Refraction And Opacity

Refraction and opacity are pretty similar in Arnold compared with other render engines, however, there are a few differences. The purpose of this document is to give users a better understanding of these differences as well as explaining when to use opacity and when to use refraction.

Arnold has two different ways of calculating transparency, refraction and opacity. They are different ray types and thus have different controls in the Standard shader as well as in the render options.

These two ways of calculating transparency have different purposes, they can be used together but most of the time you'll want to use either one or the other.

Usage for Refraction

- Glass, water or other refractive materials.

Usage for Opacity

- Sprite type of effect, such as cutting out the shape of a leaf from a polygon card.
- Making the tips of hair strands transparent.

If you leave Index of Refraction at 1.0 both methods can give similar results, however Opacity or "Auto Transparency" as it's called in the render options renders faster, for sprites. Opacity will also cut out the shape completely whereas Refraction will leave the reflections/speculars visible even on areas that are completely transparent, here are two images to show the difference when rendering sprites:

Leaf mask driving the opacity

Leaf mask driving the refraction

This is the texture and mask used for the leaf:

Leaf texture map

Leaf alpha map

Note how when using refraction the specular/reflections are still visible in the transparent areas. You can of course drive the specular weight using the same mask to fix the problem, but using refraction for this purpose is simply wrong.

Using Refraction And Opacity Together

As explained earlier it's usually best not to use Refraction and Opacity together in the same shader as it would cause unnecessary slowdowns in the render; however there are a few situations where combining them can be useful. Below are some examples of a simple glass sphere being cut out using a stripe mask:

Sphere using refraction only

Sphere rendered with the mask texture

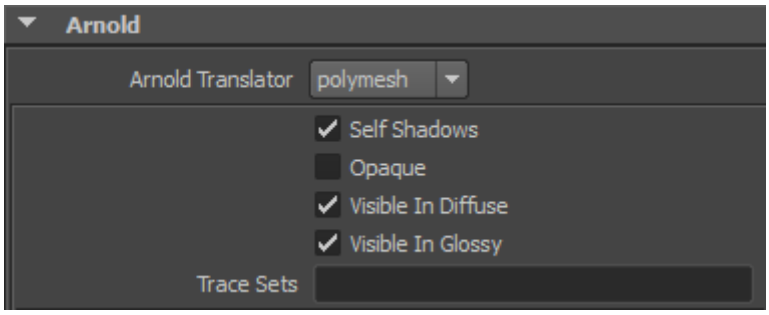
Spher

The Opaque Switch

This is where Arnold differs from most other renderers. By default all objects are flagged as 'Opaque', which allows Arnold to take some shortcuts while tracing rays, thus making rendering faster. If an object is flagged as opaque it means Arnold doesn't even need to access the parts of the shader that handles Opacity. The fact that this 'Opaque' flag is on by default means two things:

- Opacity will not work at all.
- Refraction will work however any shadows cast by the object will always be solid and not pick up the refraction color or density of the shader.

From the Attribute Editor you can see the Arnold attributes which among other things allows you to toggle the 'Opaque' switch:



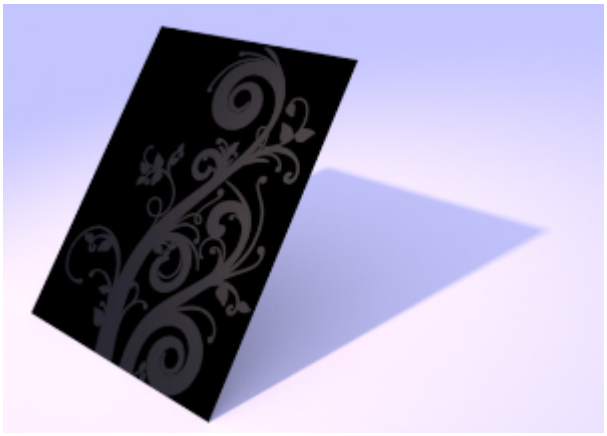
Use the ***Attribute Spread Sheet*** in order to set **aiOpaque** for multiple objects. Remember to select the shape nodes (select the transforms, and then press the Down arrow).

In the following scene you can see the Stanford dragon rendered with the opaque switch On and Off. Note that Opacity is not used at all in this example:

Opaque: On

Opaque: Off

Below is another example with Opaque enabled and disabled. In this case only ***Opacity*** is being used in the **Ai Standard** shader:



