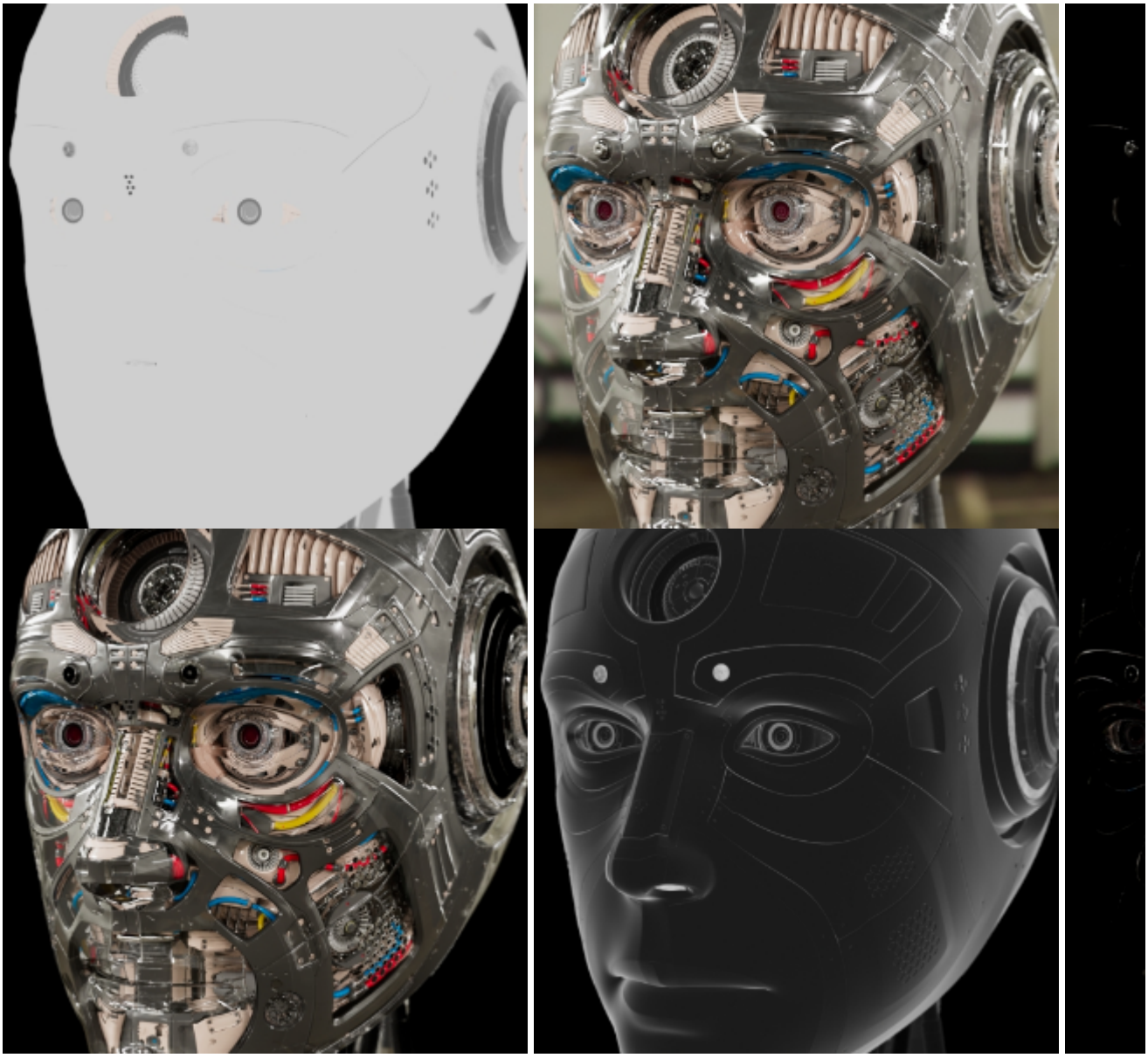
AOVs



AOVs (Arbitrary Output Variables) provide a way to render any arbitrary shading network component into different images. For example, an artist might find it convenient to separate direct and indirect lighting contributions and later recombine them during compositing. Arnold provides built-in AOVs for outputting depth, position, and motion vectors.

