ProEXR

Advanced OpenEXR plug-ins for Photoshop and After Effects

fnord

ProEXR

by Brendan Bolles

Version 1.1
September 6, 2007

fnord software
671 Ashbury Street
San Francisco, CA 94117
www.fnordware.com

For support, comments, feature requests, and insults, send email to ProEXR@fnordware.com or participate in the After Effects email list, available through www.media-motion.tv.

Plain English License Agreement

You know the drill. While actually much of the software in this package is free, not all of it is (specifically the ProEXR Photoshop plug-in). If you use it past the trial period, you should buy it. If other people you know want to use it too, they should buy their own copies. Companies should buy a copy for every employee who uses it it. Pretty standard.

A lot of work went into writing these plug-ins and creating this manual. If a few people pay for them, I'll go out and make some more neat-o graphics plug-ins. I promise.

Oh, and please enjoy using this software. Really, I insist.

About ProEXR

Industrial Light and Magic's (ILM's) OpenEXR format has quickly gained wide adoption in the "high-end" world of computer graphics. It's now the preferred output format for most 3D renderers and is starting to become the standard for digital film scanning and printing too.

But while many programs have basic support for the format, hardly any provide full access to all of its capabilities. And OpenEXR is still being developed—new compression strategies and other features are being added while most big application developers are not interested in keeping up.

And that's where ProEXR comes in. It fully supports OpenEXR in Adobe Photoshop and After Effects, at least to the extent that it's reasonably possible to do so in those programs. In particular, ProEXR does these things that Adobe's shipping plug-ins do not:

- It can read in all the channels present in an EXR, not just the standard R, G, B, and A.
- It can output layers from Photoshop, following OpenEXR layer-channel naming conventions.
- It reads and writes information about the project's color space.
- It lets the user choose between all the available EXR compression methods.
- It can greatly reduce a file's size through Luminance/Chroma subsampling.
- It can save images in 32-bit float as well as 16-bit float.

In this manual, "ProEXR" refers to the entire suite of plug-ins for Photoshop and After Effects, although most of the After Effects plug-ins are actually separate and available for free (even open source). But since they are developed in parallel and share the OpenEXR codebase, it makes sense to talk about them together.

System Requirements

To use ProEXR, you'll need CS3 versions of Photoshop and After Effects. Photoshop CS3 Extended is the first version to support layered floating point files. The previous version of After Effects (7.0) simply had a bug that prevented extra channels from working.

Installation

Since Photoshop and After Effects provide some basic OpenEXR support, they already have their own OpenEXR plug-ins—plug-ins we'll want to replace.

Adobe provides	ProEXR replaces with
Photoshop	
OpenEXR.8bi found in Plug-Ins/File Formats	ProEXR.8bi
After Effects	
OpenEXR.8bi found in Plug-ins/Format	OpenEXR.aex EXtractoR.aex IDentifier.aex OrphExtract.aex ProEXR Comp Creator.aex

Note that the extensions above are in the Windows format. On Mac, all plug-ins end with ".plugin".

To disable Adobe's plug-ins, you could of course delete them or move them to another folder the host program won't see, but we prefer to simply disable them. Here are good ways to do it for each platform:

Mac	Add a "¬" (option-L) character to the beginning of the plug-in's name. So "OpenEXR.plugin" becomes "¬OpenEXR.plugin".
Win	Change the plug-in's extension to something the host program won't recognize. For example, you might change "OpenEXR.8bi" to "OpenEXR.wtf".

Feel free to drop the ProEXR plug-ins anywhere within their respective program's Plug-In folder and create your own subdirectory.

If you don't disable the Adobe plug-ins, the programs will still work but you might get some unexpected behavior. When writing a file from After Effects, you will see two OpenEXR options. When reading an EXR in Photoshop, you may not see additional channels in a file. Trust me, it's best to disable them.

Quick Start

If you need a multi-channel EXR file to try out the extra features in ProEXR, download one here:

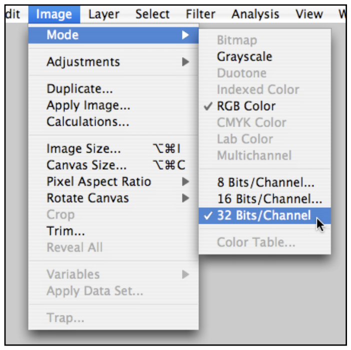
http://www.fnordware.com/OpenEXR/x35_3a.exr

This file was created by Rob Nederhorst.

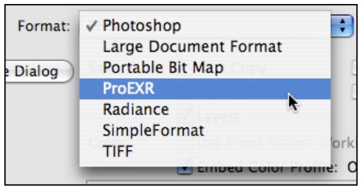
A sample After Effects project using this file is also available.

Photoshop

To save an EXR, your document must be in 32 Bits/Channel mode, as set from the Image menu. Any OpenEXR files you read in will already be set to this mode.

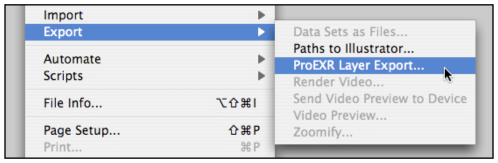


Go to File ➤ Save As. Select ProEXR as the Format.



Configure settings in the Options dialog. Click OK.

Alternatively, you can export a series of layers as individual EXR files by choosing File ➤ Export ➤ ProEXR Layer Export.

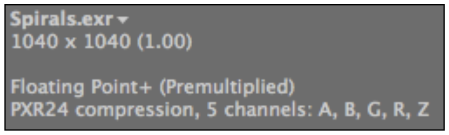


When opening an .exr file in Photoshop's Open dialog, ProEXR should appear as the format and any additional channels should appear as extra layers.