Sprite with 'Opaque' on

Sprite with 'Opaque' off

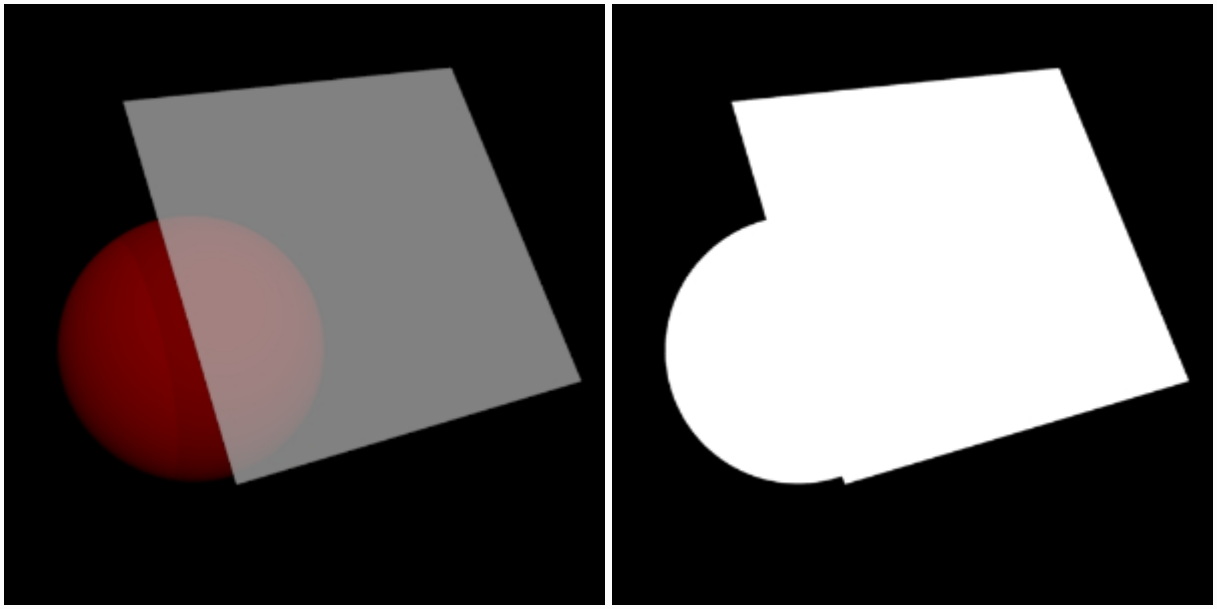
Optimization

As mentioned earlier using the right method for the right task will give you optimal rendering speed, however there are some things which can further speed things up:

- When using opacity for sprites, make sure that your mask is pure black and white, for example in the black areas there shouldn't be any noise or other imperfections as that will have a negative effect on the rendertimes, in other words it's not a good idea to use jpeg textures for masks as they often have compression artifacts.
- When combining Refraction and Reflection such as on the dragon above it can speed things up a lot if you make sure that the refraction part of the shader doesn't pick up any reflections, in other words disabling internal reflections, you can do this using a Ray_Switch shader.

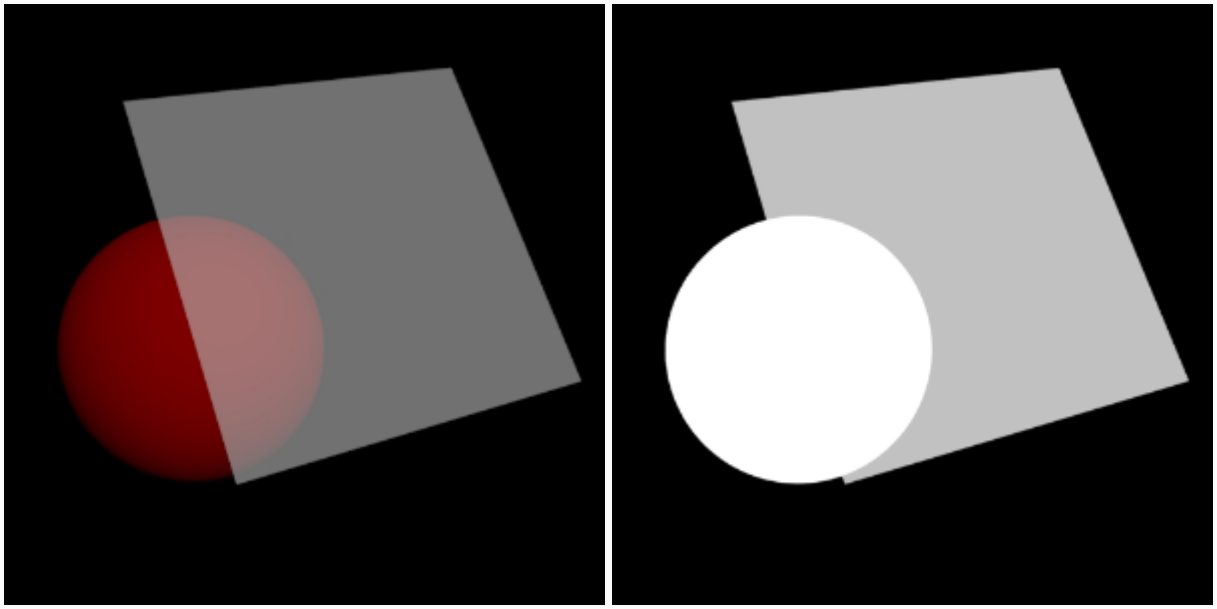
Transparency and alpha

Bear in mind that **Refraction Weight** will not propogate through to the alpha channel. The examples below show the difference in the alpha when using **Opacity** (correct) rather than **Refraction Weight** (incorrect). **'Opaque'** has also been disabled for the polygon plane.



Refraction Weight (incorrect alpha)

If you want to see the alpha channel, use **Opacity** instead of **Refraction Weight**:



Opacity (correct alpha)