape to luminance values on the monitor, is built into the video hardware.

Since displaying images on a video monitor can be done quickly, and computer monitors can do a pretty good job simulating video playback, artists can check their images directly on the reference display or on a simulated reference display. Previewing on a separate preview display is not necessary.

However, if we want to transfer our video to film, we can treat the process that begins with recording images on film and ends with projecting a print in the theater as a kind of preview display. In this case the RDPD converts video monitor luminance values to film density values in such a way that the appearance of colors on the theater screen matches the appearance on the monitor. In order to "preview" the final scene-referred images, the recording software converts scene-referred RGB values to film densities by concatenating the SROM, OMRD and RDPD.

Scenario 3 - Painted Images

Sometimes we want to combine painted images with photographic elements. In order to fit into our model, painted OpenEXR images must be in a scene-referred state, where pixel values are proportional to luminance values in the hypothetical scene that was painted. For images created in the computer, using paint software, this can be achieved in at least two ways:

Software that is aware of OpenEXR color management loads an appropriate SROM, OMRD and RDPD (we assume that we are producing film). Painting modifies the scene-referred image. The paint software updates the display on the computer monitor by applying the appropriate color transforms to the scene-referred data. This way, the artist looks at

the computer monitor and sees what the painting will look like on the reference display, while he or she paints a scene-referred image.

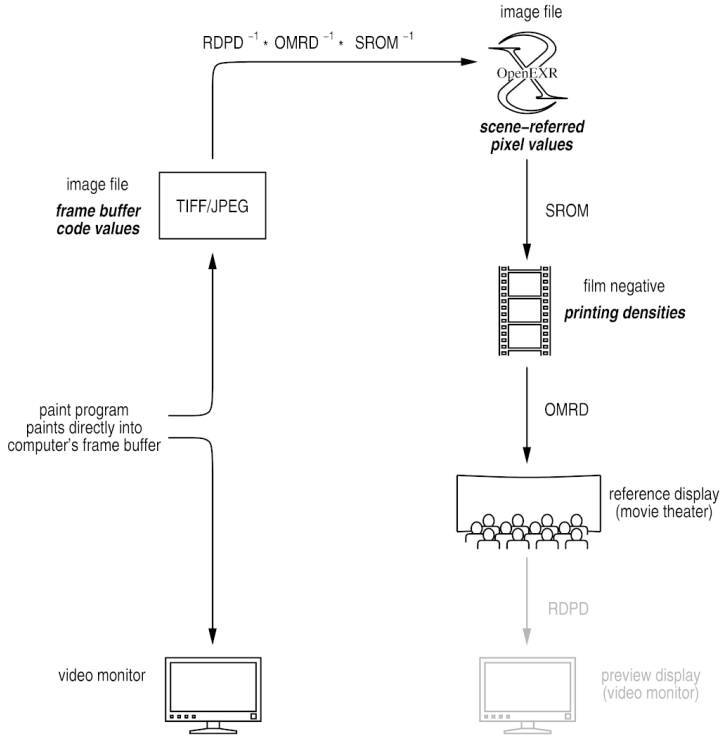

Figure 3: painting TIFF or JPEG images

With "traditional" paint software that is designed to produce TIFF or JPEG files, the artist paints directly into the computer's frame buffer. When the painted image is saved, the pixel values in the file are the same as the values in the frame buffer. In this case, the artist has painted a preview image. This preview image can be converted into a scene-referred OpenEXR file by applying the inverse of the RDPD, OMRD and SROM. (In most cases, it is not desirable to try and generate the inverse transformations automatically. Because of dynamic range and color gamut limitations, the forward transformations typically do not have an exact mathematical inverse, but it is usually possible to create good approximate inverse transformations. Ideally, the author of a color transform also supplies an inverse transform. An IMSR and the corresponding SROM are usually inverses of each other.)

Scanned paper images, for example, photographs from a book or magazine, can be handled in a similar way as painting with traditional paint software: the scanned image is loaded into the paint program, and the colors are adjusted until they look right on the preview display. Applying the inverses of the RDPD, OMRD and SROM generates an image that is approximately scene-referred.

Implementation of Color Transforms

To make the color management scheme outlined above work, we must be able to describe color transforms in such a way that a program can automatically apply the transforms to pixel data. To be sufficiently general, we describe color transforms as functions written in a *color transformation language*, CTL. CTL is a simple interpreted programming language with a C-like syntax. In order to make CTL programs run sufficiently fast, execution is vectorized; the CTL interpreter operates on many pixels at the same time. Input pixels and image metadata (for example, OpenEXR attributes) are passed to a color transformation as function arguments; output pixels and image metadata are returned the same way.