There are some limitations:

- Using closures as AOVs is not currently supported.

- Light path expressions can be used to output light into specific AOVs.
- HtoA supports Cryptomatte AOV shaders.
- An introduction to AOVs for compositing tutorial can be found here.

Per-light AOVs

AOV Shaders

AOVs

Light Path Expressions

Per-light AOVs

It is possible to view AOVs while rendering in MPlay, Render View (right click and enable the **View Bar** to see the AOV chooser) and Render Region (right click and select the AOV from the **Image Plane** menu).

A set of AOV shaders exists to allow easy reading and writing of AOVs from within a shader network.

Enable AOV Composition is now always enabled and the option has been removed.

AOV Shaders

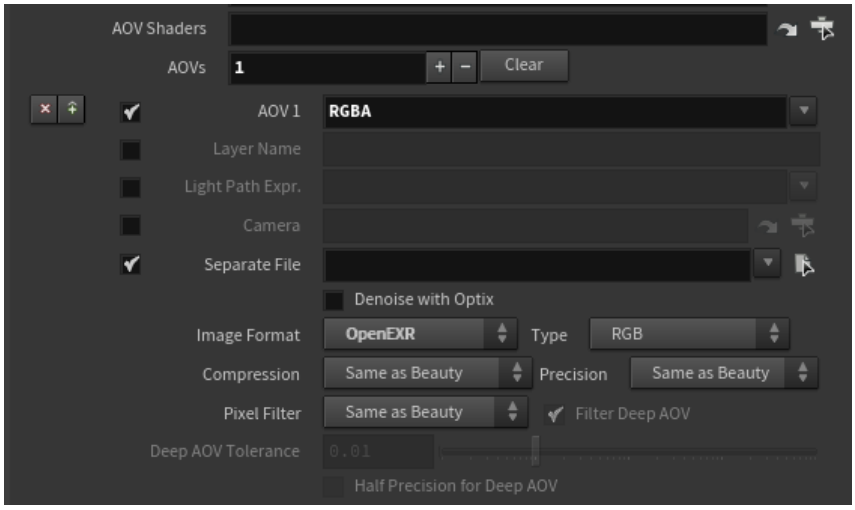
A list of shaders can be defined that will be evaluated after the regular surface shader. With this, it's possible to add shaders to set specific AOVs without modifying the original shader tree. Shaders intended for this purpose should add a booleanmetadata named `aov_shader` on the node itself, as a user-interface hint. If `options.atmosphere` or `options.background` are set, these global AOV shaders will also be run for atmosphere and background contexts.

Only shaders with AOV-writing capabilities should be added to 'AOV Shaders' in AOVs.

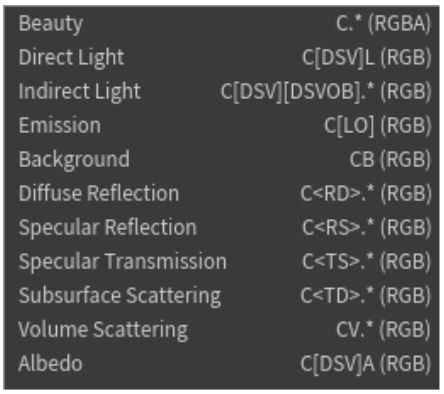
AOV shaders such as *Cryptomatte AOVs* can also be added to 'AOV Shaders' in AOVs.