After Effects

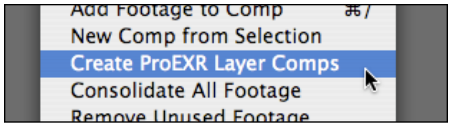
OpenEXR files will appear to import and render as before with the replacement file format plug-in installed. When using an OpenEXR Output Module in the Render Queue, you'll notice that the Format Options button is enabled and a settings dialog appears when clicked.

On import, EXR files appear mostly normal, although the Footage status information now also tells which compression method was used and which extra channels are included in the file, if any.


Spirals.exr ▾
1040 x 1040 (1.00)

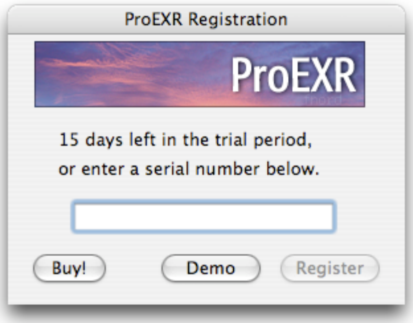
Floating Point+ (Premultiplied)
PXR24 compression, 5 channels: A, B, G, R, Z

To access the extra channels, use one of the plug-ins in the 3D Channel category, particularly EXtractoR or IDentifier. Or select the EXR file and choose "Create ProEXR Layer Comps" from the File menu and a series of comps will be created and configured for you.



Registration

Most of the plug-ins in this package are free, with one exception—the actual ProEXR Photoshop plug-in. After installation, you are given 15 days to try it out before it will refuse to run any longer without a serial number. To purchase a serial number, visit this link:


ProEXR Registration

15 days left in the trial period,
or enter a serial number below.

http://www.fnordware.com/ProEXR/buy_proexr.html

ProEXR History

The plug-ins in this package began their life while I was working as a visual effects artist and occasional programmer at The Orphanage. Around 2002, HDR images and floating point compositing were becoming all the rage at Siggraph and other places but After Effects, with its 16-bit limitation, was largely left out of the party.

To try to work around this, Stu Maschwitz, Ryan Tudhope, myself, and other Orphans created eLin, originally through some AE presets and later through custom plug-ins I wrote. These plug-ins were first used on *Hellboy*, and then refined for *The Day After Tomorrow.* They seemed to do the trick, so eLin was released as a commercial package around this time. The release also contained our first OpenEXR format module—a Photoshop plug-in that presented images to AE in log space.

When ramping up on *Sin City,* the need to access 3D Channels in an EXR was apparent, so an AE-native plug-in was created that could pass on Z-Depth and Velocity channels. Most of the motion blur in our shots was rendered in After Effects using the Velocity channels.

Years later, there was some talk of updating the AE plug-in for After Effects 7, which finally had real floating point but no 3D channel support for EXR files. The Orphanage graciously agreed to let me take the code I had written there and release it publicly. Everything was fine until Steve Cho sent me an image by Rob Nederhorst that didn't just have a few standard 3D channels, but a total of 49 channels containing various render passes, image layers, velocity channels, etc. Steve mentioned that Nuke could easily pull all the channels from this file, and so I decided I had to get the same functionality into After Effects, which involved writing some additional 3D channel plug-ins as well as generalizing the format importer.

At some point in this process, it occurred to me that a similar plug-in would also be great to have in Photoshop. While the interface for loading layers into Photoshop was completely undocumented, I managed to figure it out and was amazed that it all actually worked more or less.

And now after all that, I'm really excited to get to use ProEXR in my own work. And even more excited to share it with the global graphics community… and you (awwww).

Acknowledgments

First and foremost, I have to thank everyone who has contributed to the OpenEXR code at ILM. After all, their code makes up the bulk of this package. I have the individual contributors' names listed later on in this manual.

Special thanks to Florian Kainz for heading up the project at ILM and for answering many of my stupid questions.

Thanks also to whatever corporate big-wig(s) at ILM decided it was OK to release the OpenEXR code to the public. We all owe you.

Thank you to the After Effects team at Adobe, who make a program I really love using. Special thanks to Dan Wilk (DWilk), Dave Simons (DaveS), Bruce Bullis (bbb), and Vladimir Potapyev (Vlad) who have helped me fix my own bugs, fixed some of their own, and generally helped me figure out what's going on.

Thanks to Lars Borg, Manish Kulkarni, and Tom Raurk also from Adobe, who have helped me over the years.

Thanks to everyone I have ever worked with at The Orphanage. Especially Stu Maschwitz, Luke O'Byrne, Alex Prichard, Ralph Procida, Tim Dobbert, Ben Grossmann, and Steve Cho.

Thanks to Jean-loup Gailly and Mark Adler for zlib. Thanks to Marti Maria for Little CMS. Both open source libraries are used throughout ProEXR.

Thanks to everyone who buys these plug-ins—it really means a lot to me.

And thanks to everyone else who ought to be listed here, but is escaping my mind as I write this at 4:45 in the morning.

ProEXR in Photoshop

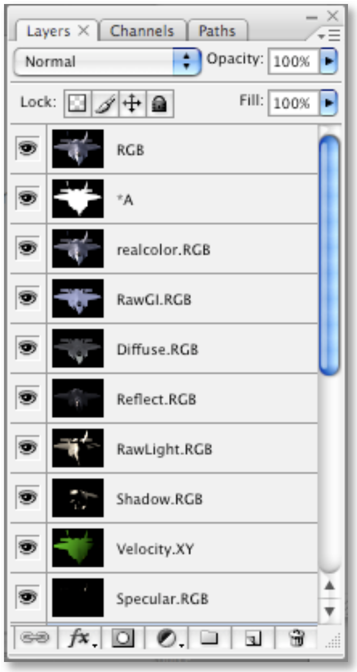
File Reading

The OpenEXR format can support any number of channels, each one defined by a unique name. But most readers (including Adobe's) only pay attention to the channels "R", "G", "B", and "A" to create a single image. ProEXR reads those channels along with any other image channel found in the file.

