10.6 Render - Post Processed Effect

Post Processed Effects

There are several effects you can enable in the Render Settings that add visual elements to rendered images, after the rendering has completed. These are not done in camera, but rather composited on top of the image.

Composited and *Sequence* are discussed in *Output Options*.

Fields are discussed in *Video Output*.

- Render Layers
  - Layer List
  - Layer Panel
  - Using Render Layers
- Post Processing Panel
  - Dithering
- Metadata
- Color Management
  - Scene Linear Color Space
  - Settings
  - Image Files
  - OpenColorIO Configuration
  - Compatibility

Render Layers

Reference

Editor: *Properties*
Context: *Render Layers*

Render layers allow you to render your scene in separate layers, usually with the intension of compositing them back together afterwards.

This can be useful for several purposes, such as color correcting certain elements differently, blurring the foreground as a fast manual method of creating DoF, or reducing the render quality for unimportant objects.

Using Render Layers can also save you from having to re-render your entire image each time you change something, allowing you to instead re-render only the layer(s) that you need.

Layer List

This is a list of all the Render Layers in the current scene.

Only layers which are enabled (checkbox on right is ticked) will be rendered. If the *Pin* icon at the bottom right of the list is enabled, only the active (highlighted) layer will be rendered.

Render Layers can be added and removed using the `+` and `-` buttons on the right, and existing layers can be renamed by double clicking on their name.

Layer Panel

The Layer Panel shows the settings of the active Render Layer from the list above.

You can select multiple layers using `Shift-LMB`.

Scene
The Scene Layers, showing which are currently visible and will be rendered.

Layer
The Scene Layers which are associated with the active Render Layer. Objects in those Scene Layers will be rendered in that Render Layer. When an object is in the Scene Layers but not the Render Layer, it will still cast shadows and be visible in reflections, so it is still indirectly visible.

Mask Layer
Objects on these will mask out other objects appearing behind them.

Material Override

Overrides all material settings to use the Material chosen here.

Examples of where this might be used:

- To check lighting by using a plain diffuse material on all objects
- Render a wireframe of the scene
- Create a custom render pass such as an anti-aliased matte or global coordinates.

See also

Additional options shown in this panel are different for each render engine. See these options for:

- *Blender Render*
- *Cycles*

Using Render Layers

Each Render Layer has an associated set of *Scene Layers*. Objects which are on one of the associated Scene Layers are shown in that Render Layer, as long as that Scene Layer is also visible.

Warning

Only the objects in visible Scene Layers will be rendered. So, if only Scene Layer 1 is visible and your Render Layer set specifies to render only Layers 2 and 3, nothing will be rendered.

Post Processing Panel

Compositing
Use compositing for the final image.
Sequencer
If sequencer strips are used render them instead of an image.

Dithering

Dithering is a technique for blurring pixels to prevent banding that is seen in areas of gradients, where stair-stepping appears between colors. Banding artifacts are more noticeable when gradients are longer, or less steep. Dithering was developed for graphics with low bit depths, meaning they had a limited range of possible colors.

Dithering works by taking pixel values and comparing them with a threshold and neighboring pixels then does calculations to generate the appropriate color. Dithering creates the perceived effect of a larger color palette by creating a sort of visual color mixing. For example, if you take a grid and distribute red and yellow pixels evenly across it, the image would appear to be orange.

The *Dither* value ranges from 0 to 2.

Metadata

The *Metadata* panel includes options for writing meta-data into render output.

Stamping can include the following data:

Time
Include the current scene time and render frame as `HH:MM:SS.FF`
Date
Include the current date and time.
RenderTime
Include the render time in the stamp image.

Frame
Include the frame number.

Scene
Include the name of the active scene.

Camera
Include the name of the active camera.

Lens
Include the name of the active camera’s lens value.

Filename
Include the filename of the .blend file.

Marker
Include the name of the last marker.

Seq. Strip
Include the name of the foreground sequence strip.

Note
Include a custom note.

Note

Only some image formats support metadata: See *image formats*.

Stamp Output

You can optionally stamp this into the image its self (adding text over the rendered image) which can be useful for test renders and animation previews.

Stamp Text Color
Set the color and alpha of the stamp text.

Stamp Background
Set the color and alpha of the color behind the text.

Font Size
Set the size of the text.

Hint

It can be useful to use the *Note* field if you’re setting up a render-farm.

Since you can script any information you like into it, such as an identifier for the render-node or the job-number.

