Light Path Expression AOVs

For Arnold 5.0 shaders using closures, light path expressions are used to output light into specific AOVs. Rather than writing to specific AOVs from shaders, LPEs can be used to extract specific light paths using regular expressions.

There is also an Introduction to Light Path Expressions tutorial.

Builtin LPEs

The following light path expression AOVs are built-in.

Name	Expression	
RGBA	C.*	Beauty AOV with all light paths
direct	C[DSV]L	Direct light from all surfaces and volumes
indirect	C[DSV][DSVOB].*	Indirect light from all surfaces and volumes
emission	C[LO]	Lights and emissive objects directly visible from the camera
background	CB	Background emission
diffuse	C<RD>.*	Diffuse reflection
specular	C<RS[^'coat']>.*	Specular reflection
coat	C<RS'coat'>.*	Coat reflection
transmission	C<TS>.*	Specular transmission
sss	C<TD>.*	Subsurface scattering and diffuse transmission
volume	CV.*	Volume scattering
albedo	C[DSV]A	Albedo (reflectivity)
denoise_albedo	((C<TD>A)|(CVA)|(C<RD>A))	Denoise albedo
diffuse_direct	C<RD>L	Diffuse direct light
diffuse_indirect	C<RD>[DSVOB].*	Diffuse indirect light
diffuse_albedo	C<RD>A	Diffuse albedo
specular_direct	C<RS[^'coat']>L	Specular direct light
specular_indirect	C<RS[^'coat']>[DSVOB].*	Specular indirect light
specular_albedo	C<RS[^'coat']>A	Specular albedo
coat_direct	C<RS'coat'>L	Coat direct light
coat_indirect	C<RS'coat'>[DSVOB].*	Coat indirect light
coat_albedo	C<RS'coat'>A	Coat albedo
sheen	C<RS'sheen'>.*	Sheen
transmission	C<TS>.*	Transmission
transmission_direct	C<TS>L	Transmitted/refracted direct light
transmission_indirect	C<TS>[DSVOB].*	Transmitted/refracted indirect light
transmission_albedo	C<TS>A	Transmission albedo
sheen_direct	C<RS'sheen'>L	Sheen direct
volume	CV.*	volume
sheen_indirect	C<RS'sheen'>[DSVOB].*	Sheen indirect
sheen_albedo	C<RS'sheen'>A	Sheen albedo
sss	C<TD>.*	SSS
sss_direct	C<TD>L	SSS and diffuse transmission direct light
sss_indirect	C<TD>[DSVOB].*	SSS and diffuse transmission indirect light
sss_albedo	C<TD>A	SSS and diffuse transmission albedo
volume_direct	CVL	Volume direct light
volume_indirect	CV[DSVOB].*	Volume indirect light
volume_albedo	CVA	Volume albedo

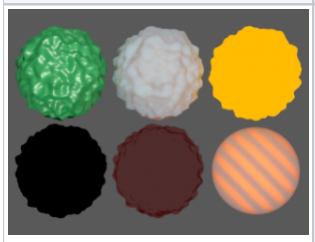
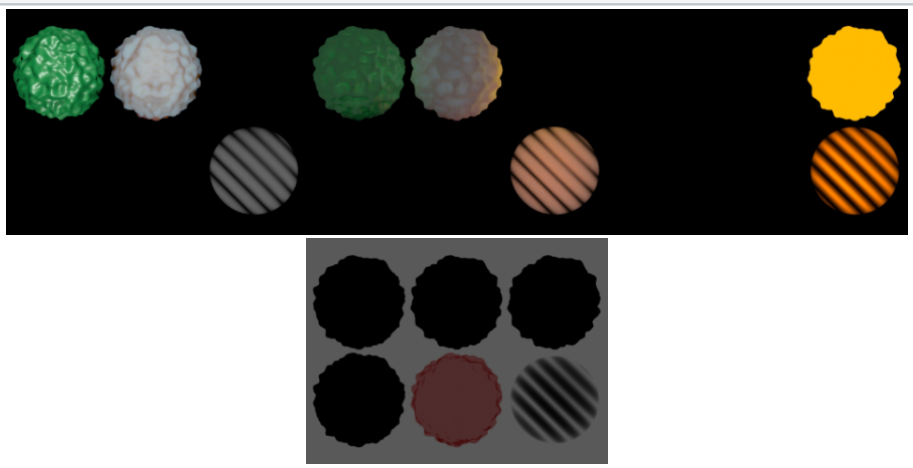
Composing the Beauty AOV

