Understanding EXR Data Compression

Information compiled by Nicolas Dufresne for Rainbox Productions and Rainbox Laboratory. Contact us if you see any mistake or have suggestions for this document. https://rainboxlab.org
Many thanks to Dino Muhić for his expertise about DWA.

When using the OpenEXR format, it may be hard to determine what is the best data compression to use, and the question is raised very often. This document is an attempt to answer it, depending on what is your need with the file and what type of image you're storing.

A compression is said to be the best for a specific need when its read or write speeds better fit the application, and the file size is the smallest available.

Types of images

In this document, we're distinguishing between the following different types of images:

Grainy video or animation
Photographic images (including realistic CGI) or animation with grain

Video
Photographic images (including realistic CGI) without grain

Animation, Graphics
Highly stylized images such as 2D Animation, Motion Graphics, stylized 3D Animation, or 3D passes such as *Z-Depth*, *Normal*...

Solid colors, large flat areas
Images mostly consisting of solid colors, such as alpha or id channels

Texture maps
Multi-Resolution files

Summary

Lossy (final) exports

Especially for final exports, you probably don't need lossless compression, and with the right settings, the exported file size can be quite small with just a very small reduction of quality, and the *Master* or backup of your movie can be stored this way.

In this case, **DWAA** (with a small compression level) is very efficient in all cases. Be careful, **DWA is not supported by ffmpeg (yet)**. In this case **PXR24** is a good (lossless) alternative.
Solid colors (e.g. alpha channels) can be compressed very efficiently using **RLE** without losing quality.

If you're not exporting for complex compositing (e.g. choma keying), and especially from a video source or if the export is to be used as a *Master* for further *YUV 422* or *421* video exports (like h.264), the **Luminance/Chroma** option will divide the file size by two with only a very small reduction of quality. Be careful, **ffmpeg does not support the Luminance/chroma option (yet).**

Lossless (intermediary) exports

If the file is to be used in a compositing software for example, you may want to export without losing quality.

In any case, if your rendering an image with AOV (most likely from a 3D software), and you can accept a (very small) quality loss, **DWA** is the best option, as it will compress only the RGB channels (or Y, RY, BY in case of Luminance/Chroma) and use RLE for alpha and ZIP for any other channel.
In this case, be careful that DWA uses the channel names (case sensitive):
- Lossy channels: R, G, B, Y, RY, BY
- RLE: A
- ZIP: any other name (Red, red, r, Green, green, g, Blue, blue, b, x, y, z, U, u, V, v, etc…)

Be careful with the names of the channel. For example using XYZ will result in the Y channel being lossy. You can use "xyz" instead.

When the **images don't have grain**:
If you don't need full 32-bit float precision, **PXR24** is the best compression you can use.
If you need full 32-bit float precision, **ZIP** is the best option.

When the **images have grain**, **PIZ** is always the best option.

For **stereo** images, the best is **ZIP**.

For **solid colors** such as alpha channels, it's better to use **RLE.**

Special cases

There are more specific uses for OpenEXR:

On systems for **real-time playback**, **B44** is preferred (or B44A for alpha channels and solid colors).

If you need to render **small-size lossy proxies**, you can use **DWAA** with a high compression level.

Gray-scale images will benefit a lot from the **Luminance/Chroma** option, without losing quality.

Luminance/Chroma images

Encoding flat images with one luminance and two chroma channels, rather than as RGB data, allows a simple but effective form of lossy data compression that is independent of the compression methods listed here. The chroma channels can be stored at lower resolution than the luminance channel. This leads to significantly smallerfiles, with only a small reduction in image quality.

This is the same kind of method as what is called *YUV 422* for video.

That means you can use this option if you're exporting to or from a standard *YUV 422* or *421* video without losing quality.
If you're exporting to or from *YUV 444* or *RGB*, the reduction in quality is still very small.

As luminance is stored in full quality using the Luminance/Chroma option, it can store gray-scale image much better than standard RGB without losing any data.

Note : this option is not supported by ffmpeg (yet).

