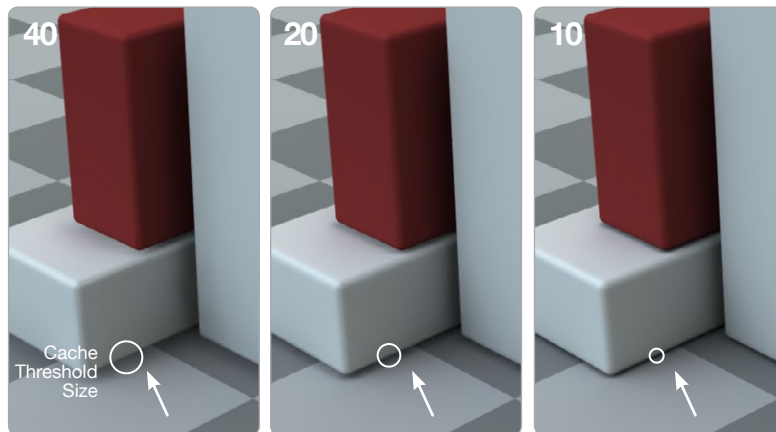
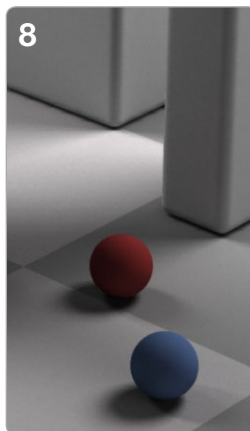
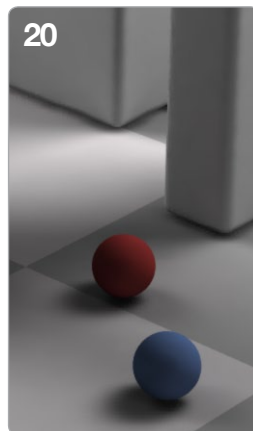
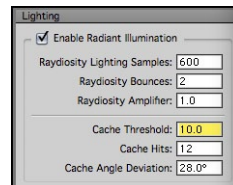
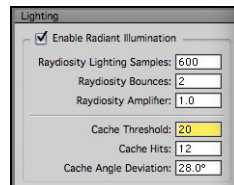
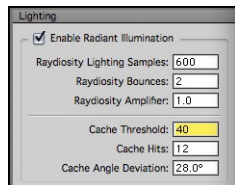


CONTROLLING RAYDIOSITY

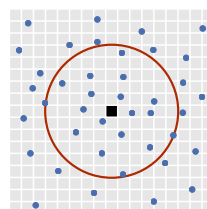
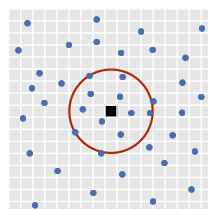
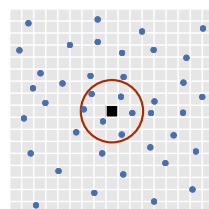


Notice the larger cache threshold makes a smudgy rendering, especially in shadows.



These two renderings use indirect illumination in the form of 3 'glow panels'. The right image using a cache value of 8 is more accurate, something that can be seen with the more highly defined shadows around the spheres. But it also took quite a bit longer to render than the image at left.

There is a trade off between accuracy and speed.



Blue dots represent cached radiance information.
Black square is the current pixel being rendered.
Red circle is the Cache Threshold boundary.

CACHE THRESHOLD

As the renderer computes the raydosity information for a scene, it stores this information in a cache. Because each pixel's value depends on the surrounding scene to determine its final appearance, a great many computations need to be done. However, there is a fair chance some parts of the nearby scene already have had their own lighting values computed and stored. This information is stored in the raydosity cache.

Instead of doing a full, time-consuming computation, the renderer can look within the cache threshold for any of this previously computed and stored radiance data. There's a fair chance these values will be somewhat close to what the current pixel should be. If it finds sufficient data it can interpolate radiance information. This is much faster than doing a full raydosity computation.

What the renderer needs to know is, how far around the pixel being rendered should it look for this stored information? This is the Cache Threshold.

The cache threshold is an actual pixel value. 40 means a boundary diameter of 40 pixels. This is a static variable. If you render your image at 500x500 pixels, it looks around the current pixel with a 40 pixel diameter. If you render the same image 5000x5000, this variable stays at 40. It does not scale as the size of rendering increases. This is an important consideration to make when doing test renders which may be small and then doing a final large scale rendering.

