Light Path Expressions

Quick Introduction
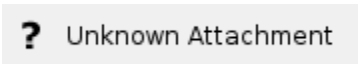
In RenderMan, in addition to the built-in AOVs, light path expressions (LPEs), which are adopted from Open Shading Language's Light Path Expression, specify what light transport paths to output to a display channel. In other words, we don't have to modify the shaders or plugins to output a custom AOV that uses the LPE.

The setup is the same as an AOV except we use a light path expression instead of an AOV channel name. Additionally, we can group the outputs by light groups and/or by geometry set up by a lpegroup.

LPE Tokens

To better understand how an LPE is constructed, the below tokens are used to define interactions. For most users, the pre-built LPE are easily selected and explained below.

LPE Events	Description	Example
`C`	Camera	All LPEs must contain a camera event.
`<type event 'group'>`	Scattering event specified by *scattering type*, *scattering event*, or lpe group	`C<RD'floor'>[LO]` matches diffuse reflections from the *floor* lpe group
`<L.'group'>`	Light, i.e. analytic and geometric lights	`C.*<L.'key'>` matches all light paths from the *key* light group
`O`	Emissive object, i.e. camera visible lights and material glow	`C.*O` matches all emissive object light paths
Wildcard		
`.`	A period will match any relevant scattering type or event	`C.L` which is the same as `C<..>L` or `C<[RT][DS]>L`
Scattering Types		
`R`	Reflection	`CR[LO]` which is the same as `C<R.>[LO]` or `C<R[DS]>[LO]` matches direct reflection
`T`	Transmission or refraction	`CT[LO]` which is the same as `C<T.>[LO]` or `C<T[DS]>[LO]` matches direct transmission
Scattering Events		
`D`	Diffuse lobes (`D1`, `D2`, `D3`, `D4`)	`CD[LO]` which is the same as `C<.D>[LO]` or `C<[RT]D>[LO]` matches direct diffuse.

S	Specular/Glossy lobes (`S1`, `S2`, `S3`, `S4`, `S5`, `S6`, `S7`, `S8`)	`CS[LO]` which is the same as `C<.S>[LO]` or `C<[RT]S>[LO]` matches direct specular
User Event		
U	User lobes to output material properties, not light transport (`U1` to `U12`) *Note: User events must be explicit and will not match wildcards*	`CU2L` matches the camera visible user2 lobe, typically albedo
Grouping		
( )	Parentheses allow grouping of a sequence of LPE events	`CDSL`, `C(DS)L`, and `CD(SL)` are all functionally equivalent
Alternatives		
|	A vertical bar separates alternatives	`CD|(SD)L` matches `CDL` or `CSDL`
[]	Square brackets that match any one of the enclosed events	`C[S2 S3 S4]L` matches `CS2L`, `CS3L`, or `CS4L`
[^]	Square brackets that match any one event not contained within the brackets	`C[^D2D3]L` matches all light paths besides diffuse2 and diffuse3 (including specular)
Qualifiers		
*	An asterisk will match the preceding event zero or more times	`C.*[LO]` matches all light paths, including emissive
+	A plus sign will match the preceding event one or more times	`C.+[LO]` matches direct and indirect light paths, but not emissive
?	A question mark will match the preceding event zero or one times	`C.?[LO]` matches direct and emissive light paths
{n}	Curly brackets that match the preceding event exactly *n* times	`CT{2}[LO]` matches double transmission light paths
{min,}	Curly brackets that match the preceding event *min* or more times	`C.{2,}[LO]` matches all indirect light paths
{min,max}	Curly brackets that match the preceding event at least *min* but not more than *max* times	`C.{0,2}[LO]` matches all light paths with up to one bounce of indirect

In RIB

To specify an LPE in RIB, reference a DisplayChannel in the Display. This is the most basic way to view what's being output.

```
DisplayChannel "color directDiffuse" "string source" ["color lpe:CD[<L.>O]"]
Display "myRender.exr" "openexr" "directDiffuse"
```

Usage

LPE tokens are not necessarily the most artist-friendly things. For this reason, we provide a list of LPEs in our plugins that can be easily selected and output. We also provide options on how specific you want to be when it comes to output; from outputting all your direct or indirect interactions at once per type or outputting them per-lobe which provides greater control for advanced workflows.

Basic LPEs

For basic workflows, the following LPE are recommended. Note that we are specifying the DisplayChannel and a name (which can be most anything). Note that these LPE will collect the light interaction from different lobes of materials and store them in the same AOV. For example: reflections from the Glass Lobe in PxrSurface will be combined with other specular reflections in the direct and indirectSpecular AOVs. This can simplify the outputs.