The best way to see how ProEXR works is to open a multi-channel EXR file with it. Each channel will appear, with some channels grouped together in layers. For more information about how layers are formed from channels in an EXR, see Channel Info.

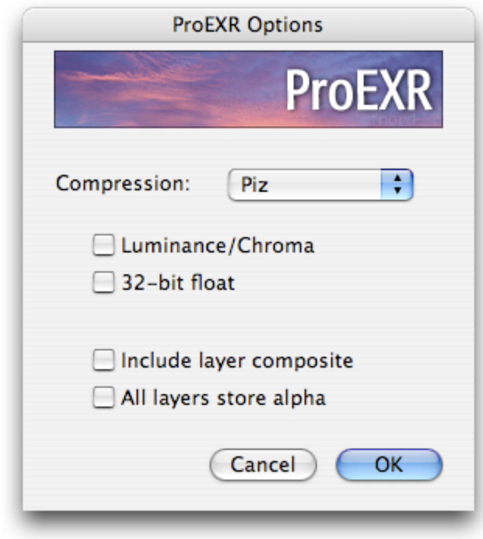
By default, each layer will be given transparency if an alpha channel ("A") is present in the file or if a layer is explicitly assigned its own alpha. For ways to override this behavior, see Modifier Keys.

OpenEXR files always appear in Photoshop in 32-bit (float) mode. Non-float channels such as Object ID will not appear in Photoshop, although you can extract them in After Effects.



File Writing

ProEXR will appear as a format option in the Save As dialog when your document is in 32-bit (float) mode. Unlike most other formats, you will be allowed to save your Photoshop layers. You will then be presented with this nifty options dialog:



Compression: Piz

☐ Luminance/Chroma

☐ 32-bit float

☐ Include layer composite

☐ All layers store alpha

Compression—Choose from the available compression methods. For more information, see OpenEXR Compression.

Luminance/Chroma—Translate RGB into Luminance/Chroma channels and subsample. Only available for single-layer files. See the Luminance/Chroma section.

32-bit float—By default, OpenEXR stores pixels in a 16-bit float format. This is usually ideal, but if you have a need for 32-bit float pixels (perhaps because you have values over 64,000.0?), check this box. Otherwise it will just increase your file size unnecessarily.

Include layer composite—ProEXR lets you explicitly say which channels will be saved in the file, but under this method you will not end up with the standard "R", "G", "B" channels unless you create a layer called "RGB". With this box checked, ProEXR will take a snapshot of your composited layers and use that for the standard RGBA channels that most programs look for (so it's good for compatibility's sake). In general you'll want to leave it on.

All layers store alpha—This forces all layers to save out an alpha channel, even if they ordinarily wouldn't, given the layer's name. See Channel Info for further details.

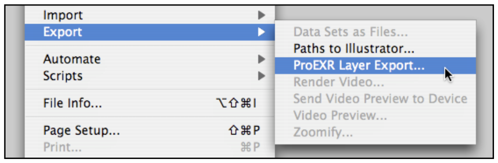
About Alphas

If an EXR has an alpha channel (named "A"), ProEXR will assume that it should be applied to every layer, burning transparency into all of them. If for some reason you disagree with this behavior, you may use a modifier key to override it. On output, the alpha channel is re-saved.

Here's where it gets tricky: OpenEXR files with alpha channels are stored as premultiplied, while Photoshop and After Effects uses straight images internally. By default, ProEXR handles all the conversion back and forth between the two formats. There may be times when you want to overrule ProEXR's alpha conversion—for those times I'll once again point you to our section about modifier keys.

If all this alpha talk is making your head spin, or maybe you can't ever hear enough about alpha channels (I know I can't), see the section called "All About Alphas".

Layer Export



While we all know it's great to be able to save all our layers into a single OpenEXR file [insert pat on back here], the fact is that many programs don't understand such files (including umm…stock Photoshop and After Effects). To work better with such programs, ProEXR has the ability to save out each layer as a separate EXR file. Simply go to File ➤ Export ➤ ProEXR Layer Export.

Upon selecting this option, you will be asked to enter a file name which will serve as the basis for the layers to be exported. So if you had layers called "sky", "buildings", and "lights" and exported a base file called "painting.exr", your layered files would be called "painting_sky.exr", "painting_buildings.exr", and "painting_lights.exr".

If you check the box to "Include layer composite," a full layered EXR with composited RGB channels will be placed as the base file.

Suggested Workflow

Because there are some items in a Photoshop project that aren't supported in an EXR file, we recommend that all Photoshop work be saved as real, honest-to-dog Photoshop files. And then when it's time to export your project to another program using an industry-standard, open-source, production-tested, layer-friendly, floating point format, we'll that's where ProEXR comes in.

In some cases, you may have to do a little project clean-up before exporting, merging this, or rasterizing that. In those cases we hope you can find a way to automate the process using the scripting built in to Photoshop. To help you in this endeavor, ProEXR supports the Photoshop Actions scripting model.

Below is a table of some things ProEXR supports and doesn't, and even why:

Photoshop item	Support?	eh?
Layers	Yes!	That *was* the whole point, right?
Text Layers	Sort of	They get rasterized into pixels.
Adjustment Layers	Not really	They appear in the file, but without an adjustment.
Layer Sets	Not really	The layers do appear, just not in a set.
Transparency	Yes	We like to call it the alpha channel.
Layer Masks	Yes	It gets combined with transparency.
Vector Masks	No	Try a vector format, bub.
Layer order	Yes	Gets stored as a custom attribute read by PS and AE.
Visibility	Yes	Stored in the same custom attribute for AE.
Transfer Mode	Yes	That custom attribute again. AE only.
Opacity	Yes	Yeah, custom attribute. Stored by PS, read by AE.
Color Space	Yes	We store Chromaticity. See OpenEXR Color.
Channels Palette	No	Adobe made me choose either Layers or Channels.
Paths	No	Try EPS.
Layer Styles	No	Ha!
Smart Filters	No	See *Layer Styles.*
Some other item	*Not likely*	*EXR is a bitmap format, man. This ain't TIFF.*

Modifier Keys

As I mentioned before, ProEXR has a certain default behavior when it comes to alpha channels. While we at fnord software think we're really doing the right thing there, we've also provided some secret modifier keys (well, they *were* secret) to effect the process. You want to have these modifier keys down the instant ProEXR starts reading the file, and then you can sit back, relax, and watch the progress bars as they fly through the air.