The larger the cache threshold, the greater the chance of inaccuracy as it pulls data from farther and farther away. Look in the shadow areas in the images above. As the threshold increases, quick areas of transition from light to dark become less defined. As the renderer is more aggressive about interpolating from cached raydosity data, these fine shadows become obscured.

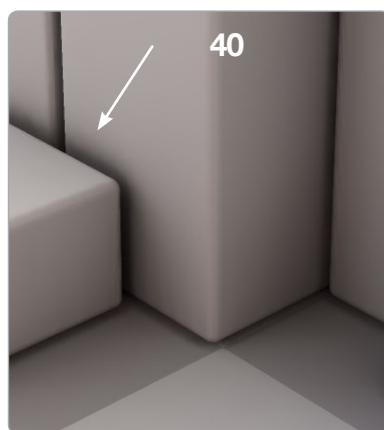
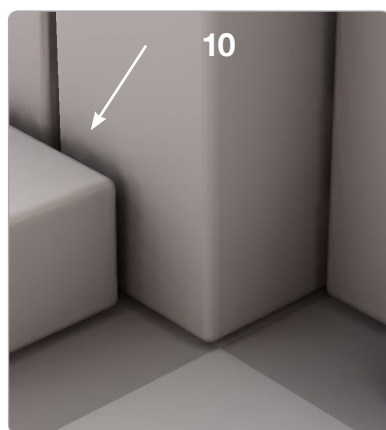
TRADE-OFFS

The larger the cache threshold, the faster the rendering. However a smaller threshold is more accurate.

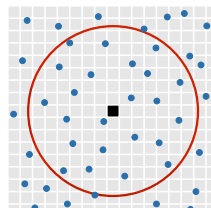
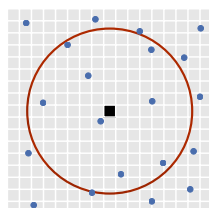
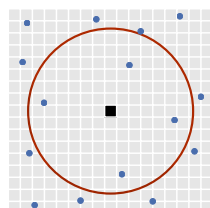
TYPICAL VALUES

Threshold values from 8 to 14 are very typical values. If the rendering is going to be very large, increasing these values somewhat can help to improve render speed without sacrificing quality.

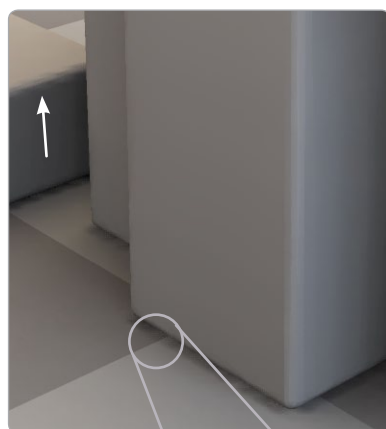
CONTROLLING RAYDIOSITY



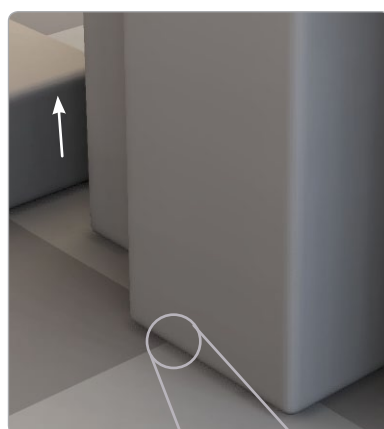
Notice the arrow. It points to an area that in the first image is rather dirty and blotchy. Because the Cache Threshold is relatively large for the image (40), cache hits value of 10 produce fairly inaccurate results. By increasing the amount of data that must be in the cache, the rendering is smoother and more accurate. Look at the subtle shadows under the curved bottom edges. They are a bit cleaner in the 20/20 images.



Blue dots represent previously computed radiance information. The black square is the current pixel being rendered. The red circle is the Front Threshold boundary in which the renderer can pull this cached data. The cache hits value tell the renderer how much data must be within this area before it can interpolate the black pixel's value.



Threshold: 34
Hits: 5
Low values cause blotchy caching artifacts.



