

Beachball example image sequence

The example images in this folder are singlepart and multipart versions of the same multichannel image sequence. The multipart version is compatible only with OpenEXR-2.0 and later. These are intended to exercise many features of the (regular scanline) InputPart interface to the library and are recommended test cases for code which reads them. Code should either read them all correctly, read some parts of the file correctly, or else at the least report errors gracefully.

When viewed in a stereo viewing environment, the images form a sequence of a ball moving towards screen right. The ball should appear to float in front of the screen, not behind it.

Note: the content of channels other than RGBAZ in these examples is arbitrary, and should not be considered a standard approach for representing and naming conventions for data such as motion vectors, stereo disparity or grading masks. They have simply been included here as a realistic example of non-RGBA data.

Note the following about these images:

- These images are **stereo**, multiview images, containing data for both left and right eyes.
The single part file has a `multiView` attribute, the first part of the multipart file has a `view` attribute
- The right eye is the **default** or **hero** eye in this case
The first part in the multipart image has `view` attribute set to `right`; the single part image lists "right" as the first view in the `multiView` attribute.
- **View names are present** in the channel names of the **single part** file, except for the right view's RGBAZ views, which have no view names
- **View names are not present** in the **multipart** file channel names
- The **first part** of the multipart exr contains the **default channels** of the **default view**
Software recompiled against EXR-2.0 which doesn't use the multipart API will only load the default channels of the default view
- **Layer names are present** in both the single and multipart file – they are not dropped, nor is the part name used to derive layer names.
- **No part in a multipart file can contain channels for multiple views**
In a file with more than one part, the `view` attribute is used to identify the view for all channels in that part, and all channels belong to the specified view
- **Parts have consistent displayWindow attributes**
- **Parts do not have consistent dataWindow attributes**

The ability to specify different dataWindows for different channels, by dividing them into different parts, is one of the motivating factors for the EXR-2.0 multipart extension.

Code reading from different parts must be sure not to read from scanlines which are not present in the part's dataWindow, as that will result in an exception.

Reading a channel with a dataWindow smaller than the memory allocated could result in uninitialised memory.

The first part's dataWindow is not guaranteed to enclose the dataWindow of other parts

- In frames 7 and 8, the **dataWindow extends outside the displayWindow**
- **Division of channels within one view between parts is arbitrary.**

In this case, generally each part contains a separate layer, though the RGBAZ layer has been split into across one part for RGBA and another for Z

The decision of how to 'package' channels into different parts is generally driven by read performance and filesize requirements. If only RGBA channels are commonly read, it would be best to store only those channels in a part, as in this case. For realtime playback of just RGB, it may be advantageous to store A separately to RGB. Many rarely read data channels which have similar data content and dataWindows would compress better if stored in the same part. Notice the scheme used here leads to files which are approx 20% larger than storing all channels in one part.

- The disparityR and disparityL channels are **not associated with a view**

There is no view name in the channel names in the single part file, even though it has a layer name, indicating it is not view associated. The disparity parts in the multipart file do not have a `view` attribute

Arguably disparity data is associated with the source view - it is included here merely to illustrate that channels needn't be associated with a view.

- The **depth** channel is called "**Z**" in all cases, in keeping with the convention for deep images

The convention is optional for regular scanline and tile images, but is practical to maintain it for all image types.

This means that depth is part of the RGBAZ layer in EXR parlance. Many software packages internally associate depth differently to RGBA.