Modifier	Result
Option (Mac) / Alt (Win)	Alpha channels are not applied to layers but float separately in their own layer. All channels appear as they are in the file—premultiplied.
Option/Alt + Shift	Alpha channels get a separate layer, but the RGB channels are **un-multiplied** by the Alpha. You'll see straight pixels in all their jaggy glory! Hope you know what you're doing.
Control	Ignores the neat-o custom attribute ProEXR stored to remind Photoshop what the original layer names and order was. Instead, layers get assembled old-school.

Channel Info

Photoshop and OpenEXR have different notions of how to store layers, and getting them to sync up isn't always easy. Photoshop thinks of each layer as a group of 4 RGBA channels with a layer name that isn't necessarily unique. Layers also have a stacking order, transfer modes, layer masks, opacity, etc.

OpenEXR, on the other hand, only has a list of channels, each with a unique name. A weakly-enforced layering method has been created through a naming convention: if you have a layer named “reflection” and it has three RGB channels, what you actually have in the EXR are three channels called “reflection.R”, “reflection.G”, and “reflection.B”. You might not have any “reflection.A”, and in some cases you may have fewer channels with different names such as “velocity.X” and “velocity.Y”.

The goal with ProEXR was to retain all the original channel information so that you could open an EXR, make some changes, and the save it with the same channels it originally came in with. So in order to remember what the original channels were, we format the layer names in a special way. Don’t be afraid to ignore these formatting rules—ProEXR will just assume you want all 4 RGBA channels in that case. But if you want more control, follow the examples below:

Photoshop layer name	OpenEXR channels
spaceship	spaceship.R, spaceship.G, spaceship.B, spaceship.A
dinosaur.RGBA	dinosaur.R, dinosaur.G, dinosaur.B, dinosaur.A
alien.RGB	alien.R, alien.G, alien.B
velocity.XY	velocity.X, velocity.Y
depth.Z	depth.Z
normals.[NX][NY][NZ]	normals.NX, normals.NY, normals.NZ
texture.UVA	texture.U, texture.V, texture.A
*Transparency	Transparency (an EXR channel without any layer association)
Specular	Specular.R, Specular.G, Specular.B, Specular.A

Photoshop layer name	OpenEXR channels
RGB	R, G, B (RGB, RGBA, Y are special names that don't become RGB.R, RGB.G, etc)
RGBA	R, G, B, A

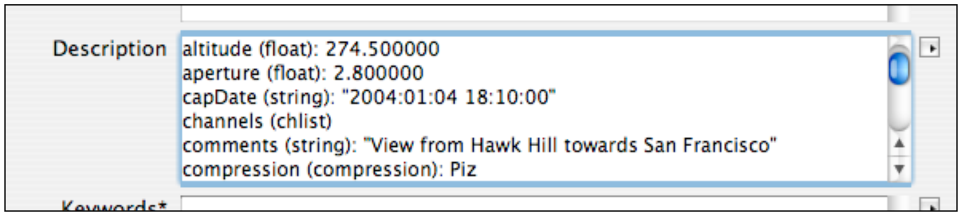
In general, channels are retrieved from Photoshop in RGBA order. The exception is that any channel.A will get the alpha channel. So in the texture.UVA example, red is texture.U, green is texture.V, and alpha (not blue) is texture.A. An asterisk indicates a channel that's not part of a layer, so the "Transparency" channel will live alone and come from that layer's red channel. If "All layers store alpha" is checked, the .A is implicitly tacked on.

If you have duplicate layer names, ProEXR will increment the name to prevent duplicate channel names in the EXR, which aren't allowed.

Finally, ProEXR saves a custom chunk of data with the file so that if you re-open a layered file, the layers will appear as you originally labeled them, not with the layer formatting. (You can use a modifier to ignore this data and force the layer formatting if you like.)

File Description

While ProEXR exposes all floating point channels to Photoshop, an EXR file may have any number of attributes that generally remain unseen. A complete report of all components in the file is inserted into the Description field in Photoshop's File Info window.



```
altitude (float): 274.500000
aperture (float): 2.800000
capDate (string): "2004:01:04 18:10:00"
channels (chlist)
comments (string): "View from Hawk Hill towards San Francisco"
compression (compression): Piz
```

ProEXR in After Effects

Caveat

OK, actually most of the After Effects plug-ins aren't really considered part of the commercial ProEXR package. For one thing, they're free, and even open source. But since they were written in parallel with the Photoshop plug-in and share so much with it, it seemed silly to put them in a separate manual.

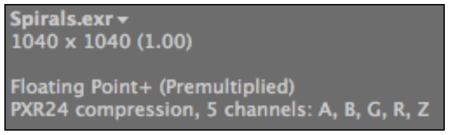
OpenEXR file module

Like the file format plug-in Adobe ships with Photoshop, stock After Effects can only read the standard RGBA channels in an OpenEXR file and provides no options upon writing. Actually, this is hardly an accident seeing as the After Effects file format plug-in *is* the Photoshop plug-in. So naturally it can't support After Effects-specific interfaces such as multiple channels, which is why it must be *replaced.*

Reading

On the surface, our replacement reader imports OpenEXR files the same as the one that ships with After Effects. EXR files appear as premultiplied floating point and the "R", "G", "B", and "A" channels appear when the footage is dragged into a comp.