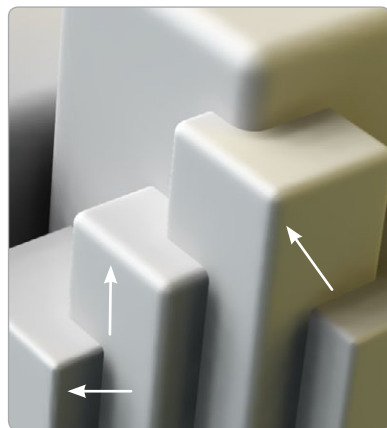
Threshold: 34
Hits: 20

CACHE HITS

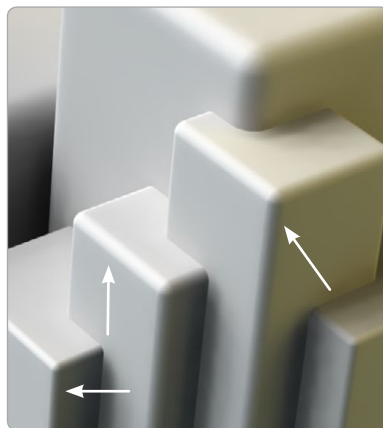
While the Cache Threshold tells the renderer how far around a pixel to look for cached illumination data, the hits value tell it how MUCH information should be within that region. It says, given the size specified how much data needs to be available BEFORE it can interpolate the current pixel's value. If there is insufficient cached data, it will perform a full computation.

The lower this value, the less accurate an interpolated value will be. But it also depends on the size of the Cache Threshold. Larger Cache Threshold values will generally require larger Hits values to avoid blotchiness and artifacts.

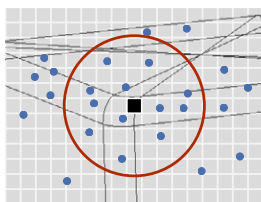
CONTROLLING RAYDIOSITY



Threshold: 34 Hits: 12
Angular Deviation: 30°
 The broad threshold, low Cache Hits and 30° Angular Deviation cause a lot of artifacts to appear on the rounded edges.

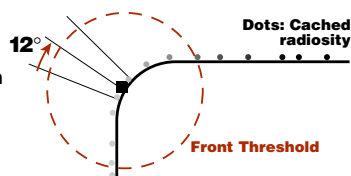


Threshold: 34 Hits: 12
Angular Deviation: 12°
 By changing the angular deviation to 12, we restrict the range that cached data can be pulled from around curved surface. The small curves become more accurate. They look 'tighter' and more defined.

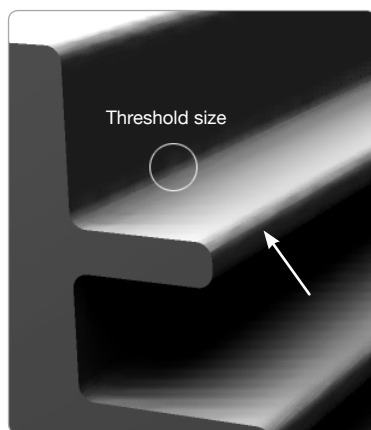
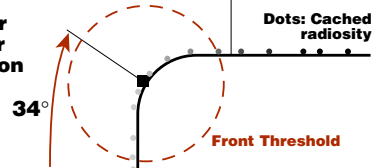


Tighter Angular Deviation

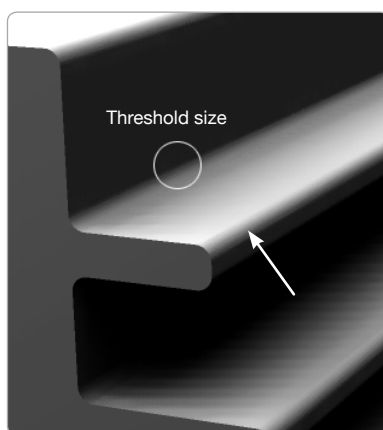
Black square is pixel being rendered.



Broader Angular Deviation



Threshold: 40 Hits: 10/10
Angular Deviation: 34°



Threshold: 40 hits: 10/10
Angular Deviation: 12°
 Notice the blotchiness on the flat surfaces. Because the caching system was set with fast but inaccurate settings, the flat areas have a lot of irregularities. However, with the tighter Angular Deviation we force the renderer to be more accurate around the small curved edges.

ANGULAR DEVIATION

This value determines the level of accuracy on curved surfaces. Larger values mean the renderer can look farther around a curved surface for cached radiance information.

When a pixel is being rendered, the renderer looks within the area defined by the Cache Threshold for possible data. If there are enough 'cache hits' available, then a full (and time consuming) computation doesn't need to be done. However, on a curved surface, cache data that falls within the cache threshold value, but outside the angular deviation value, are disregarded.

In the illustration at left, the rounded edges are small and more likely to fall within the cache threshold region (especially since it was set very large). This may cause caching artifacts to show up when data is interpolated across the curve from areas whose values are quite different. To prevent this, the Angular Deviation prohibits cached data from being used if it comes from too far around the curved surface.

Put simply, Angular Deviation trumps the Cache Threshold.