Storing Multi-View Images in OpenEXR Files

Last Update: 11/27/07

This document recommends a standard way to represent multi-view images in OpenEXR files. A "multi-view" image shows the same scene from multiple different points of view. A common application is 3D stereo imagery, where a left-eye and a right-eye view of a scene are stored in a single file.

A standard multi-view OpenEXR file is identified by the presence of a `multiView` attribute in the header, which lists the views stored in the file. Each view has its own set of image channels. A naming convention associates the channels with the views.

The Multi-View Format

The multi-view format is designed to

1. Allow more than one view to be stored in a single OpenEXR file in a standard way with which all applications can comply. In this respect the format is not limited to mono or stereo views, but allows any arbitrary number of views.
2. Formalize the names of left and right views so stereo viewing software and hardware always displays the correct view.
3. Provide a "default view" to software that does not support multi-view images. Software expecting regular single-view OpenEXR files will see all the channels it expects.
4. Support multiple channels and layers for each view.
5. Allow each view to be stored in separate files, and provide a standard by which the viewpoint of the file can be identified from internal information, without relying on the filename or user interaction.

View Naming

A multi-view OpenEXR file may contain any number of views. If there is a view which is ultimately intended for viewing in a stereo environment by the left or right eye, it must be called `left` or `right` respectively (in lower-case English).

Stereo viewing software may default to assuming that it should send the `left` and `right` views to the left and right eyes respectively, and may ignore all other views.

Other view names should be kept as short as possible to keep the channel names within OpenEXR's 31-character limit.

Channel Naming

Multi-view OpenEXR files use the convention that channel names are composed of layer names separated by periods, with the final channel name at the end.

The view name must be the ultimate layer name, that is, the penultimate period-delimited component in each channel name. In other words, the view name is followed by a period and a final channel name in the format *`layer.view.channel`* or *`view.channel`*.

For example, each of the following names corresponds to a channel in either the `left` or the `right` view:

```
lighting.left.R lighting.left.G lighting.left.B
lighting.right.R lighting.right.G lighting.right.B
noshadows.nearscene.right.R noshadows.nearscene.right.G
noshadows.nearscene.left.R noshadows.nearscene.left.G
left.R left.G left.B left.A left.X left.Y left.shadows
```

Note that, although *`view.channel`* is a valid name under this scheme, *`layer.view`* (for example, `depth.left`) is not valid. (Either use `left.depth` or `depth.left.data`.)

Where a channel is present in more than one view, the names of the channel's instances must differ only in the view part: if a channel in `view1` is called *`xxx.yyy`*`.view1.`*`zzz`*, then the same channel in `view2` must be called *`xxx.yyy`*`.view2.`*`zzz`*.

Default Channel Naming

For compatibility with single-view OpenEXR files, each multi-view file identifies one of its views as the "default view." All channel instances whose names contain no periods (for example, `R`, `G`, `B`, `A`, or `Z`) belong to the default view.

If a file contains RGBA data, then the corresponding channels in the default view should be labeled `R`, `G`, `B` and `A`. For example, if a file has a `right` and a `left` view, and the `right` view is the default view, then the channels must be named

```
R G B A left.R left.G left.B left.A
```

rather than

```
right.R right.G right.B right.A left.R left.G left.B left.A.
```

Channels labeled `R`, `G`, `B` and `A` will be understood by any OpenEXR viewing software, even if the software does not explicitly recognize multi-view OpenEXR images.

If *`view1`* is the default view, and *`xxx`* is a channel in the default view, then the same channel in *`view2`* is called *`view2.xxx`*.

Channels Not in a View

If a channel contains image data that is not associated with any view, then the channel must have at least one period in its name, otherwise it will be considered to belong to the default view. The channel's name must also not contain any view name.

For example, `background.data` is not associated with any view, but `background` belongs to the default view, and `got.it.right.now` may be part of the `right` view.

The multiView Attribute

A multi-view OpenEXR file is identified by the presence of an attribute called `multiView` in the file header. The value of the attribute is of type "array of strings" (C++ type `Imf::StringVector` or `std::vector<std::string>`). The attribute contains a list of view names, one per array element. View names are arbitrary, except that periods and spaces are not permitted within a name. The first listed view (array element 0) is always the default view. Other view names may appear in any order.

For example, the following table shows several different combinations of `multiView` attribute values and channel names:

views listed in `multiView` attribute	channel names
`left right`	`R G B A` `right.R right.G right.B right.A`

views listed in `multiView` attribute	channel names
`right left`	`R G B A` `left.R left.G left.B left.A`

`mono right left`	`R G B A`
	`right.R right.G right.B right.A`
	`left.R left.G left.B left.A`

`mono right left`	`mono.X mono.Y`
	`right.X right.Y`
	`left.X left.Y`
	(default channel naming is optional for non-RGBA data)

Files with Only One View

Multiple views of the same scene may sometimes be stored in separate files rather than in a single multi-view file. An OpenEXR file that contains one of multiple views of a scene should identify the stored view using the `multiView` header attribute.

For example, if RGBA data for the `left` and `right` views of a scene are stored in two separate files, both files should contain a `multiView` attribute. In one file the attribute lists only the `left` view; in the other file only the `right` view is listed. Each file has four channels, called `R`, `G`, `B` and `A`.

It is recommended that stereo viewing software uses the `multiView` attribute to identify views rather than relying on file names, user interaction or other external information.

Library Support

The OpenEXR file I/O library, IlmImf, provides utility functions to support reading and writing multi-view files. Header file `ImfStandardAttributes.h` defines functions to add a `multiView` attribute to a file header, to test if a file header contains a `multiView` attribute, and to access the value of the `multiView` attribute. Header file `ImfMultiView.h` defines functions related to accessing channels and views, such as finding all channels in a given view, or finding the same channel in all views.

File Name Extension Support

Users may wish to save multi-view OpenEXR files with a file name extension other than the commonly used `.exr`, for example, `.sxr` for Stereo eXR, or `.mxr` for Multi-view eXR. The 3-letter extension space is crowded, but SXR and MXR are not heavily used. Either or both of these may become a standard for multi-view OpenEXR files.

Meanwhile, it is recommended that software vendors provide a mechanism allowing users to identify which file name extensions they will employ for multi-view OpenEXR files. For example, software may understand the environment variable

```
MULTIVIEW_EXR_EXT="SXR"
```

or provide a similar facility in a configuration file or registry entry. Software can then default to saving multi-view OpenEXR files with this extension, and also try loading files with this extension as OpenEXR files before resorting to brute force attempts to interpret the file format from the header data. If not specified, the default extension for multi-view OpenEXR files should be `.exr`.