> ⚠ Understand that emissive LPE results are a special case. In the materials provided it's referred to as "Glow" or emission. Light sources directly visible to the camera are also output into this LPE. In the below example, the candle flames are using the glow lobe in PxrSurface.

	DisplayChannel	**Expression**

? Unknown Attachment	`color directDiffuse`	`color lpe:C<RD>[<L.>O]`
? Unknown Attachment	`color indirectDiffuse`	`color lpe:C<RD>.+[<L.>O]`
? Unknown Attachment	`color subsurface`	`color lpe:C<TD>.*[<L.>O]`
? Unknown Attachment	`color directSpecular`	`color lpe:C<RS>[<L.>O]`
? Unknown Attachment	`color indirectSpecular`	`color lpe:C<RS>.+[<L.>O]`
? Unknown Attachment	`color transmissive`	`color lpe:C<TS>.*[<L.>O]`
? Unknown Attachment	`color emissive`	`color lpe:C[<L.>O]`

? Unknown Attachment

RenderMan Still Life by Dylan Sisson

Per-Lobe LPEs

Bxdf materials may have more than one diffuse or specular lobe which are summed for `D` and `S`, respectively. However, in some situations, it may be desirable to output a specific lobe separately.

For example, some PxrSurface usage may involve the Clearcoat lobe. Normally this is summed under the `S` token. Routing this lobe to the `S4` token will allow you to use `S4` in your LPEs. These name-to-lobe mappings are controlled by render options. Up to 4 diffuse lobes and 8 specular lobes are available for LPEs. By default, `S1` contains all unspecified specular lobes, so use any higher tokens such as `S2` for explicit per-lobe LPEs. The standard lobe mappings are:

LPE Lobes	Named Lobes
D1	(default diffuse)
D2	Diffuse,HairDiffuse
D3	Subsurface
D4	
S1	(default specular)
S2	Specular,HairSpecularR
S3	RoughSpecular,HairSpecularTRT
S4	Clearcoat
S5	Iridescence
S6	Fuzz,HairSpecularGLINTS
S7	SingleScatter,HairSpecularTT
S8	Glass
U1	(default user)
U2	Albedo,DiffuseAlbedo,SubsurfaceAlbedo,HairAlbedo
U3	Position

U4	UserColor
U5	
U6	Normal,DiffuseNormal,HairTangent,SubsurfaceNormal,SpecularNormal,RoughSpecularNormal,SingleScatterNormal,FuzzNormal, IridescenceNormal,GlassNormal
U7	
U8	

For advanced workflows that want to separate all scattering lobes into separate AOVs, the below LPE are provided. These capture the light interaction for each lobe, direct and indirect, and store them into separate AOVs. This is useful for tweaking individual effects at the cost of added AOVs and image output. The documentation for PxrMarschner and PxrSurface list these user lobes at the bottom of their page under an Advanced Rollout as it is for informational purposes and not actual operation.

> ⚠ Note that the emissive LPE is repeated below. For materials in RenderMan it's typically called Glow.

	DisplayChannel	Expression
Unknown Attachment	`color directDiffuseLobe`	`color lpe:CD2[<L.>O]`
Unknown Attachment	`color indirectDiffuseLobe`	`color lpe:CD2.+[<L.>O]`
Unknown Attachment	`color subsurfaceLobe`	`color lpe:CD3.*[<L.>O]`
Unknown Attachment	`color directSpecularPrimaryLobe`	`color lpe:CS2[<L.>O]`
Unknown Attachment	`color indirectSpecularPrimaryLobe`	`color lpe:CS2.+[<L.>O]`
Unknown Attachment	`color directSpecularRoughLobe`	`color lpe:CS3[<L.>O]`
Unknown Attachment	`color indirectSpecularRoughLobe`	`color lpe:CS3.+[<L.>O]`
Unknown Attachment	`color directSpecularClearcoatLobe`	`color lpe:CS4[<L.>O]`
Unknown Attachment	`color indirectSpecularClearcoatLobe`	`color lpe:CS4.+[<L.>O]`
Unknown Attachment	`color directSpecularIridescenceLobe`	`color lpe:CS5[<L.>O]`
Unknown Attachment	`color indirectSpecularIridescenceLobe`	`color lpe:CS5.+[<L.>O]`
Unknown Attachment	`color directSpecularFuzzLobe`	`color lpe:CS6[<L.>O]`
Unknown Attachment	`color indirectSpecularFuzzLobe`	`color lpe:CS6.+[<L.>O]`
Unknown Attachment	`color transmissiveSingleScatterLobe`	`color lpe:CS7.*[<L.>O]`

