Absorption and Nested Dielectrics

Absorption

When light travels through a transparent medium such as glass, the material absorbs a certain amount of light. This effect becomes more obvious when the transparent media has been tinted or colored in some way, like stained-glass or fruit-juice.

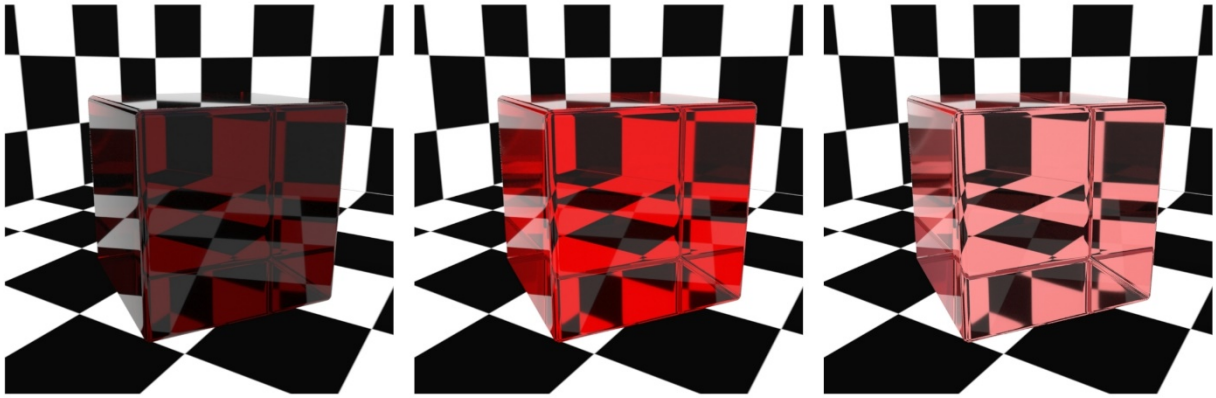
When rendering with Mantra, this effect can be simulated in variety of ways – using a volume to attenuate light as it travels through an object, for instance. However, for simplicity, a set of parameters which allow an artist to quickly achieve the desired look without the need for complex setups have been provided. These parameters can be found on the Principled Shader (As well as the Classic Shader).

Transmission Color

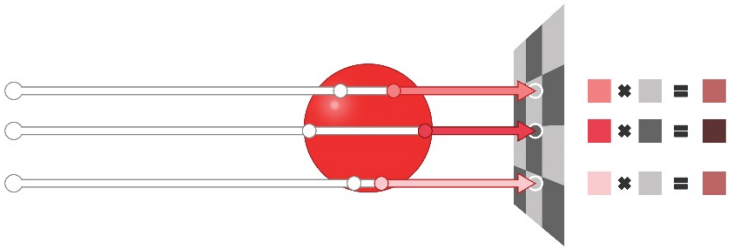
The color which will be used to tint a ray as it passes through a transparent object. It is important to remember that the amount of tinting is dependant on both the distance the ray has travelled through the object and the At Distance parameter.

At Distance

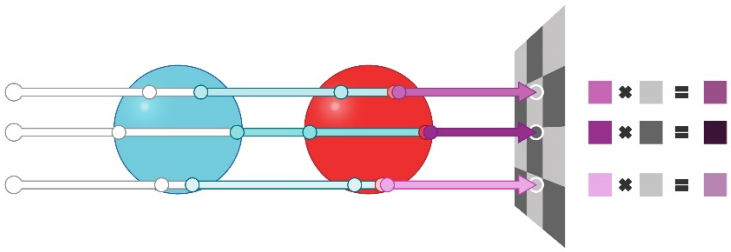
This parameter determines how the transmission color affects a ray as it travels through a transparent object. When the distance travelled through an object matches the At Distance parameter, one hundred percent of the Transmission Color will be blended with the result. In cases where the ray does not travel far enough inside the object to reach the At Distance limit, a smaller percentage of the Transmission Color is used. Conversely, if the At Distance limit is reached before the ray has travelled all the way through the object, a darker and more saturated version of the Transmission Color is used.