For details on stamping arbitrary values, see: this page

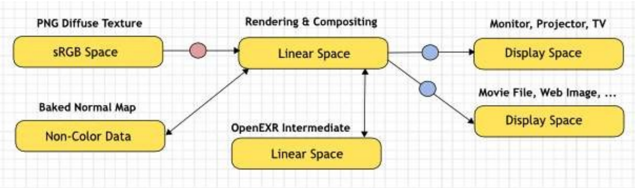
Color Management

Different views and exposures of the same render

OpenColorIO is integrated into Blender, meaning many color spaces are supported with fine control over which color transformations are used.

Scene Linear Color Space

For correct results, different color spaces are needed for rendering, display and storage of images. Rendering and compositing is best done in **scene linear color space**, which corresponds more closely to nature, and makes computations more physically accurate.

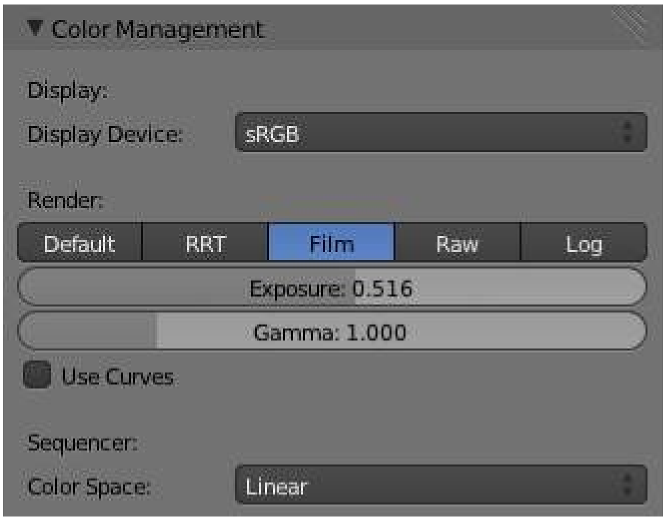


Example linear workflow

If the colors are linear, it means that if in reality we double the amount of photons, the color values are also doubled. Put another way, if we have two photos/renders each with one of two lights on, and add those images together, the result would be the same as a render/photo with both lights on. It follows that such a radiometrically linear space is best for photorealistic rendering and compositing.

However these values do not directly correspond to human perception or the way display devices work, and image files are often stored in different color spaces, so we have to take care to do the right conversion into and out of this linear color space.

Settings



Display:

Display Device: sRGB

Render: Film

Exposure: 0.516

Gamma: 1.000

□ Use Curves

Sequencer:

Color Space: Linear

Scene settings for color management

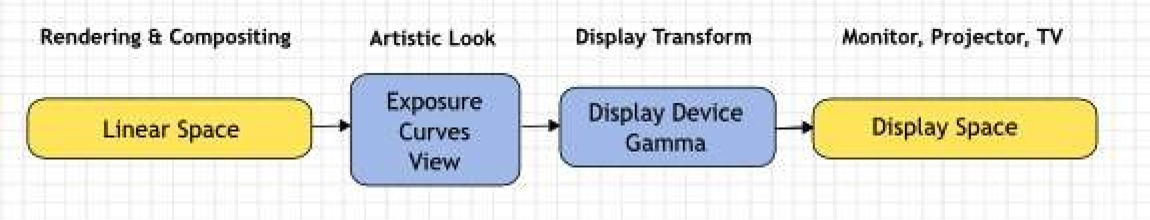
These settings are found in the scene context of the properties editor, under the *Color Management* panel.

Display

Correct display of renders requires a **conversion to the display device color space**, which can be configured here. A computer monitor works differently from a digital cinema project or HDTV. The scene properties have these settings:

Display Device

> The device that the image is being viewed on.
>
> Most computer monitors are configured for the sRGB color space, and so when working on a computer usually this option should just be left to the default. It would typically be changed when viewing the image on another display device connected to the computer, or when writing out image files intended to be displayed on another device.
>
> Rec709 is commonly used for HDTVs, while XYZ and DCI-P3 are common for digital projectors.
>
> Color management can be disabled by setting the device to None.



Conversion from linear to display device space

Render

There is also an **artistic choice** to be made for renders. Partially that’s because display devices can’t display the full spectrum of colors and only have limited brightness, so we can squeeze the colors to fit in the gamut of the device. Besides that it can also be useful to give the renders a particular look, e.g. as if they have been printed on real film.

Another common use case is when you want to inspect renders, to see details in dark shadows or bright highlights, or identify render errors. Such settings would be only used temporarily and not get used for final renders.

View

These are different ways to view the image on the same display device.

Default
Does no extra conversion besides the conversion for the display device.

RRT
Uses the ACES Reference Rendering Transform, to simulate a film-like look.

Film
This option is another film-like look.

Raw and Log
Intended for inspecting the image but not for final export. Raw gives the image without any color space conversion, while Log gives a more “flat” view of the image without very dark or light areas.