	`color directSpecularGlassLobe`	`color lpe:C<RS8>[<L.>O]`
	`color indirectSpecularGlassLobe`	`color lpe:C<RS8>.+[<L.>O]`
	`color transmissiveGlassLobe`	`color lpe:C<TS8>.*[<L.>O]`
	`color emissive`	`color lpe:C[<L.>O]`

Caustics are not included in the built-in list because they overlap with `indirectdiffuse lpe:C<RD>[DS]+[<L.>O]` However, you can select caustics with the expression `lpe:CDS+[<L.>O]` Below you can find an example of the caustics LPE output, Indirect Diffuse, and the difference between them. Note that the Caustics LPE output doesn't capture indirect lighting from object to object, in this case the red cube. You can create the image on the far right with the indirect diffuse minus the caustic result using the following source: `lpe:C<RD>D[DS]*[<L.>O]`

LPE Macro

If a custom LPE is useful for your shots, you can add your custom LPE by creating a macro in `rendermn.ini` file. The custom LPE will appear in the RenderMan for Maya (RfM) LPE list after relaunching Maya. For example:

```
/prman/lpe/macro/myDiffuse    CD<L.>
```

⚠️
- It is important to use `<L.>` instead of only `L` so that it will be compatible with light groups. Otherwise, the renderer will output a warning.
- The `lpe:` prefix will be automatically added to your macro's name when the file is parsed.

LPE Groups

An LPE group (lpegroup) allows us to specify which objects we want to use for an LPE channel. The attributes can be added in RenderMan plugins using the software's menu system such as PrmanObjectSettings in Katana.

LPE Group Example RIB

```
Attribute "identifier" "string lpegroup" ["ground"]
DisplayChannel "varying color plane2Shadow" "string source" ["color lpe:shadows;C<..'ground'><L.>"]
```

Predefined LPE Groups

For the built-in LPE such as shadowcollector, it assumes a predefined lpegroup named "collector". So for the objects that we want to collect shadow, we can simply name its lpegroup to "collector". Specifying `lpe:shadowcollector` will collect the shadow for these objects.

This may be useful for creating holdouts using a manual workflow.

Advanced LPE Group Logic

You can use the usual regular expression character classes syntax to define more complex LPEs by considering each lpegroup as a single token.

For instance, provided you are using lpegroups "foo" and "bar", you can define the following LPEs:

```
C<.D'foo'>L
C<.D[^'foo']>L
C<.D['foo''bar']>L
C<.D[^'foo''bar']>L
```

Light Groups

By default, LPEs retrieve the response to all lights in the scene. To limit the LPE to the contribution due to a single light or set of lights, assign each light to the group by setting the __group parameter on each light to the same name.

For the short LPE name, suffix the name with '_' followed by the light group name. For the long expression, place the light group name in single quotes inside the LPE.

For instance, say we have a light group named "key". The following are both valid LPEs:

```
DisplayChannel "varying color lpe:diffuse_key"
DisplayChannel "varying color lpe:CD<L.'key'>"
```

LPE Prefixes

- `unoccluded` – returns unoccluded or unshadowed result.
- `noclamp` – returns unclamped result.
- `nothruput` – does not apply thruput (thruput is the accumulative albedo of the objects hit by rays).
- `shadows` – returns collected shadows.
- `holdouts` – returns only holdout light paths (light paths with one or more holdout events)
- `overwrite` – instead of outputting the accumulated result, overwrite it. One example of using this is for the albedo output where we do not want an accumulated result.
- `noinfinitecheck` – do not do any infinite check.

User Defined Signals

- `U` tokens specify a user defined signal. A Bxdf can output any user defined signal via the RixBXLobeWeights class. There are twelve of them, from `U1` to `U12`. By default, all user defined signals are set to `U1`. More on this can be found in the Developer's Guide.
- If you have more than one user defined signal, it is important to direct each user defined signal to a different `U` token by setting them in `rendermn.ini`. Otherwise, all user defined signals that are assigned to the same U token will override each others!
- By default, `U2` is set to the albedo output in the installed `rendermn.ini`:

```
/prman/lpe/user2              Albedo,DiffuseAlbedo,SubsurfaceAlbedo,HairAlbedo
```

Examples

- In the documentation examples we often use <L.> in place of L. There's no functional difference.
- lpe:CD<L.> means camera with diffuse scattering event for light scattering type.
- lpe:CO means camera with emissive objection (since emissive does not need any light, there is no L).
- Using . in place of DS is shorthand, if you are using a User Lobe, you must specify U and cannot use the . shorthand