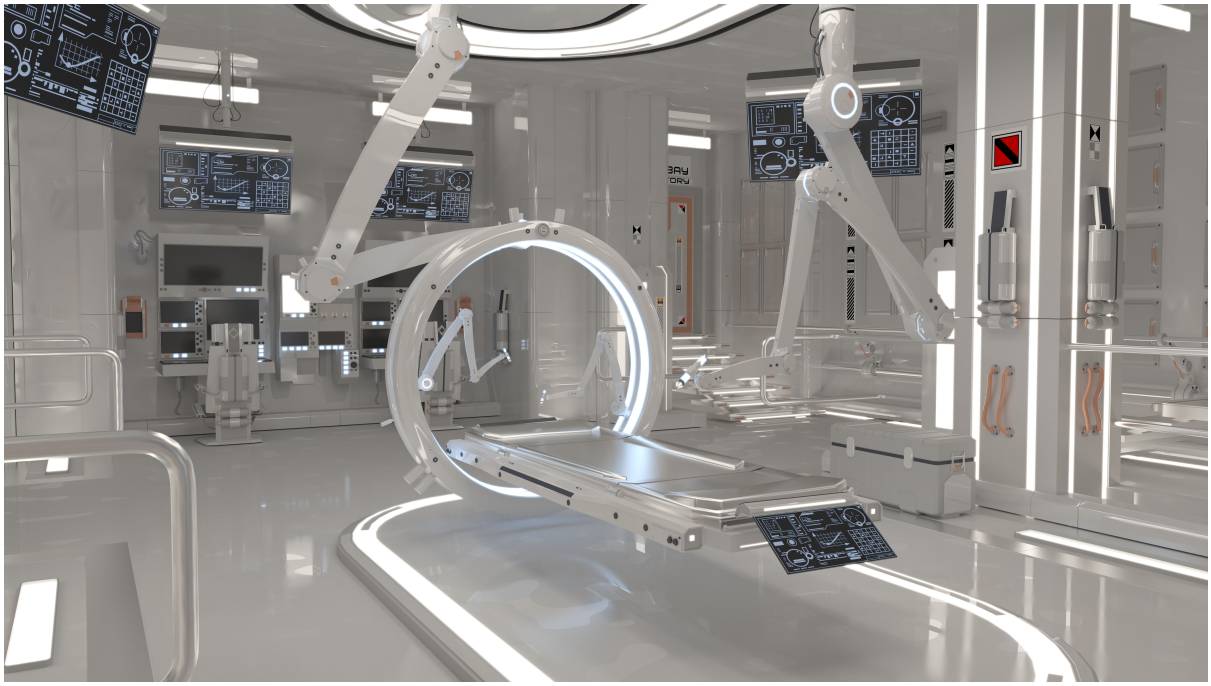
Iray for Cinema 4D

User's guide

4 April 2017
Document version 1.3.2



NVIDIA®
Iray®

Document build number 291717

Contents

1 Preface

1.1 Purpose of this document

This document describes the integration of the NVIDIA Iray renderer into the 3D graphics software MAXON Cinema 4D from a user's point of view. It hence focusses on both the available settings and controls, Iray for Cinema 4D adds to the existing user interface, as well as on practical guidelines on how to implement this product into an existing workflow. An in-depth description of the full capabilities of either the Iray renderer or Cinema 4D itself, is beyond the scope of this document. This manual is directed at an audience, fairly familiar with 3D rendering in general, and Cinema 4D in special. However, no prior experience with the Cinema 4D renderer or other render engines are required; the reader will find this document a valuable resource for rendering with Iray, even at a beginner level.

1.2 Organization of this document

The preface covers information about the document itself, its scope of purpose, a brief overview about its content structure and some notes on the used typography.

General information about Iray for Cinema 4D as a product, its license and release notes are given in the introduction (page 2). Instructions on setting up Iray for Cinema 4D (page 2) can be found here as well.

The rest of this user's guide exhibits a twofold character:

- After installing Iray for Cinema 4D you will notice several additions to your Cinema 4D environment. These elements are described in detail in chapter Iray for Cinema 4D User Interface (page 3).
- The second major part consists of task-oriented workflow guides (page 40) to help you adapting common tasks to Iray for Cinema 4D quickly.

1.2.1 Notes on typography

Italics are used to introduce technical terms that are specific to computer graphics and are otherwise not common in general written or spoken language. The first occurrence, where the term is defined, uses italic typeface; for following occurrences of the term, roman type is used. Italics are also used for emphasis.