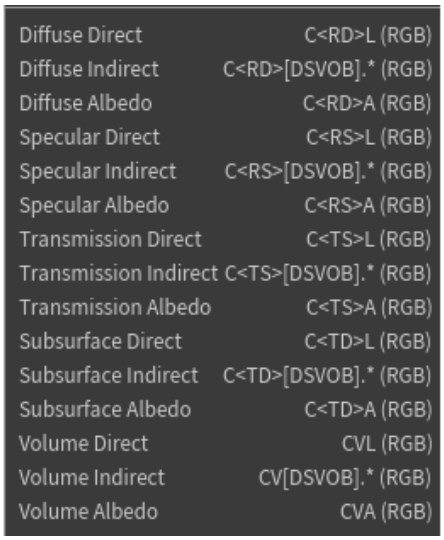
AOVs

- To add an AOV click on the + button.



- To choose the type of AOV click on the down arrow to the right of AOV 1. You will see a list of available AOVs (with their associated LPE).





Arnold provides the following 'built-in' system AOVs. These AOVs are always available, no matter what shader(s) you are using.

- **A:** Alpha.
- **AA_inv_density:** Visualizes the sample density with *Adaptive Sampling.* Use it with a *Heatmap* filter.
- **ID:** Random number value derived from the name of the shape. You can also add specific ID numbers via the user options string field for an object. ie 'id 1'.
- **N:** Smooth normal at the shading point (in world space).
- **P:** Position of the shading point (in world space).
- **Pref:** Reference position of the shading point.
- **RGBA:** Beauty AOV, containing the full rendered image**.**
- **Z:** Depth of the shading points as seen from the camera.
- **albedo:** Reflectivity, the surface or volume color without lighting or shadowing.
- **background:** Emission from the background and skydome lights visible to the camera.
- **coat:** Coat reflection.
- **coat_albedo:** Coat color without lighting or shadowing.
- **coat_direct:** Coat direct lighting.
- **coat_indirect:** Coat indirect lighting.
- **cputime:** This layer contains the CPU time (measured in “ticks”) to evaluate the samples in the pixel.
- **diffuse:** Diffuse reflection.
- **diffuse_albedo:** Diffuse color without lighting or shadowing.

- **diffuse_direct:** Diffuse direct lighting.
- **diffuse_indirect:** Diffuse indirect light.
- **direct:** Direct lighting from all surfaces and volumes.
- **emission:** Lights and emissive objects directly visible from the camera.
- **indirect:** Indirect light from all surfaces and volumes.
- **motionvector:** 2D vector representing the motion in screen space of the shading point during the given time interval. If output to an RGB format, the vector is contained in the R and G channels.
  You must set an instantaneous shutter for the camera. The reason being is that we don't want motion blur in the render, but we still want the motion velocity information in our motion vector AOV. This can be found under *Motion Blur-> Instantaneous Shutter.*
- **opacity:** RGB AOV with full three-channel opacity (as opposed to single channel alpha).
- **raycount:** Total number of rays traced for samples in the pixel.
- **shadow_matte:** Shadows in the scene, computed as the ratio of occluded direct lighting over unoccluded direct lighting.
- **sheen:** Sheen weight.
- **sheen_albedo:** Sheen color without lighting or shadowing.
- **sheen_direct:** Sheen direct lighting.
- **sheen_indirect:** Sheen indirect lighting.
- **specular:** Specular reflection.
- **specular_albedo:** Specular color without lighting or shadowing.
- **specular_direct:** Diffuse direct lighting.
- **specular_indirect:** Diffuse indirect lighting.
- **sss:** Subsurface scattering and diffuse transmission.
- **sss_albedo:** SSS and diffuse transmission color without lighting or shadowing.
- **sss_direct:** SSS and diffuse transmission direct lighting.
- **sss_indirect:** SSS and diffuse transmission indirect lighting.
- **transmission:** Specular transmission (refraction).
- **transmission_albedo:** Specular transmission color without lighting or shadowing.
- **transmission_direct:** Specular transmission direct lighting.
- **transmission_indirect:** Specular transmission of indirect lighting.
- **volume:** Volume scattering.
- **volume_z:** The Z depth for the first volume contribution is output in a flat AOV.
- **volume_albedo:** Volume color without lighting or shadowing.
- **volume direct:** Volume scatter direct lighting.
- **volume indirect:** Volume scattering indirect lighting.
- **volume opacity:** RGB AOV with the full three-channel opacity for volumes only.