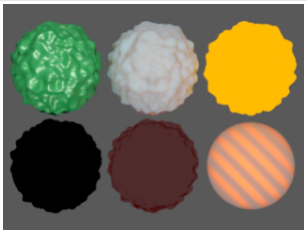
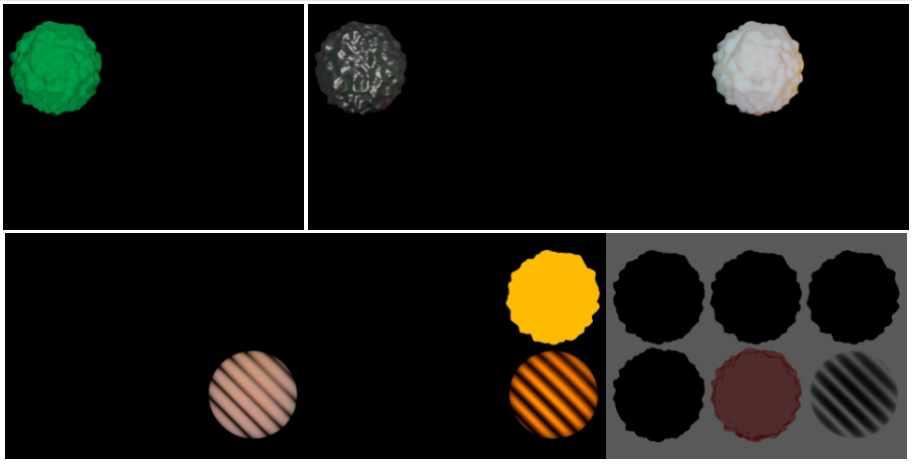
The RGBA beauty AOV can be split into smaller AOVs where each contains part of the lighting. In compositing, these AOVs can then be individually modified and added together to get the full beauty AOV.

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.

RGBA	direct + indirect + emission + background
	

RGBA	diffuse + specular + sss + volume + emission + background
	

Syntax

All expressions must start with a camera event `C`. After that, there can be zero or more scattering events, until a light, object or background is hit. The syntax is similar to regular expressions. The following events and operators are supported.

Events

Symbol	
C	Camera
L	Light emission from all lights
<L.'groupname'>	Light emission from light in AOV groupname
<L.'default'>	Light emission from lights with no AOV group assigned
O	Surface or volume object emission
B	Background emission
A	Albedo

Scattering Events

Symbol	
D	Diffuse reflection, transmission or subsurface scattering
<RD>	Diffuse reflection
<TD>	Diffuse transmission or subsurface scattering
S	Specular reflection or refraction
<RS>	Specular reflection
<TS>	Specular refraction
V	Volume scattering

Operators

Expression	
AB	A followed by B
[ABC]	A, B, or C
((AB)|(CD))	AB or CD
[^A]	Any event except A
A*	0 or more A events
A+	1 or more A events
.	Any event
'myevent'	Custom event

Custom Expressions

Custom light path expressions are added to `options.light_path_expressions`, and can then be used as AOVs in `options.outputs`.

Custom expressions can override built-in expressions with the same name if different behavior is desired.

```
light_path_expressions 2 1 STRING
        "caustics CDS.*"
        "diffuse_albedo CDA"
outputs 3 1 STRING
        "RGBA RGBA filter driver"
        "caustics RGB filter driver"
        "diffuse_albedo RGB filter driver"
```

Light Groups

Besides the `<L.'groupname'>` syntax, any built-in or custom LPE can be split into multiple AOVs to output a subset of lights with a specific AOV group assigned in the `light.aov` parameter. For this, a postfix must be added to the LPE AOV name in `options.outputs`.

```
outputs "diffuse RGB filter driver"            # all lights in one AOV
outputs "diffuse_* RGB filter driver"          # output multiple AOVs, one for each light group
outputs "diffuse_groupname RGB filter driver"  # only light in AOV group "groupname"
outputs "diffuse_default RGB filter driver"    # only lights with no AOV group assigned
```

Custom Labels

BSDF lobes, BSSRDFs, and emission closures may be tagged with a custom label to output them to separate AOVs.

For built-in shaders standard hair is labeled with 'hair', and the standard surface coat is labeled with 'coat'. All other built-in shaders components have no labels. So the following expressions could be used for example:

```
light_path_expressions 2 1 STRING
        "hair C'hair'.*"
    "specular_except_coat C<RS[^'coat']>.*"
```

Custom shaders could for example label BSSRDFs and emission, and then use expressions like these:

```
light_path_expressions 2 1 STRING
        "sss_skin C<TD'skin'>.*"
    "emission_fire C<O.'fire'>"
```