Sans serif font indicates user interface elements. These include actual dialog names, menu items, attributes and parameters. All boldface names are written out exactly how they appear in the user interface, including the respective capitalization, for example Depth of Field.

`Monospace font` indicates examples of text that you enter exactly as shown. Monospace font is also used for system environment variables, file names and paths.

2 Introduction

2.1 About Iray for Cinema 4D

Iray for Cinema 4D is a plugin for the 3D graphics software MAXON Cinema 4D, extending it by the powerful render engine NVIDIA® Iray®. Iray is a highly interactive and intuitive physically based rendering technology that generates photorealistic imagery by simulating the physical behavior of light and materials. Unlike traditional production renderers, Iray delivers results reflecting real-world behaviors. Designers don't need expert knowledge of computer graphics techniques to quickly achieve photorealistic results.

Developed with a strong focus on seamless integration, Iray for Cinema 4D is designed to join your existing workflow with minimized amount of adaptation. You don't need to learn new ways of working, but it is useful to consider the best practices of this guide to get the most out of Iray for Cinema 4D.

2.2 Installing Iray for Cinema 4D

Follow these steps to install Iray for Cinema 4D:

1. Close Cinema 4D if it is currently running.
2. Download the Iray for Cinema 4D plugin to your desktop.
3. Open the folder and start the installer executable. This is an `.exe` file on Windows and an `.app` on the Mac.
4. Follow the setup wizard instructions to install the plugin. Note that the plugin must be installed in the `plugins` folder of a specific Cinema 4D installation. On windows, the installer tries to find an existing installation of Cinema 4D and, if successful, provides its `plugins` path as default entry. If you want to install Iray for Cinema 4D for another Cinema 4D installation, or if no default path is given, choose the `plugins` path of your Cinema 4D installation manually.
5. Before the installation finishes, the Iray Licence Manager setup is started. Follow the instructions of the Iray Licence Manager setup wizard to complete the installation.

3 The Iray for Cinema 4D user interface

After successfully installing Iray, you will notice several new elements in the Cinema 4D user interface:

- A new Iray menu (page 3), containing central access to every addition Iray brings to Cinema 4D.
- Viewport Rendering (page 3), to dynamically render the current Cinema 4D editor view with Iray while you work on your scene.
- Iray Live Render (page 6), a dialog to interactively inspect your scene, updating automatically on scene changes.
- The Iray Environment object (page 9) to create environmental lighting using sun and sky or image-based lighting (IBL).
- An Iray Section Plane object (page 15) that creates a cutting plane to exclude one side from rendering.
- Two tags: the Iray Camera Tag (page 16) for tone mapping and depth of field settings, and the Iray Object Tag (page 21) to mark individual objects in multi-pass rendering.
- An MDL material (page 23) to create realistic and highly portable shaders, using a wide selection of common material type presets.
- A new render engine to select in the render settings (page 27): NVIDIA Iray.
- General Iray preferences (page 35) to specify resources and other application-wide Iray settings.
- Iray Streaming (page 77) to render remotely and stream results dynamically.
- Iray Queuing (page 79) for scheduled distributed network rendering.

3.1 Iray menu

The most prominent user interface addition is the new Iray menu. It is a collection of all the elements Iray adds to Cinema 4D, easily accessible in one place. Many of the items listed here are also available where you are used to when working with Cinema 4D. Tags, for example, can also be added in the Object Manager and materials in the Material Manager. The preferences and render settings of Iray integrate seamlessly into the respective native dialogs of Cinema 4D.

3.2 Viewport Rendering

A very convenient way to work with Iray for Cinema 4D becomes possible through interactive viewport rendering. Iray renders directly to the editor view, while all editor controls remain usable and active. You can dynamically control objects, lights, materials and any other element of your scene while having an immediate render feedback.