But select the footage item and look in the status indicator, and you'll see that information about the type of compression is presented. And if you have a file with channels beyond the standard RGBA, you'll see some of them listed there as well.



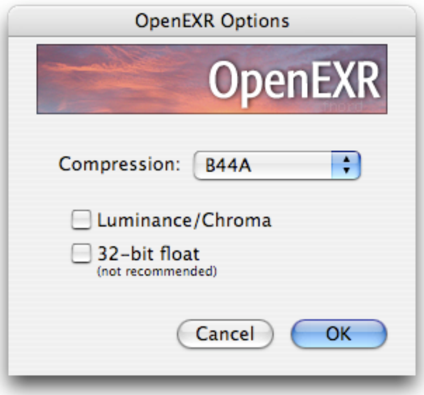
```
Spirals.exr ▾
1040 x 1040 (1.00)

Floating Point+ (Premultiplied)
PXR24 compression, 5 channels: A, B, G, R, Z
```

This is all just the tip of the iceberg, for this format module can read any channel in an EXR file, although to do so it needs to work in concert with 3D Channel plug-ins, some of which are included with this package and described below.

Writing

Once again imitating its Adobe counterpart, the replacement file module initially appears just like the standard Adobe plug-in. But notice that the Format Options button is enabled and clicking it takes you to this pretty dialog:


OpenEXR Options

Compression: B44A

☐ Luminance/Chroma

☐ 32-bit float
(not recommended)

Compression—Choose from the available compression methods. For more information, see OpenEXR Compression.

Luminance/Chroma—Translate RGB Channels into Luminance/Chroma channels and subsample. For more information, see Luminance/Chroma.

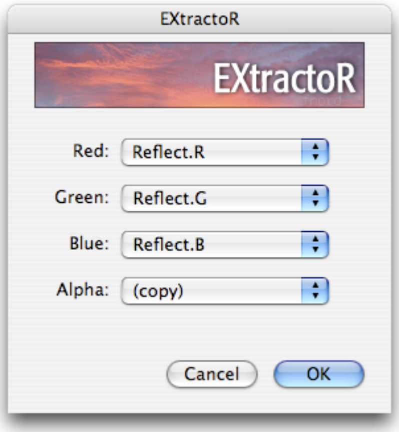
32-bit float—By default, OpenEXR stores pixels in a 16-bit float format. This is usually ideal, but if you have a need for 32-bit float pixels (perhaps because you have values over 64,000.0?), check this box. Otherwise it will just increase your file size unnecessarily, which is why it's not recommended. 32-bit is not available in conjunction with Luminance/Chroma.

Channel Mapping

While the ProEXR package comes with tools to easily extract any channel from a piece of footage, some plug-ins rely on channels being categorized. To play nice with these plug-ins, there is a editable text file that accompanies the OpenEXR reader called *OpenEXR_channel_map.txt*. Instructions for use are found in the file itself.

EXtractoR

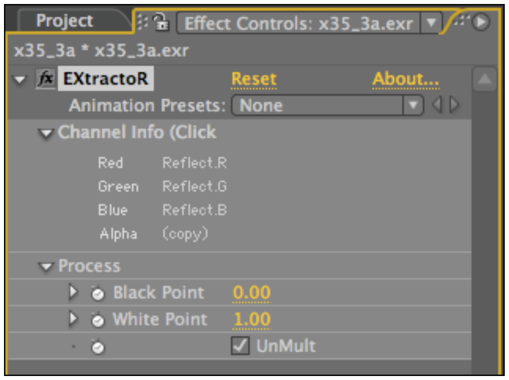
Whereas most 3D Channel plug-ins are limited to a certain set of pre-defined channels, EXtractoR will open any channels available so long as they're floating point. The extracted channel names are listed in the Channel Info pane. As with all 3D Channel plug-ins, you must apply EXtractoR directly to the footage layer, not to a pre-comp.



Red: Reflect.R
Green: Reflect.G
Blue: Reflect.B
Alpha: (copy)

When you click on said **Channel Info** pane, a dialog will pop up allowing you to pick from menus that reflect channels present in the current image. You may also choose to simply copy a channel from the input.

The **Black Point** and **White Point** controls are provided to re-scale the floating point values. At their defaults, no re-scaling will take place. If a project is in floating point mode, the user may mouse over the image and see the raw values in the Info Panel.


Project | Effect Controls: x35_3a.exr

x35_3a * x35_3a.exr

fx EXtractoR   Reset   About...

Animation Presets: None

Channel Info (Click

Red	Reflect.R
Green	Reflect.G
Blue	Reflect.B
Alpha	(copy)

Process

Black Point 0.00

White Point 1.00

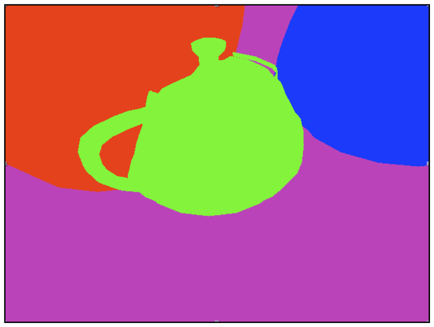
✓ UnMult

Channel values will depend on the renderer. Z-depth values are often negative, which might call for setting the White Point to a negative number, perhaps a large one. Velocity channels can be positive or negative, but usually should be scaled to 0-1 range with 0.5 meaning no movement. This can be done by setting the white and black points to opposite values like 50 and -50.

Most 3D renderers output images premultiplied, while AE holds them as straight internally. If you are bringing in a premultiplied RGB channel, you probably want to click the **UnMult** checkbox. More detail on this exciting topic available in the All About Alphas section.

IDentifier

Whereas EXtractoR pulls floating point channels out of an EXR file, this plug-in gets the non-image channels such as Object and Material ID. It uses a dialog similar to EXtractoR's.