The other AOV groups correspond to the shader nodes being used (assuming those shader nodes support AOV). For example, *Shadow Matte* provides:

- **shadow:** Direct light shadow.
- **shadow diff:** A difference AOV which can be used to eliminate the shadow from the direct component.
- **shadow mask:** This AOV can be used in comp to localize and tweak the shadow.

Other shaders used in your scene will support various other AOVs. Multiple shaders can contribute to the same AOV (for example a *Standard Surface* and a *Lambert* shader both write to the **diffuse_direct** AOV).

Composing the Beauty AOV

The RGBA beauty AOV can be split into smaller AOVs where each contains part of the lighting. In compositing, these AOVs can then be individually modified and added together to get the full beauty AOV.

More AOVs give more control in compositing, but also extra work to handle, and they take up more memory and disk space, especially combined with light groups.

Some example sets of additive AOVs for the full beauty AOV are:

- direct, indirect, emission, background.
- diffuse, specular, coat, transmission, sss, volume, emission, background.
- diffuse_direct, diffuse_indirect, specular_direct, specular_indirect, coat, transmission, sss, volume, emission, background.

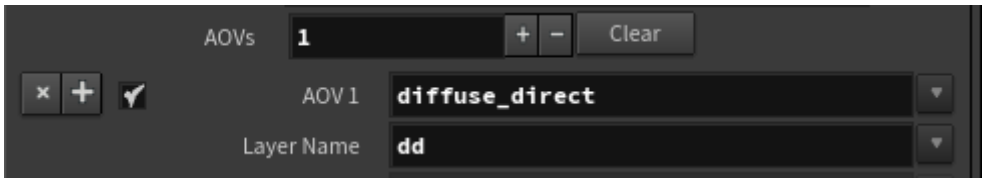
Simply adding together such AOVs is all that is needed for the beauty AOV. The albedo AOVs are not needed to reconstruct the beauty AOV but may be used for example to get just the lighting without the surface texture, by dividing diffuse by diffuse_albedo, or for denoising just the lighting while keeping the texture detail intact.

A tutorial about compositing AOVs can be found here.

Layer Name

Output layer names can be customized in file formats that support layers (regular or deep EXR) by adding an optional custom layer name after the driver name in the `options.outputs` string. E.g.

```
outputs 2 1 STRING
 "RGBA RGBA gaussian_filter driver_exr beauty"
 "RGBA RGBA variance_filter driver_exr beauty_variance"
```