Viewport rendering can be enabled and disabled in the Iray Live menu of the editor view. For a good live rendering experience, ensure the following properties are met:

- If the option En-/Disable in the Iray Live menu is disabled, "grayed out", go to the render settings and select NVIDIA Iray as renderer.
- Linear Workflow Shading should be switched off for the current view. When you turn on Iray viewport rendering, you are prompted whether this option should be set correctly automatically. To manually turn this setting on or off, click [Options ▸ Linear Workflow Shading] in the current view menu, and make sure it is disabled while you use Iray viewport rendering.
- Enhanced OpenGL must be enabled for the current view. When you turn on Iray viewport rendering, you are prompted whether this option should be enabled automatically, if it is not enabled already. To manually turn this setting on or off, click [Options ▸ Enhanced OpenGL] in the current view menu.
- Post Effects must be enabled for the current view. When you turn on Iray viewport rendering, you are prompted whether this option should be enabled automatically, if it is not enabled already. To manually turn viewport post effects on or off, click [Options ▸ Post Effects] in the current view menu.

In the Iray Live menu of the editor view you can also control the effective resolution of the rendering. The generated image always matches the editor size, but resolutions lower than 100% are scaled up to fit the viewport. Hence smaller settings result in significantly faster renderings, improving interactivity but of course also reducing image quality. With these options you can balance render performance and quality with respect to your needs and the current scene.

A small text overlay in the editor displays the current state of the viewport rendering. During the rendering it shows and updates the current render iteration, which is the level of refinement. You can control the maximal iteration in the Iray render settings. When the maximal iteration is reached, or another end criterion (page 28) is met, the overlay displays Iray: Ready!. Render progression is then stopped and the image does not refine further.

The following image shows the Iray viewport rendering interface and how it integrates into the Cinema 4D editor view.