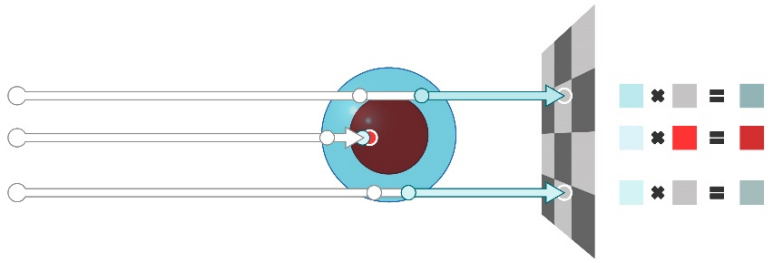
For simplicity, imagine that each ray which encounters a transparent surface is “tagged” with data like the hit location, Transmission Color, and At Distance value. When this ray hits the other side of the transparent object, it can now use this data to calculate how far it has travelled and therefore how to apply the Transmission Color. Eventually, this ray may hit an opaque object where it evaluates the surface shader and combines the result with the previously calculated absorption color.



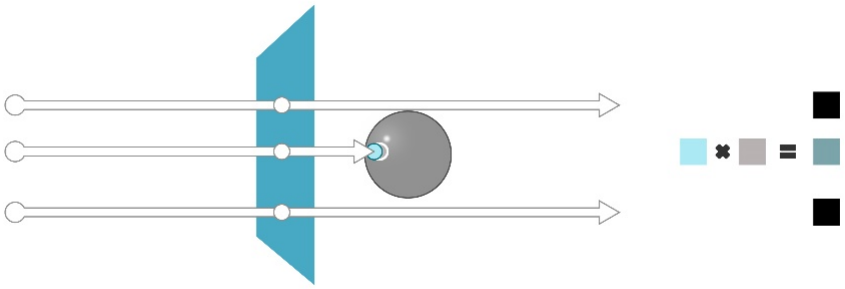
Using this method, it is possible to accumulate absorption colors across multiple, non-overlapping, transparent objects. (For a solution to overlapping transparent objects, see the section on Nested Dielectrics) Essentially, the process of tagging a ray and calculating its color can be repeated for every transparent object encountered. Rather than replacing the absorption color, the values are multiplied together before being combined with the color returned upon hitting an opaque object.



It should be noted that a ray does not have to exit a transparent object for absorption to take place. Consider the case of an opaque object embedded in a transparent one. Upon entering the transparent surface, the ray is still tagged with the relevant data to calculate the absorption color. The only difference is that the distance travelled is calculated at the point the ray collides with the opaque object rather than upon exiting the transparent one.



Additionally, the transparent object does not have to be a closed surface - single sided surfaces can take advantage of absorption as well. For places where a ray does not collide with another object in the scene, the absorption information is discarded and the color black is returned. (Essentially treating the one-sided surface as an infinitely Deep transparent object which absorbs all light)



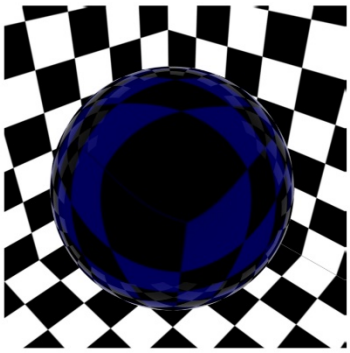
NOTE: A limitation of One-Sided transparent surfaces is that the absorption color will not be applied to Light Objects seen through transparent surfaces. If this effect is required, consider adding depth to your transparent object, or creating emissive geometry as a stand-in for your light.

Absorption Workflow Considerations

At Distance and Scale

When applying a shader with Absorption, it is always important to remember than that amount of absorption which occurs is based on the distance travelled through the object. Both the travel distance

and the At Distance parameter are measured in World Space. This means that the scale of your object can have a large influence on the final rendered results. Consider the following objects with the same shader settings:



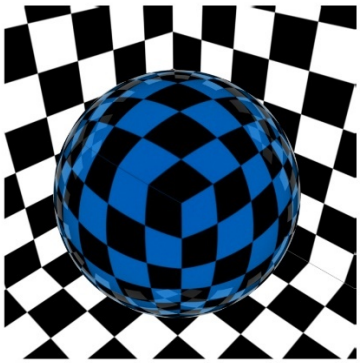
Sphere Diameter 1m
At Distance 0.1
Transmission Color

Sphere Diameter 0.1m
At Distance 0.1
Transmission Color

The example above illustrates how important it is to do your look development on an object at the scale it will be rendered when using Absorption. If you were to do your shader setup at one scale, but then the object was placed in the final scene at a drastically different scale, you will not get the same look.

Absorption and Camera Position

As explained above, absorption works by tagging rays that have intersected with transparent objects with useful pieces of data. Only when a ray intersects a surface which is facing the camera will this information be Considered (This allows mantra to track when a ray enters an object versus when it leaves an object). However, this also means that placing a camera inside a transparent object will not generate correct results.