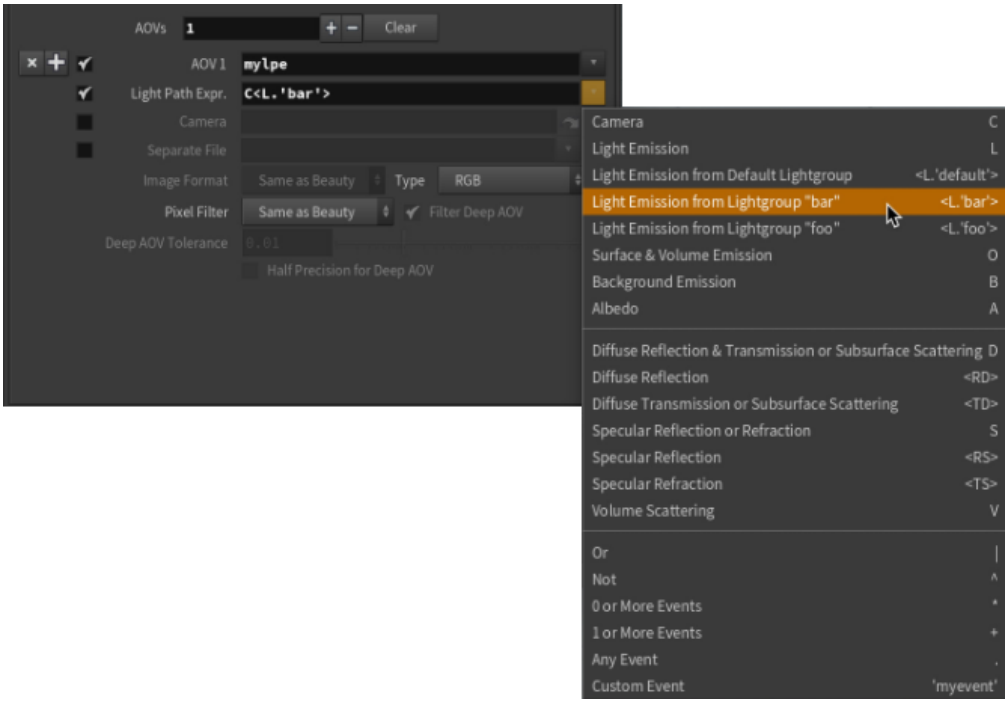
When using AOV wildcard syntax and specifying a name for an output layer, the AOV type will be replaced by the given name.

For example, using a wildcard of *diffuse_***, a layer name of *myLayer*, and AOVs of 'red' & 'blue', the layers written into the EXR will be named *myLayer_red* and *myLayer_blue* instead of the default *diffuse_red* and *diffuse_blue*.

Output layers in files that support layers (such as regular or deep EXR files) can be individually set to type HALF by adding an optional HALF modifier to the corresponding output string. For instance: "my_aov RGB filter driver HALF".

Light Path Expressions

You can define custom Light Path Expressions to write lighting components into separate AOVs.



LPE menu with existing light groups

Separate File

Render the AOV in a separate file, as opposed to storing it in as an additional channel of the beauty image.

Denoise

For use with the *Optix* denoiser (System tab). This filter can be applied to any AOV just like a normal filter.

Image Format

Specifies the image format of the AOV if it is being rendered to a separate file. See the Image Format section for more information. By default, this will be the same as the beauty.

Type

The type of data to be written in the AOV - RGBA, float, vector, etc. At the moment Arnold for Houdini will not specify a default type for each AOV so you will have to explicitly set this. This will be updated in a future release. Refer to the list below for what type to use for each AOV.

- Depth [z] - float
- Point [P] - point
- Nomal [N] - vector
- Object Labels [ID] - rgb
- Opacity - rgb
- CPU Time - float
- Ray Count - float
- Shader Layers - rgb

Precision

Specify whether 16-bit floating point (binary16) or full 32-bit precision is used. By default, this will be the same as the beauty.

Compression

Allows you to set multiple EXR compressions per AOV.

Pixel Filter

Specifies the pixel filter used on the AOV, see the Pixel Filter section for more information. By default, this will be the same as the beauty.

Filter Deep AOV

If set to false disables any filtering operation on this layer's raw data. Useful for normals or ID layers.

Deep AOV Tolerance

Tolerance over which the AOV samples will not be merged together.