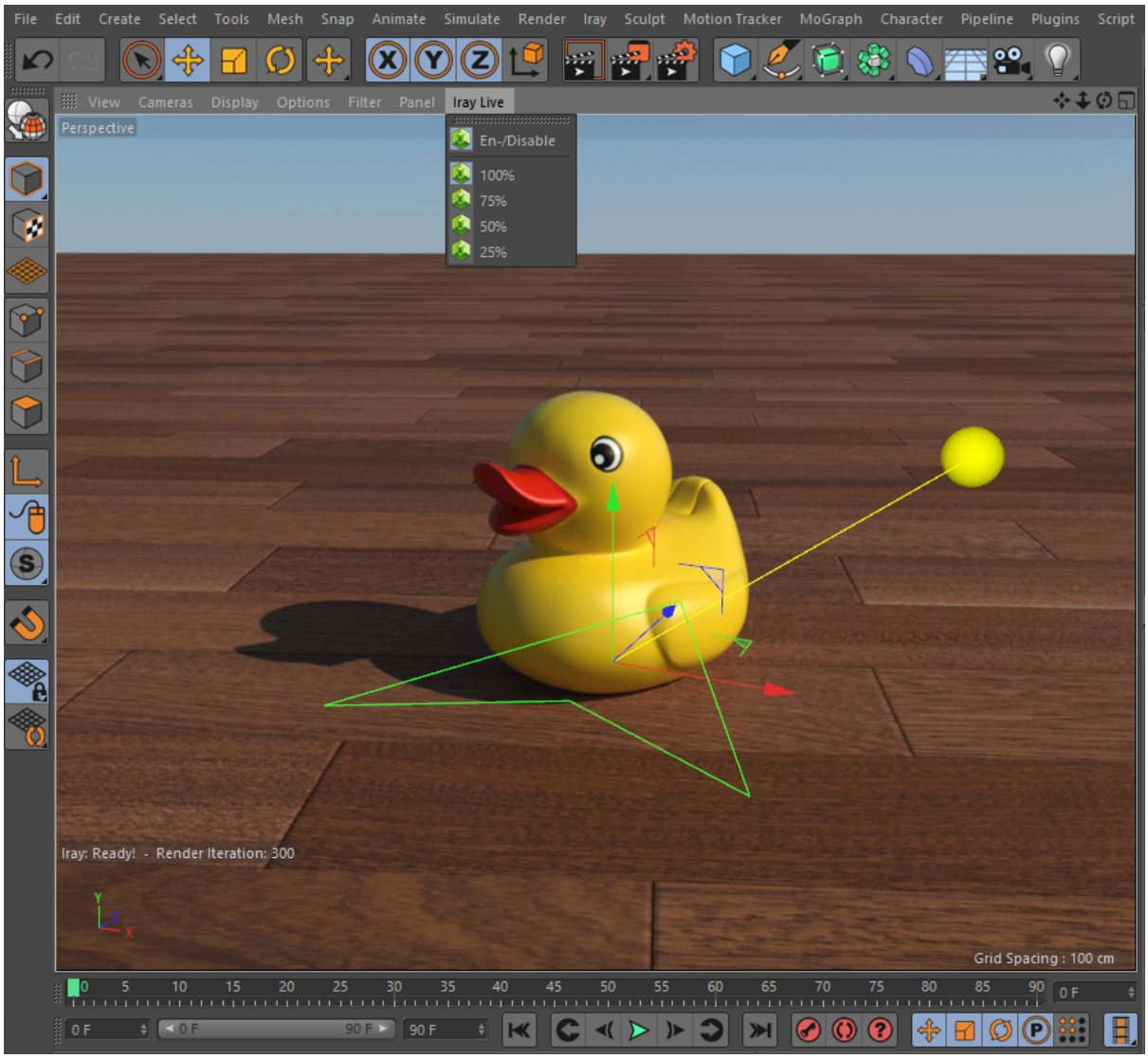


Fig. 3.1 – Iray viewport rendering: using the Iray Live menu, you can enable viewport rendering and specify its relative resolution. The text overlay on the lower left displays the current render state.

Note: Only one view can be used for interactive rendering. If you set your editor to show four views, for example, Iray viewport can be enabled for one of these four views. In most cases, the perspective view is suited best for viewport rendering. You do not need to disable viewport rendering manually when switching to another view.

Note: Viewport rendering displays the beauty pass, the full-color rendering with entire material evaluation. If you wish to interactively render other passes, please use the live render dialog (page 6) instead.

3.2.1 Update behavior

The viewport rendering update behavior is identical to the Iray live render dialog. Refer to section Update behavior (page 7) for more information; all the events that trigger an update apply for viewport rendering, too.

3.2.2 Differences between viewport rendering and production rendering

To maximize the workflow performance of viewport rendering, a few rendering features are disabled. The general differences between viewport rendering and final frame production rendering are the same as for Iray live rendering. Consult section Differences between live rendering and production rendering (page 8) for details. With the exception of the passes, these differences apply to viewport rendering as well (viewport rendering only renders the beauty pass, as mentioned above).

3.2.3 Controlling viewport rendering performance

Employing viewport rendering, you typically prioritize interactivity over image quality, as a quick feedback is is critical for a fluent workflow. The NVIDIA Iray logo is shown on the lower left of the viewport, just above the live render state overlay, indicating that Iray is busy finishing the current image. If you find this happen too often or experience other delays that restrain your workflow speed, consider these possibilities to increase render performance:

- Reducing the viewport render resolution is typically the best option to significantly reduce the render effort. Select smaller percentage values in the Iray Live menu of the editor view you render to. Smaller values result in less render time since smaller images need to be computed. The smaller images get scaled up to fit the viewport, hence of course also resulting in less overall quality for live rendering.
- Viewport rendering uses the current Iray render settings. Specifying end criteria (page 28) that are met early reduces render time. You rarely will need a high number of render iterations for viewport rendering.
- However possible, it is generally not recommended to use both viewport rendering and the separate live render dialog at the same time, considering render performance.

3.3 Interactive live render dialog

Iray for Cinema 4D comes with a fast, interactive render dialog, displayed in a separate window. It can be opened in the Iray menu (page 3). Once active, it gradually refines the render quality and automatically updates on every change of your scene. Consult section Update behavior (page 7) about details on interactivity.

Live rendering can be toggled on and off with the play/stop button in the lower right corner of the live render window. When stopped, the rendering does not update dynamically on any changes of the scene, preset or pass.

The Save button opens the currently rendered image in the Picture Viewer of Cinema 4D, where the usual options for viewing and saving are available. When the rendering in the live render window is still processing, it will not simultaneously update in the Picture Viewer: the current image is copied to the picture viewer and is henceforth separate.

3.3.1 Live render presets

For live rendering, you typically prioritize interactivity over image quality and resolution. It is hence often useful to provide different render settings for live rendering than for the final full size rendering. To control this, you can switch the render setting preset for live rendering directly in the Iray live render dialog.

Three general sources of render presets are available:

Active render settings
: The currently valid render settings of the scene as specified in the Cinema 4D render settings dialog.

Scene render settings
: A list of all render settings of the current scene.

Preset library
: All the render settings saved as a preset (also available in the Cinema 4D Content Browser).

The built-in Iray live render preset is used by default. Section Organization of render settings and presets (page 27) covers a brief overview on how to create and store individual render setting presets.