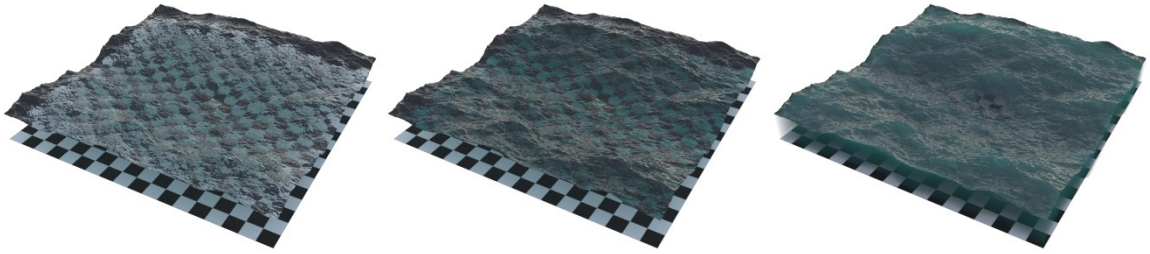
Camera placed
outside sphere

Camera placed
inside sphere

Notice how the Transmission Color has been lost when the camera is placed inside the object. The ray leaves the camera and intersects with the inside of the sphere as it leaves the transparent object - it never receives information about when it entered the sphere, so is unable to calculate the distance travelled.

Absorption and Volumetric Effects

Absorption is a very simple but effective way of representing the attenuation of light through a transparent medium. However, because it is not a truly volumetric effect, it is not appropriate for representing all types of lighting effects. In some objects, the attenuation of light is caused by particulate matter suspended in the medium, these particles can both absorb and scatter light. Consider a large fluid effect like an ocean:



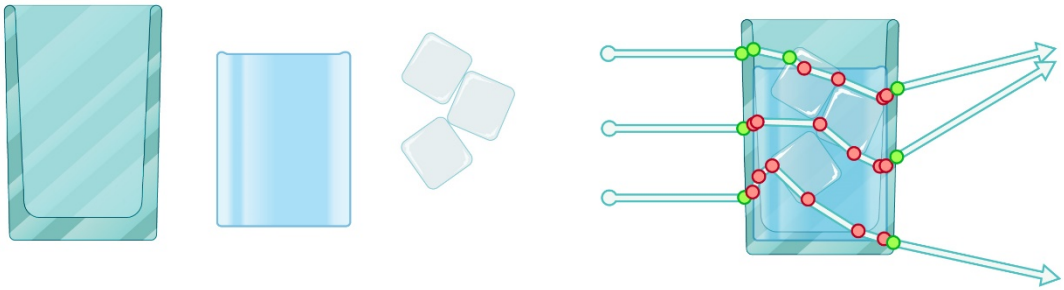
No Absorption **Absorption** **Uniform Volume**

You can see that while the ocean surface has a realistic feeling of depth using absorption, only the true volumetric rendering displays the characteristic light scattering of a real ocean. Unfortunately, volumes are costly to render, so some consideration must be given to the quality of a render versus the speed. In many cases, like a swimming pool or shallow river, absorption may be sufficient to achieve the desired result without a loss in quality.

NESTED DIELECTRICS

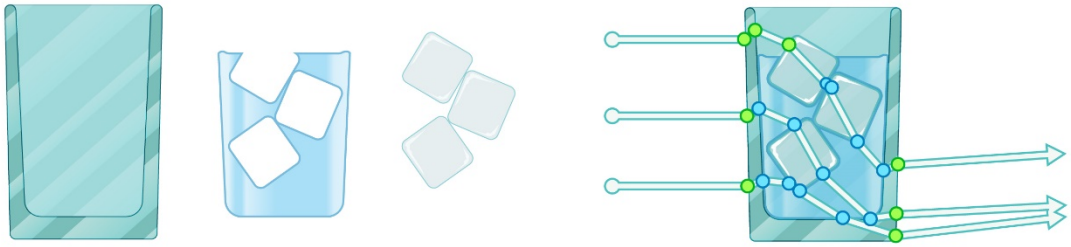
Rendering multiple transparent objects which are embedded inside each other is a complex task, both from the point of view of the renderer but also from the point of the of the artist building the scene. Consider the following example:

The simplest approach would be to model each of the objects separately and simply have them overlap. However, if you consider how a ray will travel through each of the transparent objects, it quickly becomes clear that you will generate a cascading series of incorrect refractions.