Half Precision for Deep AOV

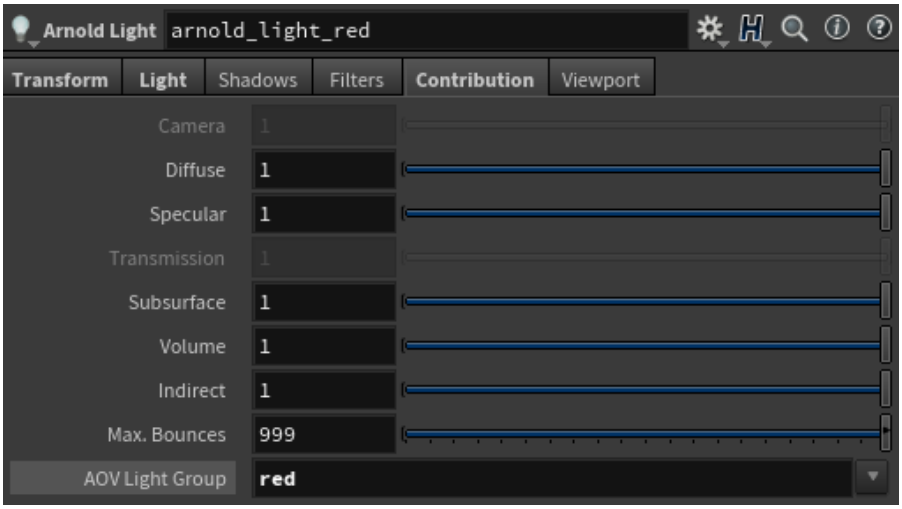
Use 16-bits float for this AOV filtering.

The Mix shader only works with AOVs of type RGB. If an AOV exists but is not specified on the AOVs tab, some limited mixing will occur, without a smooth transition: if the mix value is 0 or less, input1 will be output, else input2. Leaving an AOV name blank will disable mixing.

Per-light AOVs

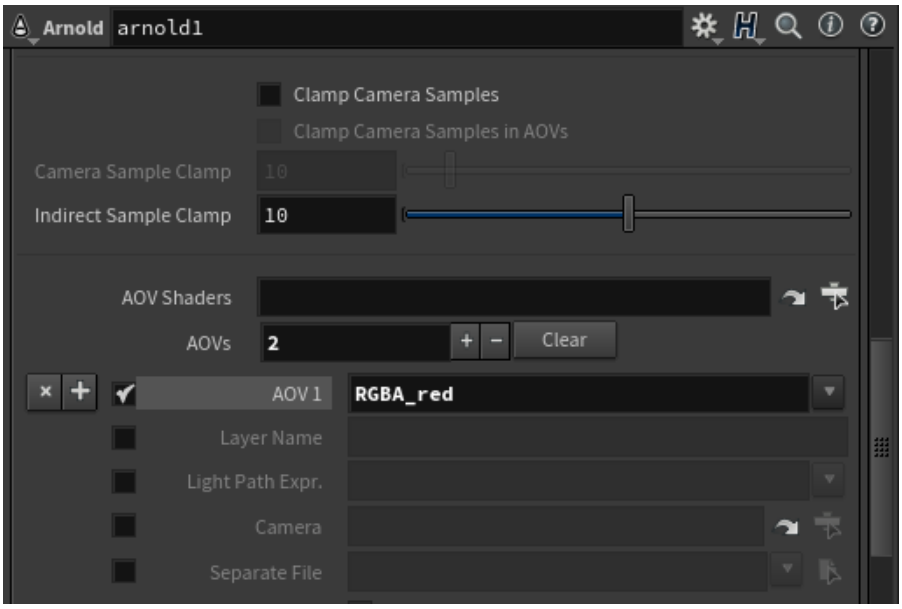
Each light object has an 'AOV Light Group' attribute which can be used to write out the light contribution to a separate AOV with a corresponding name. To create a per-light AOV you must do the following:

1. Enter a name for the per-light AOV in the 'AOV Light Group' of the light.



'Red' entered in AOV Light Group

2. Add a custom AOV in the Render Output. For example, if the light group name is “red”, then the AOV name should be **`RGBA_red`** (if you want the red light contribution to the beauty). Or **`diffuse_red`** if you want just the red light contribution to the diffuse.



RGBA_red custom AOV in Arnold Output

Light path expressions can be used to output light into specific AOVs.