Keep in mind that the selected preset also determines the end criteria for the live rendering. Those can be adjusted in the render settings for each preset individually. The details of the available parameters are described in section End criteria settings (page 28) as part of the render settings. If there are no end criteria given for the selected preset, the live rendering will keep refining continuously.

Note: When the selected render preset does not employ NVIDIA Iray as renderer, the live render window will still use Iray, but with the default render settings.

3.3.2 Render pass

Here you can specify the information that should be displayed in the live rendering. The different options divide into three logical groups:

Beauty
: The final, full-color rendering with entire material evaluation. This is the default option for any kind of rendering.

Common passes
: A selection of useful pass presets. Section Pass presets (page 57) contains details about them.

Auxiliary passes
: All available auxiliary passes (for example Alpha, Normal and Depth) Refer to the multi-pass instructions (page 58) for a complete description of the auxiliary passes.

Note that the live render window renders the chosen pass only (even if the selected render preset has Multi-Pass options set). If you need to render multiple passes, use Render to Picture Viewer instead. See Differences between live rendering and production rendering (page 8) for other distinctions to keep in mind when using the interactive live render dialog. For task-oriented instructions on multiple passes, refer to section Practical Multi-Pass rendering with Iray (page 57).

3.3.3 Update behavior

Whenever the visual render output is about to change, the live render dialog updates immediately (if it is not put on hold manually with the stop button). An update restarts the render process; the previous rendering is discarded, and a new rendering is being initiated.

Common actions that cause the live rendering to update are for example:

- Moving objects in the scene
- Changing the geometry of an object
- Creating, deleting or modifying objects
- Adding or removing assigned materials
- Editing assigned materials, shaders or textures
- Controlling the (viewport) camera
- Changing the lighting situation
- Modifying parameters of Generators or Deformers
- Selecting a different render preset in the live render window
- Switching to a different pass in the live render window
- Triggering the start button in the live render window
- Resizing the live render window

Important: There are also actions that change the render result, but can *not* trigger an update of the live rendering. Thus, whenever you expect a change in the live rendering but nothing happens, just close the live render window and open it again.

A noteworthy feature of the interactive live rendering is that all animation-relevant calculations are taken into account. This also applies to procedurally created animations such as physics evaluations and dynamics. You can move forward or backward in time dynamically (or even play the animation), and the live render dialog will update automatically. Live rendering does not slow down the viewport animation; instead the update speed of the dialog depends on the scene complexity and the available rendering hardware.

Note: If you change the scene, for example via New, Open or Close, the live render dialog automatically stops. You can then start it again by pressing the play button.

3.3.4 Differences between live rendering and production rendering

The Iray Live Render dialog performs a single rendering of the current viewport and as such closely resembles the Render View concept of Cinema 4D. However, live rendering is done in a separate window, allowing for different passes and other features that viewport rendering (page 3) does not support. There are a few differences to rendering to the Picture Viewer to keep in mind. Note that the following differences apply both for the live rendering window, as well as to viewport rendering:

- No motion blur calculations are evaluated, so independent of the currently selected render settings, there are no motion blur effects in the live rendering.
- From a selection of the most common passes, you can define which pass to display in the live render view. This overrides the pass definitions from the render settings.
- Whenever there are different detail settings for editor and render, the live render dialog uses the editor value (similar to the Cinema 4D Render View result). Examples are:
  - Subdivision Surface Generator: Subdivision level
  - Metaball Object: Subdivision level

    - Thinking Particles Emitter: Birthrate
- The architectural sampler from the Iray renderer settings (page 28) is not evaluated for live rendering.
- The Bloom effect is not supported for live rendering.
- The Degrain Filter is not evaluated for live rendering.
- Displacement mapping of the native Cinema 4D material is not evaluated for live rendering.
- The render output is displayed as fast as possible in the live render dialog: whenever a more refined image is ready, it is shown instantly. In the picture viewer, the currently rendering image is refreshed far less frequently. Naturally, longer refresh times will generally increase rendering efficiency, while shorter times yield a better interactive experience. Iray for Cinema 4D internally determines the optimal interval, depending on what is needed: for live rendering, interactivity is favored over efficiency, so the image is updated frequently; rendering to the picture viewer, one usually prefers efficiency over interactive updates.
- The render resolution and aspect ratio depends on the live render window size, not the output definitions in the render settings.