To make matters worse, not only will rays refract incorrectly, but there will be surfaces in places where none should exist (Fluid intersecting the ice cubes, glass Intersecting the fluid). Each one of these errors compounds the next creating a poor render. In the diagram above, every red dot is an incorrectly evaluated surface. This method is good in terms of workflow, but poor in terms of the result.

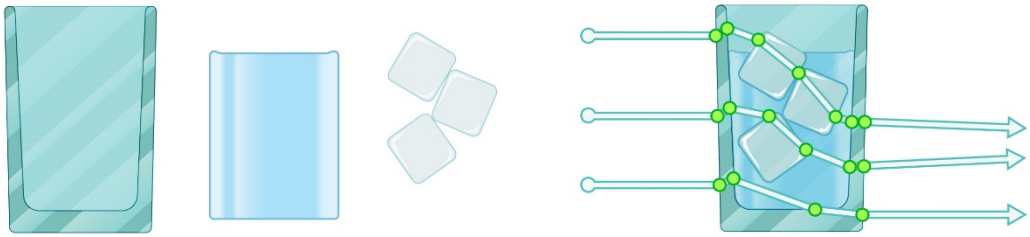
An alternative method would be to model each object without overlaps. This means fitting the fluid to the glass and carving holes in the fluid to make room for the ice cubes.



Unfortunately, this is not enough to correctly render this scene. In the diagram above, the Blue Dots indicate where a ray will need to travel through coincident surfaces. To render this scene correctly, those surfaces would need to have a special set of IOR values to correctly transition from one material to another (Glass, water, and ice all have different IO values). While this setup would result in a better final render, the work required is both tedious and error prone. Additionally, precision errors may arise from completely coincident surfaces. To correct for this, you may require a small gap between each material, but this will not give physically correct results.

The best solution would to be simply overlap the objects, but somehow let the renderer know which objects should have precedence over others. In some sense, it would be like asking Mantra to remove the overlaps at render time.

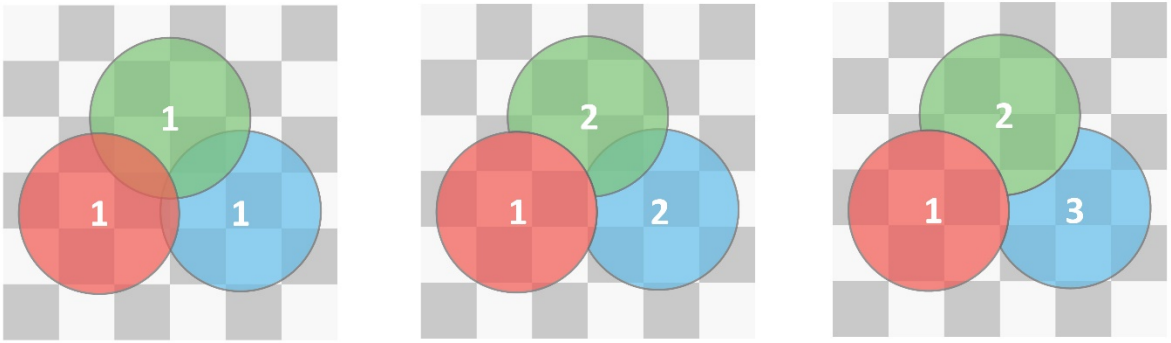
This is the concept behind Nested Dielectrics. By providing a number which represents the priority of a material, Mantra can track which object a ray has entered and simply ignore surfaces which have a lower priority. To achieve this, the Principled Shader (as well as the Classic Shader) has a parameter called Surface Priority.



Simply by setting this surface priority, Mantra would know both where and how a ray should transition from one transparent material to the other, correctly calculating the IOR values as it goes. In the above example, the glass would have the highest priority, followed by the ice, and finally the water.

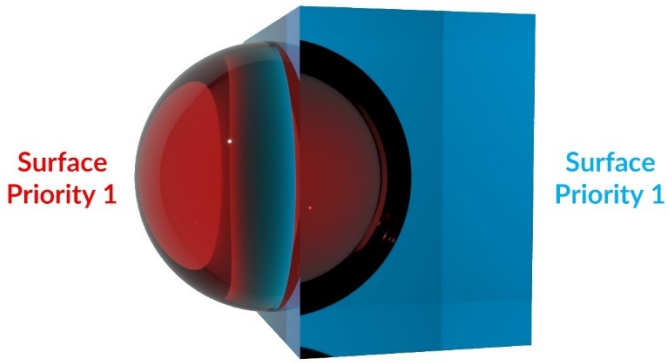
Surface Priority

A numerical value which establishes an order of precedence for transparent materials in a scene. A value of 0 indicates that the surface priority should be ignored. Increasing values indicate lower priority. Consider this simple 2d example.