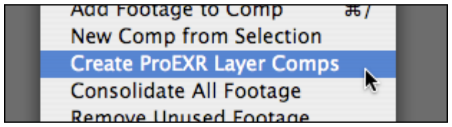
The default "Colors" display is meant to provide an easy preview, showing that the ID channel has been found and that objects in the scene have different IDs. The Luma and Alpha matte options are what you will probably end up using in a comp. Raw output (scaled to 8-bit) is also provided if you want to sample ID numbers with your mouse.

OrphExtract

This was originally developed at The Orphanage as a way to get Velocity Channels out of a file (which wasn't supported by AE's normal 3D Channel plug-ins). It still has this ability, and is now otherwise like Adobe's "3D Channel Extract" plug-in, except that it works in float. A simpler alternative to EXtractoR for basic channels.

ProEXR Comp Creator

As the name suggests, this plug-in is considered part of the non-free portion of ProEXR. Select an EXR file in the Project pane and select "Create ProEXR Layer Comps" from the File menu. This plug-in will create source comps for each layer in the file, apply the appropriate extraction plug-in, and add each source comp to an assemble comp. Files saved out of Photoshop will retain layer ordering, transfer mode, etc. through a custom attribute saved in the file. Not bad!



For times when you need extra control over the set-up of source comps, you can set the tone by dropping the EXR into a new comp and positioning it as you like. Then select the layer, apply the menu command, and all the source comps will match your original comp, which then becomes the assemble comp.

A note about color management

The extraction plug-ins take floating point data from an EXR file and put it right into AE's pixel buffer. As a result, the pixels are simply assumed to be in your project's working space, although that may not be true. After Effects tags incoming floating point files as being in linear sRGB color space or applies color space information if it is available in the EXR. Usually this is all taken care of automatically on the file import end, but EXtractoR is circumventing that step.

To use color management with extracted images, apply the Color Profile Converter, select sRGB as the input profile (or another color space if you have one), and check “Linearize Input Profile.” Keep the project working space as your output profile, and now the layer is once again color managed.

Non-image data channels such as Z-depth and Velocity don’t have any notion of color space, so you’ll usually want to leave them unmanaged.

Note that importer will tell AE about any color information found in the EXR, but the resulting profile will not be available in the Color Profile Converter. Sounds like an Adobe feature request to me.

General OpenEXR Info

About OpenEXR

In recent years as computing power has continued to increase, most everyone working at the bleeding edge of computer graphics is working in floating point. And that means a floating point format is needed. And the format of choice is OpenEXR, created by some folks known to trip right over the bleeding edge—Industrial Light and Magic.

OpenEXR's predecessors were never ideally suited for the job. Formats like floating point TIFF and Radiance were either inefficient with their storage of floating point data, limited from a production standpoint, or both. ILM needed a format to suit their needs, so they went out and created one. Then they released the code for it as open source.

The OpenEXR format is very flexible. It can hold any user-defined attributes in addition to a set of standard ones. An EXR can hold any number of channels, each identified by an arbitrary name. Although I should mention that since many programs only seek out the following sets of reserved channels, it's always good to have them present to insure maximum compatibility:

RGB	Grayscale	Luminance/Chroma	Alpha channel	Z-Depth
"R" "G" "B"	"Y"	"Y" "RY" "BY"	"A"	"Z" *(unofficial)*

OpenEXR images are always floating point. But because full 32-bit floating point image data tends to be unwieldy and compresses poorly, OpenEXR uses a 16-bit ("Half") floating point format that matches what is found in some modern graphics cards. For digital images which generally have some degree of noise or sampling error, the extra precision from a 32-bit float file is really meaningless. Of course, OpenEXR does support full 32-bit float if you need it, as well as integer pixels for non-image data like Object ID.

To store pixel data more effectively, OpenEXR utilizes image compression, typically lossless to avoid any kind of image degradation, yet still provide a significant reduction in file size. When image fidelity is less crucial, lossy compression and chroma subsampling are available to further crush file sizes.

So all in all, OpenEXR is a pretty amazing format. Let's give a shout-out to the ILM authors who made it for us, shall we?

Florian Kainz
Rod Bogart
Drew Hess
Paul Schneider
Bill Anderson
Wojciech Jarosz
Andrew Kunz

Thanks guys!

For more information on OpenEXR, visit the official site:
http://www.openexr.com/

OpenEXR Compression

OpenEXR supports a variety of compression methods, described in the table below:

None	No compression, but reads quickly.
RLE	Run Length Encoding. Still reads fast, but somewhat smaller.
Zip	Compresses each line with Zip. Slow compression, but faster decompression.
Zip16	Zip 16 lines together. Particularly ideal for CG renders, textures.
Piz	Wavelet encoded, then Zipped. Best for photographic HDRs.
PXR24	Use with 32-bit float. Rounds to 24-bit for better compression.
B44	Good lossy compression, squeezing channels to 44% their original size.
B44A	Same as B44, but with extra compression on areas with flat color.

Except for the last three, EXR compression methods are lossless.

Compression performance depends on the image content. For images with a lot of continuous color (such as motion graphics or CG renders), Zip16 is almost always the best-performing method, beating even B44 in both file size and speed. B44A was created for lossy compression of such images.

For photographic HDR images (especially with film grain), 16-bit Piz usually gives the best combination of image fidelity and file size. If you are less concerned about fully lossless compression, Luminance/Chroma sampling and B44A compression will make a much smaller file.