3.4 Environment object

In Cinema 4D there is a Sky and a Floor object to visually define spatial boundaries of your scene. A Physical Sky object further allows to specify the environment lighting via time and location. While Iray for Cinema 4D can render the native Sky object directly, it also supplies a more powerful Iray Environment object that encompasses the functionalities of these objects. It provides two distinct modes of environment lighting:

- Sun and Sky, described directly below (page 9)
- Image-Based Lighting, described in section Image-based lighting settings (page 12)

Both modes can make use of a Ground (page 14), which plays a similar role as the Floor object. A third mode None can be used to (temporarily) disable environment lighting. As you are used to, most of the environment parameters are animatable (page 15).

As the environment is infinite in size, you cannot translate the Iray Environment object: it is always located at the world space origin. Also, scaling the environment would not make any sense; the object thus has no size and a uniform scale of 1, and it cannot be scaled. It further cannot be rotated around its X and Z axis as the orientation must be level with the world space ground plane (which is the XZ plane). However, the environment can be rotated around its Y axis, so you can define which direction in your scene is pointing north.

In Sun and Sky mode, the environment object is displayed in the viewport with an arrow pointing north and a yellow sphere visualizing the sun direction.

Note: When you use the Cinema 4D Sky object with a procedural Cinema 4D shader, remember that this shader will be baked into a texture so baking resolution is important. You can adjust the baking resolution in the Iray render settings. See Automatic shader baking (page 26) for more information on this feature.

3.4.1 Sun and sky settings

The Sun and Sky mode of the Iray Environment object is designed to enable physically plausible daylight simulations and very accurate renderings of daylight scenarios. By specifying a geographical location and a point in time, the direction of the sun and the visual appearance of the sky will be evaluated automatically. In the following sections, we outline the available parameters and settings to control the sun disk, the sky, the virtual ground plane and the horizon.

3.4.1.1 Time and location

If the Time and Location option is enabled, the direction of the sun gets determined by a point in time and a location on earth. This will make the visible sun disk automatically follow the direction of the actual sunlight. Hence if you want your scene to take place on Independence Day in Los Angeles, setting up physical environment lighting takes these simple steps:

1. Click [Iray ▸ Iray Environment] to add an environment object to your scene.
2. From its properties, set the Environment parameter to the Sun and Sky mode.
3. Turn on physical environment lighting by setting the Enable option under [Sun and Sky ▸ Time and Location].
4. Using the Time parameter, set the date to the 4th of July in 2015 and a time of 13:00 p.m. for example.
5. Set the Time Zone to the local time zone you want the scene to take place. For Los Angeles, that is Pacific Time Zone, so select GMT-8 here.
6. Since the 4th of July falls into the daylight saving time period, set the DST option. This virtually advances the clock by one hour, to the effective time zone offset is now GMT-7.
7. Finally, provide the geographical location of Los Angeles by setting the corresponding coordinates for Latitude (34° 3′ 7″ N) and Longitude (118° 14′ 34″ W).

Rendering the scene, you should now see environment lighting with automatically determined sun direction, intensity and sky color for the given properties. In the following, the available parameters are described individually.

Enable
: Enables or disables calculating the direction of the sun by time and location. If disabled, you can set the direction manually using Elevation and Azimuth, as described below.

Time
: Here you can set the time and date to use for calculating the sunlight direction.

> **Note:** A useful feature is that this parameter can be animated. See Animating the environment object (page 15) for more information.

Pressing the Today or Now button sets the date or time to the current value of your local system. Note however that the time zone will not be set to your operating system setting. Use the Time Zone parameter to match the time zone with the location your scene takes place.

Time Zone *and* DST

With the Time Zone parameter, you can match the local time zone to the geographical location. If the daylight saving time (DST) parameter is set, the clock is advanced by one hour, respecting daylight saving time.

> **Note:** When you change the Time parameter, the DST setting will not change automatically. You thus have to keep track of the DST setting manually, especially if the time and date of your animation changes and includes a DST switch.

Latitude *and* Longitude

Using these parameters you can set the position on earth at which the sunlight direction is evaluated. Time zones will be taken into account automatically: if you alter these values, the direction of the sun will change accordingly. Increase in eastward Longitude thus makes the sun rise earlier. Be therefore sure to set the Time Zone parameter to correctly correspond to the geographical location given here.

3.4.1.2 Sun parameters

If the parameter Time and Location, which automatically defines the direction of the sun, is disabled, it can be set manually using Elevation and Azimuth.