Exposure
Used to control the image brightness (in stops) applied before color space conversion.

Gamma
Extra gamma correction applied after color space conversion. Note that the default sRGB or Rec709 color space conversions already include a gamma correction of approximately 2.2 (except the Raw and Log views), so this would be applied in addition to that.

Look
Choose an artistic effect from set of measured film response data which roughly emulates the look of certain film types. Applied before color space conversion.

Use Curves
Adjust RGB Curves to control image colors before color space conversion. Read more about using the Curve Widget.

Sequencer

Color Space
The color space that the sequencer operates in. By default the sequencer operates in sRGB space, but it can also be set to work in Linear space like the Compositing nodes, or another color space. Different color spaces will give different results for color correction, cross fades, and other operations.

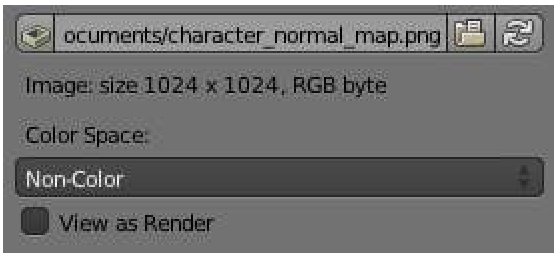
Image Files

The other place to keep color management in mind is when **loading and saving image files**. File formats such as PNG or JPEG will typically store colors in a color space ready for display, not in a linear space. When they are, for example, used as textures in renders, they need to be converted to linear first, and when saving renders for display on the web, they also need to be converted to a display space. Other file formats like OpenEXR store linear color spaces and as such are useful as intermediate files in production.

When working with image files, the default color space is usually the right one. If this is not the case, the color space of the image file can be configured in the image settings. A common situation where manual changes are needed is when working with or baking normal maps or displacement maps, for example. Such maps do not actually store colors, just data encoded as colors. In such cases they should be marked as **Non-Color Data**.

Image data-blocks will always store float buffers in memory in the scene linear color space, while a byte buffer in memory and files on disk are stored in the color space specified with this setting:

Color Space
The color space of the image on disk. This depends on the file format, for example PNG or JPEG images are often stored in sRGB, while OpenEXR images are stored in a linear color space. Some images such as normal, bump or stencil maps do not strictly contain 'colors', and on such values no color space conversion should ever be applied. For such images the color space should be set to None.



ocuments/character_normal_map.png

Image: size 1024 x 1024, RGB byte

Color Space:

Non-Color

□ View as Render

Image settings for color management

By default only renders are displayed and saved with the render view transformations applied. These are the Render Result and Viewer image data-blocks, and the files saved directly to disk with the Render Animation operator. However when loading a render saved to an intermediate OpenEXR file, Blender can't detect automatically that this is a render (it could be e.g. an image texture or displacement map). We need to specify that this is a render and that we want the transformations applied, with these two settings:

View as Render
Display the image data-block (not only renders) with view transform, exposure, gamma, RGB curves applied. Useful for viewing rendered frames in linear OpenEXR files the same as when rendering them directly.

Save as Render
Option in the image save operator to apply the view transform, exposure, gamma, RGB curves. This is useful for saving linear OpenEXR to e.g. PNG or JPEG files in display space.

OpenColorIO Configuration

Blender comes with a standard OpenColorIO configuration that contains a number of useful display devices and view transforms. The reference linear color space used is the linear color space with Rec. 709 chromaticities and D65 white point.

However OpenColorIO was also designed to give a consistent user experience across multiple applications, and for this a single shared configuration file can be used. Blender will use the standard OCIO environment variable to read an OpenColorIO configuration other than the default Blender one. More information about how to set up such a workflow can be found on the OpenColorIO website.

We currently use the following color space roles:

scene_linear
color space used for rendering, compositing, and storing all float precision images in memory.

default_sequencer
default color space for sequencer, *scene_linear* if not specified

default_byte
default color space for byte precision images and files, *texture_paint* if not specified.

default_float

default color space for float precision images and files, *scene_linear* if not specified.

The standard Blender configuration also includes some support for ACES (code and documentation), even though we have a different linear color space. It’s possible to load and save EXR files with the Linear ACES color space, and the RRT view transform can be used to view images with their standard display transform. However the ACES gamut is larger than the Rec. 709 gamut, so for best results an ACES specific configuration file should be used. OpenColorIO provides an ACES configuration, though it may need a few more tweaks to be usable in production.

Compatibility

Compatibility with existing files should mostly be preserved. Files that had color management enabled should be entirely compatible, while older files with the color management option disabled are mostly compatible but different for vertex colors and viewport colors.