For example, I created a 32-bit HDR photograph in Photoshop and saved it using various file formats and settings. In descending order of size they were:

File	Settings	Size
TIFF	32bpc	3.9 MB
Photoshop (PSD)	32bpc	3.9 MB
OpenEXR	32bpc, None	3.9 MB
OpenEXR	32bpc, Piz	3.7 MB
OpenEXR	32bpc, PXR24	2.1 MB
OpenEXR	16bpc, None	2.0 MB
OpenEXR	16bpc, RLE	1.7 MB
OpenEXR	16bpc, Zip	1.4 MB
OpenEXR	16bpc, Zip16	1.3 MB
OpenEXR	16bpc, Piz	1.3 MB
OpenEXR	16bpc, B44, L/C	443 K

The first two EXR formats held the original 32-bit float pixels without any change at all. Using PXR24 compression and then 16bpc pixels results in progressively more loss, but the changes are mostly academic, resulting in no loss that could possibly be detected visually. While B44 compression and Luminance/Chroma sampling made significant changes to pixel values, the image still had no visual artifacts, and slight differences were only detectable in areas of high contrast and saturation. (Note that B44 compression and Luminance/Chroma are newer features of the OpenEXR format and may not be supported by all current readers.)

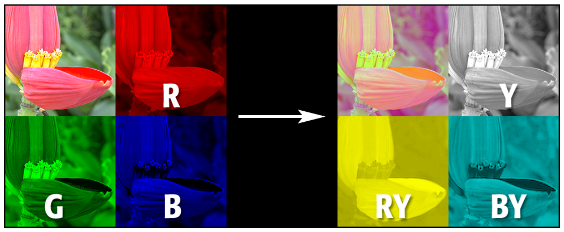
Luminance/Chroma

In any level of graphics production, it's never desirable to use formats such as JPEG that will degrade the quality of the image you're working on. Such formats *are* valuable at the final delivery stage where they can dramatically shrink file size without compromising too much image fidelity, but fidelity should always be maintained in production.

OpenEXR was created for a production environment, so image fidelity was always a top consideration. Like TIFF and PNG, OpenEXR files are generally saved losslessly, even if the the image data is compressed within the file. No pixels are harmed.

But now OpenEXR has added some lossy compression features to emulate JPEG compression in floating point. The first feature is the B44 (and B44A) compression mentioned above. B44 is similar to the lossy DCT compression used in JPEG for compressing blocks of pixels.

The second feature, also found in JPEG, is the breaking down of standard RGB channels into Luminance and Chroma channels ("Y", "RY", and "BY"), and then shrinking (subsampling) the color-difference channels separately.



Because the RY and BY channels contain ¼ the number of pixels, the overall file data is reduced by ½. And yet, in most cases there is no visual difference due to human perception being more sensitive to slight changes in brightness and less to slight changes in color. Combine Lumninance/Chroma sampling with B44A compression and you have an ideal way to save bandwidth when sending EXR files through email.

Linear color space

Most floating point formats, including OpenEXR, assume that pixels are stored in a linear color space. This means that pixel values displayed directly on a monitor will appear unusually dark because your display is non-linear (it has a gamma of 2.2 or so) and is darkening the image. Other formats such as JPEG store pixels brightened up to compensate for this darkening (using the inverse gamma), which is why they appear normal.

The reason linear color space is important is because that's the color space of the real world. Gamma encoding is something made up for use by computers and their 8-bit displays. By performing all your image transformations in linear color space, you are therefore better simulating real light. Many happy accidents are the result, such as blurry highlights spreading out more realistically and blowing through a motion blurred object in the foreground.

Visual effects are generally trying to simulate happenings in the real world. That, and the need to represent overbright (> 1.0) pixel values, is why ILM created OpenEXR as a linear floating point format.

However, some operations are in fact not trying to simulate natural phenomena. One good example is the adding of grain, which in the real world happens on a piece of film or in a camera's CCD—on the encoding end. To simulate the adding of grain, you should convert the image to the appropriate color space (such as Cineon Log), add the grain, and then convert back.

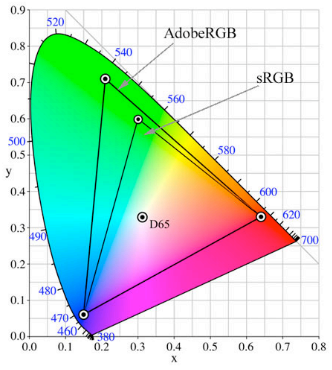
You'll find that by using color space to your advantage and performing operations the "right" way, your work will tend to elevate and your process will make sense.

All 32-bit Photoshop documents are automatically assigned a linear color space. In After Effects, enter the Project Settings, choose a Working Space profile, and check "Linearize Working Space" to composite in linear. And of course, always set your AE project depth to 32-bit float.

OpenEXR color information

All Adobe color management is done using ICC Profiles. Profiles define the color space for a document—the mapping of RGB pixel values in the computer to their actual appearance when viewed on a particular device. The idea is that if we know the color space a file was created with and also know the color space our new device, color management software can tweak values so that the actual appearance is the same both. Well, that's the theory.

ICC profiles typically define the RGB color primaries and then provide a gamma-ish response curve. Since OpenEXR files are always defined to use a linear response curve, a full ICC Profile was not deemed necessary and instead the RGB primaries are defined using chromaticity, which is then stored in the EXR.



You may recognize the diagram to the right—it's a chromaticity diagram. **Chromaticity** for a color is specified with a pair of coordinates (x, y) on this graph. As the name implies, chromaticity only contains information about color without containing any information about brightness. The colored area of the graph represents all the colors that a human can see.

A complete description of a linear color space can be made by providing (x, y) color coordinates for each of the three RGB "tristimuli", along with the white point coordinates (the D65 dot in the diagram).

To pass color space information to the Adobe apps, ProEXR converts between the ICC Profiles from the host and Chromaticity settings in the EXR. A custom attribute is also saved to remember the color space's name.