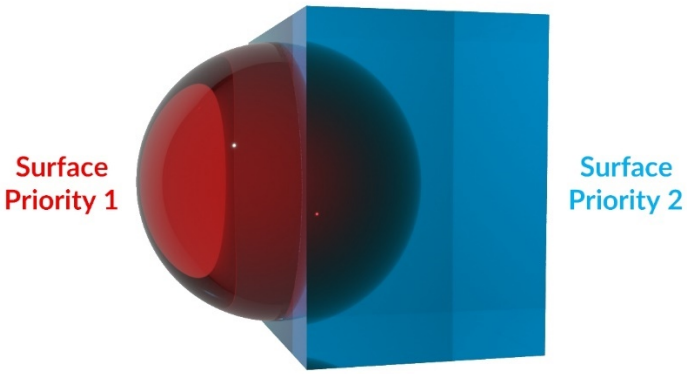


Notice that when all circles have equal priority, they simply overlap, creating multiple intersecting surfaces. However, as the priority value is increased for the green and blue circle, the interior surfaces are removed.

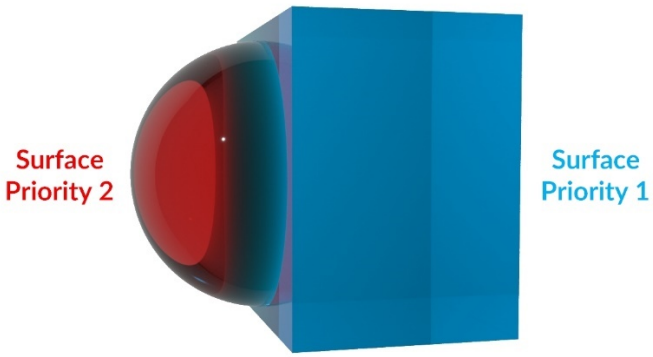
In 3d, on refractive surfaces with absorption, the effect of setting correct surface priority can be dramatic.



In the above render, both the red sphere and the blue box have the same surface priority. You can immediately see the problems in both the refraction and absorption. These problems arise because it is unclear to Mantra which surface is “inside” another surface and therefore it cannot correctly calculate the result.



In this case, the red sphere has a higher priority than the blue box (Remember, lower numbers mean higher priority. Red Sphere = 1, Blue Box =2). The absorption and refraction on the parts of the sphere inside the box is now correct and it appears as if there is no overlap between the two objects. Mantra simply ignores whichever parts of the Blue Box have overlapped the Red Sphere. This setup would work well for something like Ice Cubes floating in Water.



In the above example, the Surface Priority values have been switched (Blue Box =1, Red Sphere =2). This has the effect of removing any parts of the red sphere which overlapped the blue box. This would be an ideal setup for having water droplets resting on the surface of a glass.

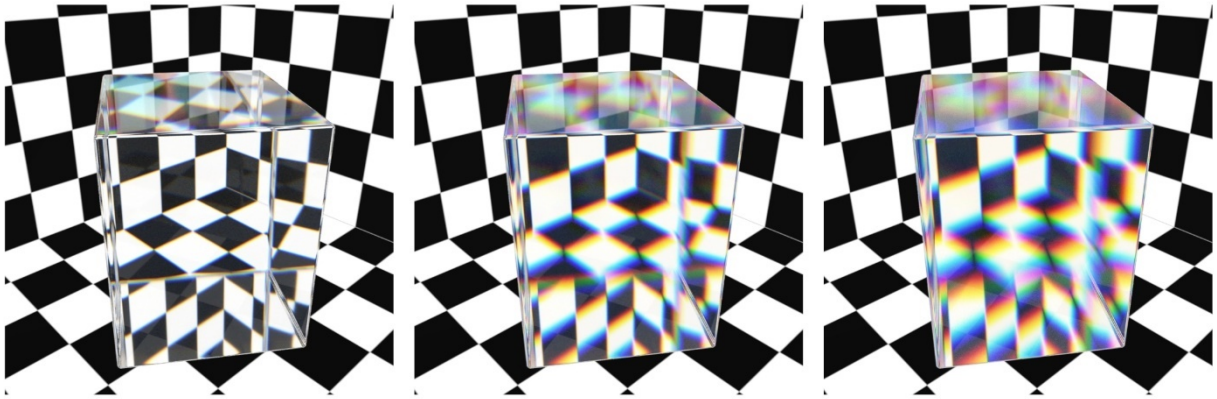
DISPERSION

In optics, dispersion can refer to the separation of light into its component wavelengths as it travels through a refractive material. A classic example of this effect is the spectrum produced by light travelling through a dispersive prism.