Here is an example that shows what a CTL program might look like: a function that converts from Rec. ITU-R BT.709 component video to scene-referred OpenEXR pixel values. For simplicity, the code assumes that the video camera

implements the video gamma curve exactly, without any dynamic range compression. The code also assumes that the video signal has been normalized to a zero-to-one range:

```
float
gammaToLinear (float x)
{
   if (x <= 0.08125)
       return x / 4.5;
   else
       return pow ((x + 0.099) / 1.099, 1 / 0.45);
}

void
transform_Rec709Video_RGB (varying float Y,
                           varying float Cb,
                           varying float Cr,
                           output varying float R,
                           output varying float G,
                           output varying float B,
                           output uniform float chromaticities[8])
{
   float R1 = (1.5748 * Cr) + Y;
   float B1 = (1.8556 * Cb) + Y;
   float G1 = (Y - 0.0722 * B1 - 0.2126 * R1) / 0.7152;

   R = gammaToLinear (R1);
   G = gammaToLinear (G1);
   B = gammaToLinear (B1);

   chromaticities[0] = 0.640;  // red
   chromaticities[1] = 0.330;
   chromaticities[2] = 0.300;  // green
   chromaticities[3] = 0.600;
   chromaticities[4] = 0.150;  // blue
   chromaticities[5] = 0.060;
   chromaticities[6] = 0.3127; // white point
   chromaticities[7] = 0.3290;
}
```

Function `transform_Rec709Video_RGB()` has three input parameters, `Y`, `Cb` and `Cr`. The parameters are declared `varying`, that is, their value changes from pixel to pixel. In other words, `Y`, `Cb` and `Cr` are the video image channels.

`R`, `G` and `B` are output parameters that correspond to the red, green, and blue channels of the OpenEXR image. Like the input parameters, `R`, `G` and `B` are declared `varying`.

The last output parameter, `chromaticities`, has been declared `uniform` to indicate that its value does not vary from pixel to pixel. `Chromaticities` is an OpenEXR header attribute; its value specifies the CIE (x,y) chromaticities of the red, green, and blue primaries and the image's white point.

The body of `transform_Rec709Video_RGB()` is fairly straightforward. First `Y`, `Cb`, and `Cr` are converted to gamma-encoded red, green and blue signals. Then function `gammaToLinear()` is called to make the red, green and blue values proportional to scene luminance. Finally, the chromaticities attribute is set so that the red, green, blue and white point chromaticities match the Rec. 709 specification.

The color transform in this example is easy to specify using only arithmetic expressions. Other transforms may be more conveniently expressed with the help of lookup tables. CTL provides built-in functions for interpolated lookups in one, two and three-dimensional tables.

CTL programs are stored in text files that are separate from OpenEXR and other image files. The programs are executed by a CTL interpreter. When asked to load a specific CTL program, the interpreter reads the corresponding text file and translates the program into an executable form. The CTL program can then be invoked from C++.

Attributes in OpenEXR Files

OpenEXR files refer to color spaces, via four new header attributes of type `StringAttribute`:

`inputMedium`	Symbolic name of the input medium from where the OpenEXR image was imported, for example"Kodak5212", or "Rec709Video". This attribute is not present in images that were generated directly in scene-referred form, for example 3D computer graphics.
`sceneReferredSpace`	Symbolic name of the scene-referred color space used by the OpenEXR image file. This should normally be set to "RGB", and if the attribute is not present, "RGB" should be assumed. In some cases, scene-referred color spaces other than RGB, for example CIE XYZ, may be useful.
`outputMedium`	Symbolic name of the output medium for which the image is being produced, for example "Kodak5242" or "Rec709Video".
`referenceDisplay`	Symbolic name of the reference display for this image, for example "Kodak2393IlmCTheater" to indicate a print made on a particular film stock, projected in a specific theater.

One could also define a `previewDisplay` attribute, but storing the name of the preview display in the image file is not very useful; it should be possible to preview the image on an arbitrary display, provided an appropriate CTL function for the corresponding RDPD exists.

The name of the CTL function that should be invoked to transform an image from one color space to another can be derived from the values of the corresponding header attributes:

transform	CTL function name
IMSR	`transform_`*`inputMedium`*`_sceneReferredSpace`
inverse of IMSR	`transform_`*`sceneReferredSpace_inputMedium`
SROM	`transform_`*`sceneReferredSpace_outputMedium`
inverse of SROM	`transform_`*`outputMedium`*`_sceneReferredSpace`
OMRD	`transform_`*`outputMedium_referenceDisplay`
inverse of OMRD	`transform_`*`referenceDisplay_outputMedium`
RDPD	`transform_`*`referenceDisplay_previewDisplay`
inverse of RDPD	`transform_`*`previewDisplay_referenceDisplay`

Real or Vaporware?

A version of the color management scheme described above has been employed successfully by Industrial Light & Magic for several years, although ILM's current terminology and how the scheme is currently implemented in ILM's software differ from this description.

The fact that the scheme works in practice indicates that the overall concept is sound. From an artist's point of view, the scheme is easy to use. The color transformations are mostly hidden from artists. The IMSR takes place during film scanning or during conversion from video to OpenEXR. When an artist first sees the image it is already scene-

referred. Similarly, the SROM is part of the film or video recording process. When previewing an image, artists can choose to see the scene-referred image, or to preview the image as it would appear on the reference display, simply by enabling or disabling "film look".

We are formalizing our color management scheme in order to make it possible to use the scheme outside ILM. Standardizing how color transformations are described should allow companies to exchange the descriptions when necessary. An interpreter for CTL and an API to access the interpreter from C++ programs are currently under development. We intend to release both as open source. We also intend to publish useful IMSR, SROM, OMRD and RDPD examples, and we will make our sample image viewing program, exrdisplay, aware of CTL.

Finally, it should be noted that although the description of our color management scheme implies that images are stored as OpenEXR files, our method would also work with other high-dynamic-range image file formats, provided the files can store metadata equivalent to the `inputMedium`, `sceneReferredSpace`, `outputMedium` and `referenceDisplay` attributes.