Formats you don't know you already know

ZIP is the compression method used in **Portable Network Graphics (.png)** files. It is **NOT** based on the **Zip file format**, despite its name). That means when this document states you can use ZIP, it can be replaced by PNG files (if you don't need AOV, keeping in mind that most software is slower handling PNG, especially *After Effects*).

RLE is a compression used in **TGA** files, and it is close to the way data is compressed with **Quicktime Animation** sometimes used in **.mov** files. This means you can replace RLE by TGA files (if you don't need AOV)

DWAA is close to the way data is compressed in **JPEG** files. DWA could be replaced by a JPEG file sequence, if you're careful about the compression rate (and you don't need alpha or AOV).

Summary by usage and type of image

• Final render/export

Animation, Graphics or Video

DWA, (a bit) lossy (use a small compression level) – not supported by ffmpeg (yet).
PXR24, lossless*
If exporting to *YUV 422* or *421* video (e.g. h.264) or gray-scale, use the Luminance/Chroma option – not supported by ffmpeg (yet).

Grainy video or animation

DWA, a bit lossy – not supported by ffmpeg (yet).
PIZ, lossless
If exporting to *YUV 422* or *421* video (e.g. h.264) or gray-scale, use the Luminance/Chroma option – not supported by ffmpeg (yet).

Solid colors, large flat areas (alpha and id channels)

RLE

* In the specific case of a final export, PXR24 is considered lossless, as there should not be any need for 32-bit float data.

• Intermediary 32-bit float

Texture maps, Animation, Graphics or Video

ZIP
DWA if you can afford a small quality loss – not supported by ffmpeg (yet).
Note that DWA will only compress R,G,B channels (or Y,RY,BY in case of Luminance/Chorma) and automatically select either RLE for alpha or ZIP for AOV (Z, U, V, Normal...).

Grainy video or animation

PIZ

Solid colors, large flat areas (alpha and id channels)

RLE

• Intermediary 16/24-bit float, 16/32-bit int

Texture maps, Animation, Graphics or Video

PXR24, if unavailable: ZIP

DWA if you can afford a small quality loss – not supported by ffmpeg (yet).
Note that DWA will only compress R,G,B channels (or Y,RY,BY in case of Luminance/Chorma) and automatically select either RLE for alpha or ZIP for AOV (Z, U, V, Normal...).