When rendering with Mantra, it is possible to simulate this effect using the Dispersion parameter included on the Principled Shader (As well as the Classic Shader).

Dispersion

When this parameter is set to a non-zero value, refracted rays are tagged with a single wavelength in the visible spectrum. Each of these wavelengths modify the underlying IOR causing the rays to separate as they travel through the refractive material. The larger this value, the larger the separation.

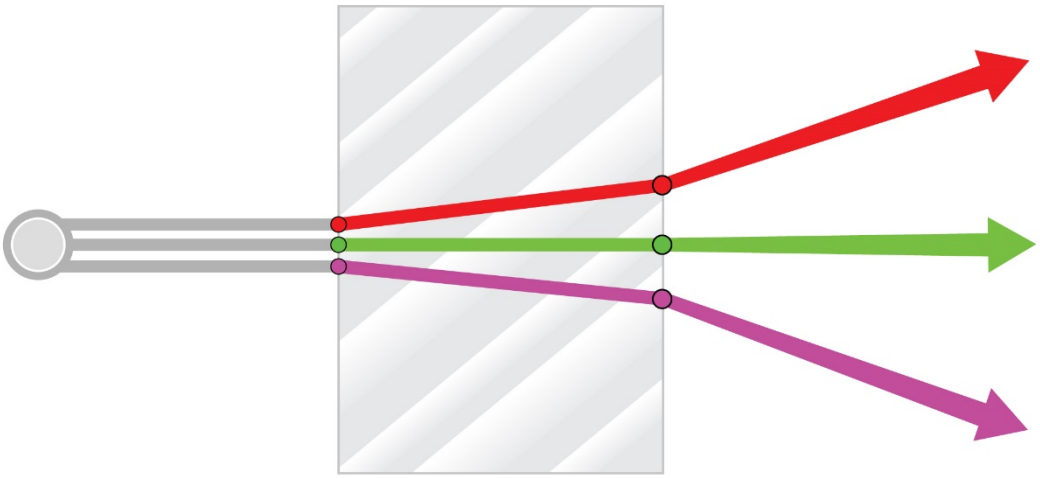


Dispersion 0.1 **Dispersion 0.5** **Dispersion 1.0**

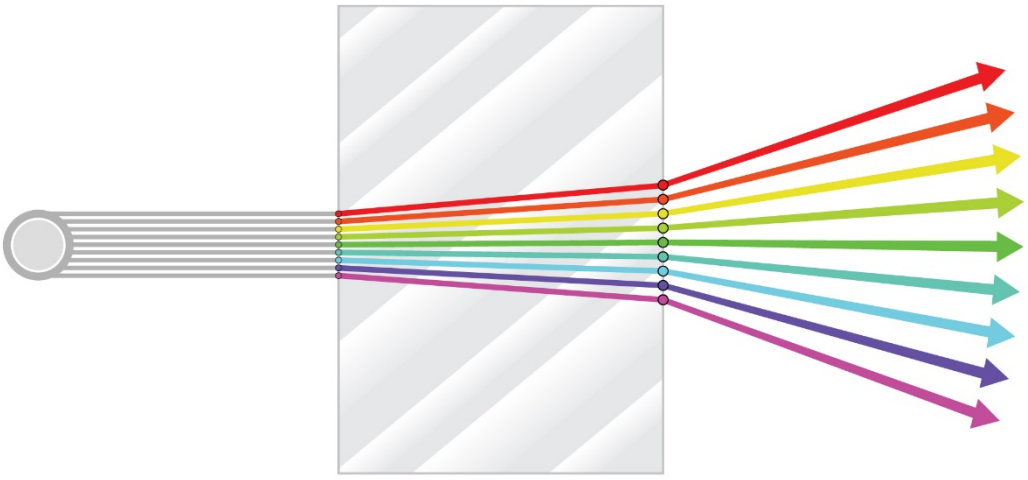
Because each ray is tagged with a single wavelength, it is important to have enough samples to represent the entire spectrum. Each ray is randomly assigned a wavelength; however, some attempt is made to ensure that the visible spectrum is uniformly sampled per Pixel Sample.

