Flythrough Animations

Introduction

In this chapter we'll cover another type of animations supported by V-Ray - the flythrough animations. This means scenes where the camera "flies through" the scene, e.g. has its position animated. The other name which is used is "walk-through" animations.

Parameters

Light cache

If we use light cache, it's best if we calculate it for the entire walk-through animation, and not for a single frame only. Note that this is not strictly necessary - we can render the animation with the light cache being calculated each frame; however, rendering it only once will save rendering time, especially for long animations.

For this to work, we need to set the light cache settings parameter *mode* to *Fly-through* (`SettingsLightCache::mode = 1`). Make sure that the timeline animation range (set in `SettingsOutput`) matches the range which you want to render. This is important because the light cache will look at the current timeline animation range when calculating the fly-through cache.

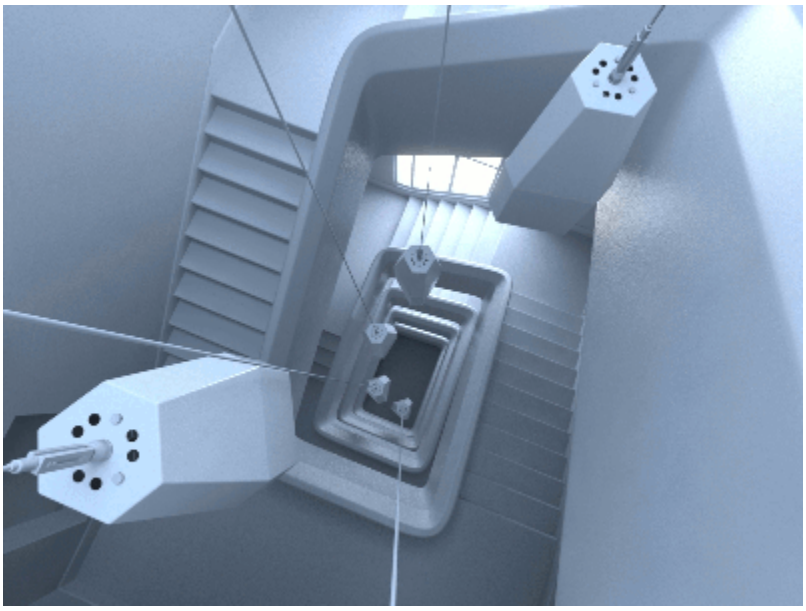
Since all the light cache samples will be distributed among all the animation frames, we will need to increase the light cache *Subdivs* value (for example Se`ttingsLightCache::subdivs = 2000`, the default is 1000). The exact value depends on the quality you want to achieve and on your specific animation. If the camera moves slowly, or the path is relatively short (e.g. just one room of a house) then you can use lower Subdivs value, since most of the samples will fall in the same place anyways. If the camera moves quickly or covers larger parts of the scene, you will need more samples to get adequate coverage everywhere.

Here's an explanation of the Fly-through mode taken from the docs: "Fly-through – Computes a light cache for an entire fly-through animation, assuming that the camera position/orientation is the only thing that changes. The movement of the camera in the active time segment only is taken in consideration. Note that it may be better to set *Scale* to *World* (`SettingsLightCache::world_scale = true`) for fly-through animations. The light cache for the entire animated sequence is computed only at the first rendered frame and is reused without changes for subsequent frames."

We'll cover in depth the light cache parameters in a later chapter about GI.

The preferred method for rendering animation is by using Brute force/Light cache as primary/secondary engine or Brute force/Brute force.

Example



The above animation is generated by rendering the file "Animated_Camera.vrscene" from the scene bundle. Check the comments inside the file and the included ones to see how GI can be used more optimally in an animation.

Code Example

This example is similar to the one from the previous chapter, but instead of animating an object, we'll animate the renderView's position. Here, the so called Animation2 API (available in AppSDK 6 and above is used).

Animation2 API is particularly useful, in the case when applying changes to animated parameter values during rendering. In this case, there's no need to change the current time/frame with `renderer.setCurrentTime(t)`.