Grainy video or animation
PIZ

Solid colors, large flat areas (alpha and id channels)
RLE

• Stereo images

Texture maps, Animation, Graphics or Video
ZIP

Solid colors, large flat areas (alpha and id channels)
RLE

• Real-time playback

B44A or B44
If unavailable: PXR24

• Proxies

DWAA (use a high compression level)
If unavailable: PXR24, ZIP, PIZ

Name	Loss	Description*	Ratio**	Summary	Good for***
PIZ	Lossless	A wavelet transform is applied to the pixel data, and the result is Huffman-encoded. This scheme tends to provide the best compression ratio for the types of images that are typically processed at Industrial Light & Magic. Files are compressed and decompressed at roughly the same speed. For photographic images with film grain, the files are reduced to between 35 and 55 percent of their uncompressed size. PIZ compression works well for scan-line based files, and also for tiled files with large tiles, but small tiles do not shrink much. (PIZ-compressed data start with a relatively long header; if the input to the compressor is short, adding the header tends to offset any size reduction of the input.) PIZ compression is only supported for flat images.	35~55% (photo with grain)	- Write speed = read speed - Best for grainy images	**- Photo/Video (with grain)** **- 3D Animation (with grain)**
ZIP	Lossless	Differences between horizontally adjacent pixels are compressed using the open-source zlib library. ZIP decompression is faster than PIZ decompression, but ZIP compression is significantly slower. Photographic images tend to shrink to between 45 and 55 percent of their uncompressed size. Multi-resolution files are often used as texture maps for 3D renderers. For this application, fast read accesses are usually more important than fast writes, or maximum compression. For texture maps, ZIP is probably the best compression method. In scan-line based files,16 rows of pixels are accumulated and compressed together as a single block.	45~55% (photo without grain)	- Faster reading, significantly slower writing - Same as PNG - Supported for stereo images	**Only when *32bpc float* is needed (otherwise, PXR24 is better):** **- Texture maps** **- Photo/Video (without grain)** **- 3D Animation (without grain)** **- 2D Animation, Graphics**
ZIPS	Lossless	Uses the open-source zlib library for compression. Like ZIP compression, but operates on one scan line at a time.		- Same as PNG - Supported for stereo images	
RLE	Lossless	Differences between horizontally adjacent pixels are run-length encoded. This method is fast, and works well for images with large flat areas, but for photographic images, the compressed file size is usually between 60 and 75 percent of the uncompressed size.	60~75% (photo)	- Fast - Same as TGA - Better with large flat areas (alpha and id channels) - Supported for stereo images	**- Solid colors, large flat areas (alpha and id channels)**
PXR24	Lossless (16bit float, 16/32-bit int) Slightly Lossy $3x10^{-5}$ (32-bit float)	After reducing 32-bit floating-point data to 24 bits by rounding(while leaving 16-bit floating-point data unchanged), differences between horizontally adjacent pixels are compressed with zlib, similar to ZIP. PXR24 compression preserves image channels of type HALF and UINT exactly, but the relative error of FLOAT data increases to about . This compression method works well for depth buffers and similar images, where the possible range of values is very large, but where full 32-bit floating-point accuracy is not necessary. Rounding improves compression significantly by eliminating thepixels' 8 least significant bits, which tend to be very noisy, and therefore difficult to compress. PXR24 compression is only supported for flat images.	Better than ZIP for 32bpc Same as ZIP otherwise	- Faster reading, significantly slower writing - Turns 32-bit float to 24-bit	**Only for *16bpc float* or *16/32bpc int*, or when *24bpc float* is sufficient instead of *32bpc*** **- Photo/Video (without grain)** **- 3D Animation (without grain)** **- 2D Animation, Graphics** **- Texture maps**
B44	Lossy	Channels of type HALF are split into blocks of four by four pixels or 32 bytes. Each block is then packed into 14 bytes, reducing the data to 44 percent of their uncompressed size. When B44 compression is applied to RGB images in combination with luminance/chroma encoding (see below), the size of the compressed pixels is about 22 percent of the size of the original RGB data. Channels of type UINT or FLOAT are not compressed. Decoding is fast enough to allow real-time playback of B44-compressed OpenEXR	44%	- Fixed file size - Very fast read speed	**- For systems needing real-time playback**

		image sequences on commodity hardware. The size of a B44-compressed file depends on the number of pixels in the image, but not on the data in the pixels. All images with the same resolution and the same set of channels have the same size. This can be advantageous for systems that support real-time playback of image sequences; the predictable file size makes it easier to allocate space on storage media efficiently. B44 compression is only supported for flat images.			
B44A	Lossy	Like B44, except for blocks of four by four pixels where all pixels have the same value, which are packed into 3 instead of 14 bytes. For images with large uniform areas, B44A produces smaller files than B44 compression. B44A compression is only supported for flat images.	< 44%	- Very fast read speed - Better with large flat areas (alpha and id channels)	**- For systems needing real-time playback**
DWAA	Lossy	JPEG-like lossy compression format contributed by DreamWorks Animation. Compresses 32 scanlines together.	Varying depending on chosen compression level	- Same as JPEG	**- Proxies (high compression)** **- Final exports when a small compression is acceptable**
DWAB	Lossy	Same as DWAA, but compresses blocks of 256 scanlines.	Varying depending on chosen compression level		**- For proxies** **- Final exports when a small compression is acceptable**

* The description is taken from the official *technical introduction* from https://www.openexr.com
** Compressed / uncompressed. A lower ratio is better. For example, 45% means the compressed file size is 45% of the uncompressed size.
*** A file is good for a given use when its read or write speeds best fits how it is going to be used, and has the best compression ratio available.