In the following diagram, you can see how a single Pixel sample, with 3 secondary rays, will not be able to adequately cover the visible spectrum. This will almost certainly result in noise in the final render as

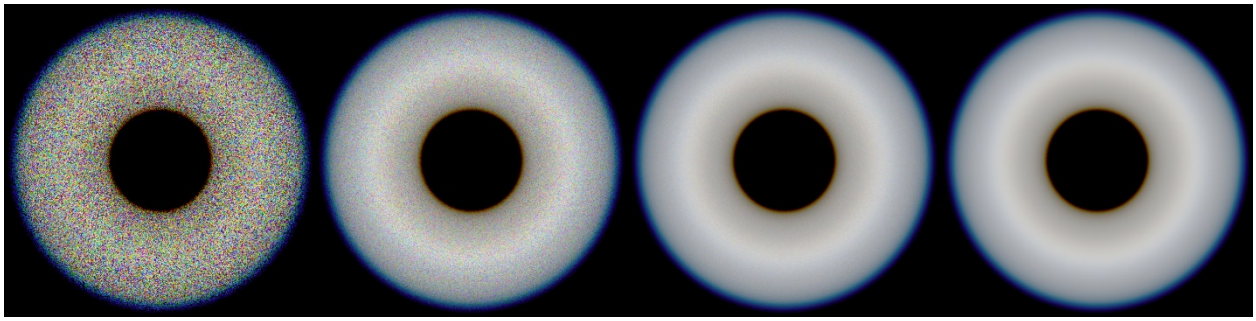
each pixel sample returns a random distribution of wavelengths.



However, as the number of secondary rays are increased, we see that more of the visible spectrum is represented in a single Pixel Sample. This will result in more consistency pixel to pixel and therefor less noise in the render.



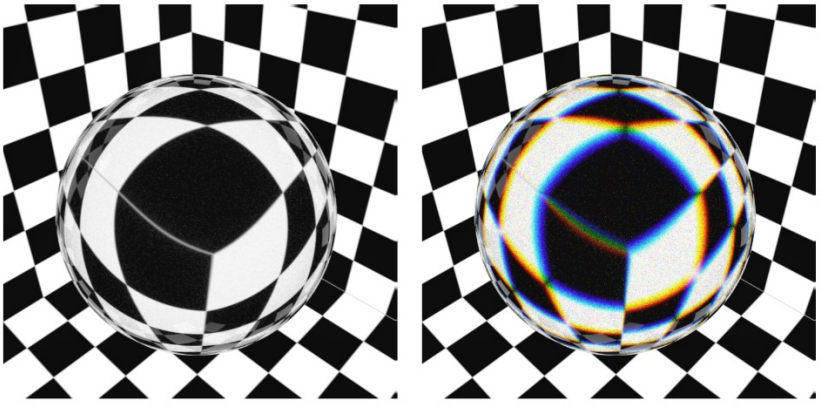
The following sequence of renders demonstrate how noise from Dispersion changes with the number of secondary rays. On the left, with only 1 Pixel Sample and 1 Secondary Ray, each pixel essentially receives a random color from the visible spectrum. As the number of secondary rays is increased to 5, 25 and 100, the noise caused by dispersion is completely resolved.



Dispersion Workflow Considerations

Removing Noise

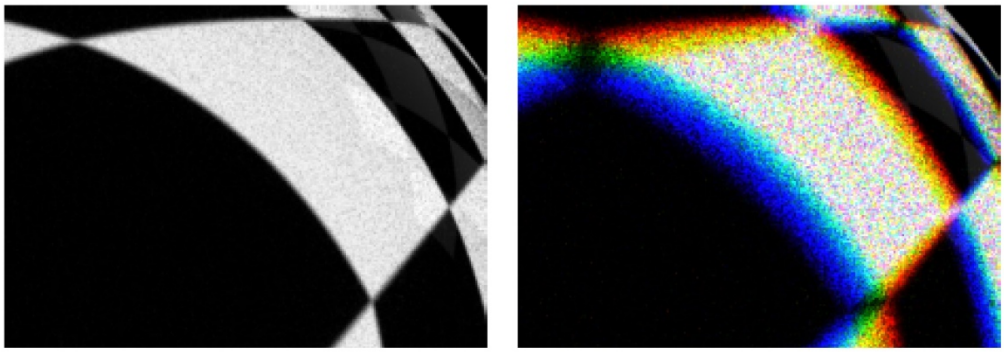
Most often, noise in a render has the appearance of small changes in brightness from one pixel to its neighbours. In the case of dispersion, this grainy look can be amplified by the introduction of color noise alongside the changes in luminance.