OpenEXR support in other apps

Wondering who you might exchange these fancy EXR files with? Wonder no more…

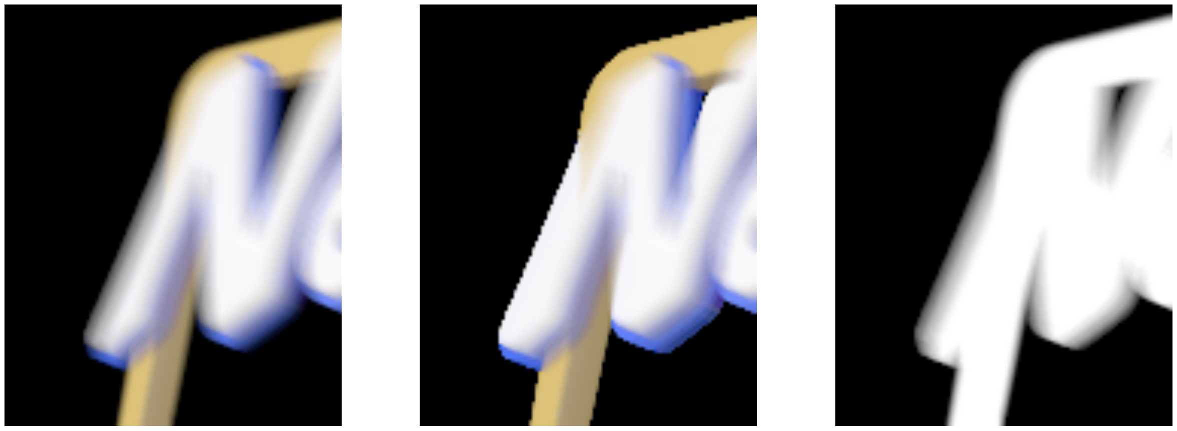
Application	OpenEXR support
Maya	Basic support with included Mental Ray renderer. 3Delight, Renderman other renderers also support OpenEXR.
3DS Max	Max now includes Splutterfish's OpenEXR format module, which saves all 3D channels available in Max.
Lightwave	Full multi-channel output support using exrTrader, www.exrtrader.com.
Softimage	Multiple channel support possible using Mental Ray shaders.
Cinema 4D	Basic RGBA support.
Houdini	Full support for piping arbitrary data to EXR channels.
Shake	Full access to all channels from the FileIn node.
Fusion	Good multi-channel input and pipeline support.
Nuke	Full support for multiple channels through the entire pipeline.

All About Alphas

Transparency in an image is indicated through the use of an alpha channel. A 3D renderer will include an alpha and transparent parts of a Photoshop layer (made with the eraser, a layer mask, or other means) become the alpha channel.

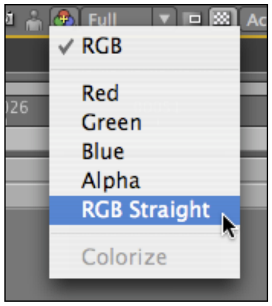
When an alpha channel is present, the RGB channels can be in one of two forms: **premultiplied** or **straight**. The difference between the two is seen in translucent areas of the image, such as the motion blurred edges in the sample below. Fully opaque and fully transparent areas are not effected by this.

The edge of the CG object in the image below is white, but the pixels have been made translucent by motion blur, so they mix with the black background and become grey. In a **straight** image, this original state is remembered—the white pixels on the edge stay white until the end of the object and it is left for the alpha channel to provide a soft edge. In the **premultiplied** image you see the result of the alpha getting applied—the edge appears soft as it has already been composited over black. It turns out that this operation is a *multiplication* of the straight RGB with the alpha, hence the image is pre-*multiplied*.



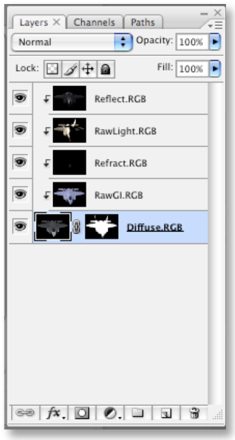
In general, it doesn't matter which type of image is being stored because each one can be converted into the other. But it *is* important that the programs you use receive the type of image they expect.

If they make an assumption, most programs today assume a premultiplied image is being used. Conveniently, the OpenEXR spec specifically specifies premultiplied images. 3D renderers usually create premultiplied images and most compositing programs expect them … *except for the Adobe programs*. Both Photoshop and After Effects use straight pixels internally, so ProEXR must un-multiply (UnMult) the RGB channels to create a transparent layer. In After Effects, you can view raw straight pixels using the channel view pulldown. These straight pixels should look jagged around the edges—if they don't, maybe you forgot to interpret them premultiplied or click UnMult in EXtractoR.



On the way out, ProEXR premultiplies layers in Photoshop before saving them in the EXR. In After Effects, the user should leave the color premultiplied in the output module.

There may be times in Photoshop where you don't want ProEXR to UnMult the RGB channels or when you want to use them without the alpha being applied to the layer. See the modifier key section for information on how to do that.



One specific case of alpha manipulation appears when you want to use multiple render passes to make a composite in Photoshop. To do this correctly, you must load the EXR with channels UnMulted (hold down option/alt + shift). Then create a layer mask on your bottom layer and copy the alpha channel into it. From there, other layers can be dropped on top of the base layer and then **grouped** by option-clicking on the lines between them in the layers palette. Only when your channels are straight and the layers are grouped together will the edges merge together properly in Photoshop. It ain't pretty, I know.

I should mention one more thing: when you copy the alpha channel into the layer mask, Photoshop will think it's doing you a big favor by converting it from linear space to perceptual space and brightening it up. For best results, counteract this move by applying a Levels call to the layer mask with a gamma of 0.4545.

Version History

1.0—August 20, 2007—Initial release using OpenEXR version 1.6.0.
1.1—September 6, 2007—Added File Info description, enhanced Comp Creator.

OpenEXR Copyright

All ILM wants me to do in return for using their free OpenEXR library is print this message:

Fin

fnord