Dispersion 0.0
Max Samples 3
Refraction Quality 1

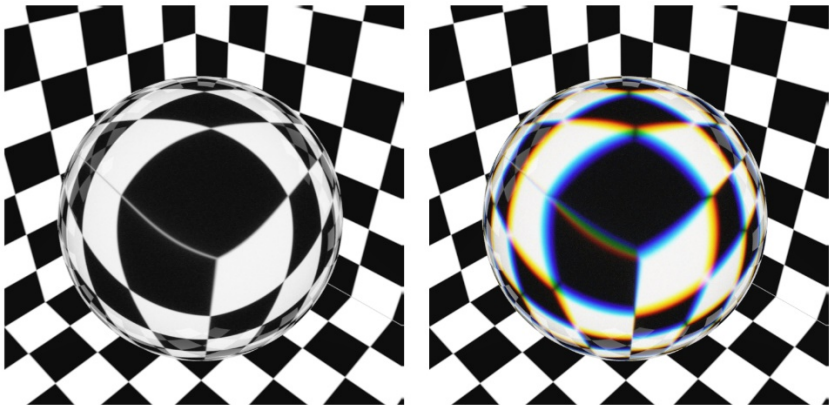
Dispersion 0.125
Max Samples 2
Refraction Quality 1

In the above example, both are similarly under-sampled but the image on the right appears to exhibit much more noise. This is because small changes in brightness are far less obvious than dramatic change in color caused by insufficiently sampling the color spectrum.



If you look closely at the white areas of the sphere, you'll see very similar noise patterns. However, the image on the right appears dramatically noisier due to the chromatic nature of the noise. It can often be necessary to increase the amount of sampling on objects with dispersion compared to similar objects

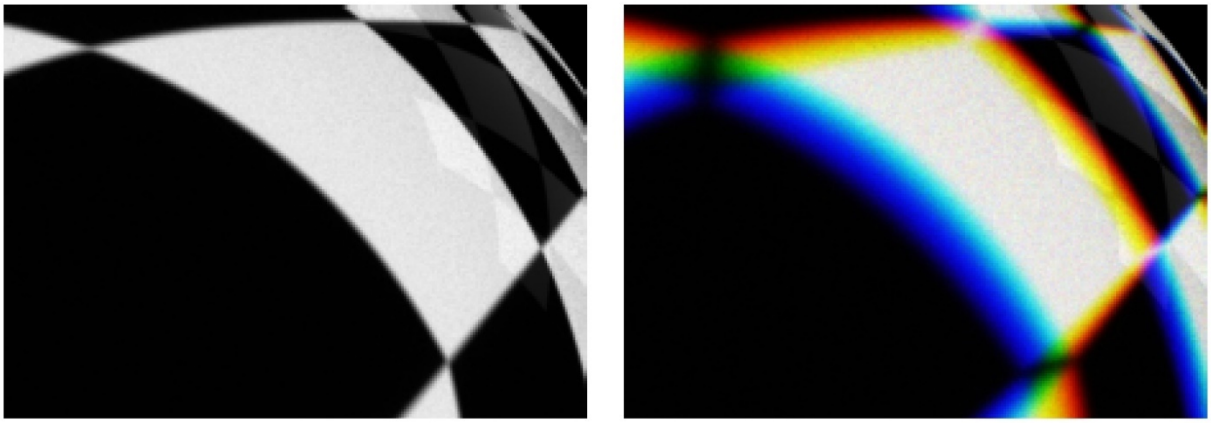
without dispersion. In the following example, you can see that significantly more sampling was required to achieve similar amounts of noise between both objects.



Dispersion 0.0
Max Samples 9
Refraction Quality 3

Dispersion 0.125
Max Samples 9
Refraction Quality 5

In this cropped version, you can see that the white areas of the spheres now have very similar noise levels and patterns. This is because enough of the visible spectrum has been sampled to converge back to the color white. But, it required almost twice the number of samples to achieve this result.



Because of this difference in sampling, it may be useful to override Refraction Quality parameters on any transparent object with dispersion enabled. This way you can be sure that you are sending extra refraction samples only to the objects which require them.