Elevation

Specifies the elevation angle of the sun direction if not specified by Time and Location. The value range is -90.0 degrees, which means "straight from below", to 90.0 degrees, which means directly from above (zenith). A value of 0.0 degrees means horizontal level.

Azimuth

Represents the angle of the sun's direction around the global Y axis, given within the range of 0.0 to 360.0 degrees.

Size

Sets the size of the visible sun disk. The value 1 is the physically correct size. Due to the fact that people tend to misjudge the proper size of the sun in photographs, we recommend higher values like 4.

The default value is 4, being slightly more visually pleasing than the correct size.

Glow

Defines the glow or bloom effect of the visible sun disk. Higher values blur the border of the disk more strongly.

3.4.1.3 Sky parameters

Haze

Use this to control the amount of haze in the air. The range is from 0 (a completely clear day) to 15 (extremely overcast, or sandstorm in Sahara). The amount of haze influences many lighting effects:

- The intensity and color of the sky and horizon.
- The intensity and color of sunlight.
- The softness of the sun's shadows.
- The softness of the glow around the sun.
- The strength of the aerial perspective.

Red Shift

The Red Shift parameter gives you artistic control over the "redness" of the light. Its default value of 0.0 is the physically correct value, but it can be changed within the range from -1.0 (extremely blue) to 1.0 (extremely red).

Night Color

Using this parameter you can limit the darkness at night. It is the minimum color of the sky, the sky will never become darker than this value.

As the sun sets and the sky darkens, the contribution from Night Color is unaffected and remains as the "base light level".

3.4.1.4 Floor parameters

Here you can control the visual properties of the horizon and the appearance of the virtual floor. The Iray environment creates a *virtual ground plane* that exists "below" the model. This makes it unnecessary to actually model geometry all the way to the horizon line; the virtual ground plane provides both the visuals and the bounce-light from such ground.

Note: Although called *Floor* parameters, they do not relate to the Cinema 4D Floor object. To create a solid floor plane that will catch shadows and clip geometry, use the Ground (page 14) object, which is described below.

Horizon Height

Use this parameter to tune the position of the horizon; it sets the "level" of the horizon. The default value of 0.0 puts the horizon at standard "height". But since the horizon is infinitely far away this can cause trouble joining up with any finite geometry that is supposed to represent the ground. It can also cause issues rendering locations that are supposed to be at a high altitude, like mountain tops or the top of New York skyscrapers where the horizon really is visibly "below" the viewer.

Note that the horizon does not actually exist at any specific "height" in 3D space; it is a shading effect for rays that go below a certain angle. This parameter tweaks this angle. The total range available is somewhat extreme, reaching from -10.0 (the horizon is "straight down") to 10.0 (the horizon is at the zenith). In practice, only much smaller values are actually useful, like for example -0.2 to push the horizon down just below the edge of a finite visible ground plane.

Horizon Blur

You can blur the visual border of the horizon using this value. At 0.0 the horizon is completely sharp. Generally we recommend using values lower than 0.5, but the full range is up to 10.0 for a horizon which only consists of blur and no actual horizon at all.

Floor Color

With this parameter you can define the color of the virtual ground plane. Note that this is a diffuse reflectance value, also called *albedo*. It thus acts like a diffuse bounce light, tinting objects that are near to the ground.

Note: The floor is a *virtual* ground plane. As such, it does not receive shadows or clip geometry. Use Ground (page 14) if you need these behaviors.

3.4.2 Image-based lighting settings

Selecting Image-Based Lighting mode, you can provide an image Iray uses to evaluate the environment lighting. This approach, utilizing a real-world image to illuminate the scene, is called *image-based lighting* and usually abbreviated as *IBL*. We recommend to use *high dynamic range images* (*HDR*) with a reasonable high resolution to achieve good results without artifacts.

For a detailed introduction to the principles of image-based lighting and best practices using IBL in Iray for Cinema 4D, see the imaged-based lighting guide (page 42).

3.4.2.1 Image-based lighting parameters

Environment Texture
: The image file to use as the texture for the environment dome illumination. While you can choose any common image file types like PNG here, it is highly recommended to use HDR panoramas or probes for environment lighting. Supported high dynamic range formats are HDR and OpenEXR.

Intensity
: A linear scale that is applied to the illumination intensity. Using this parameter you can adjust the lighting strength. See the imaged-based lighting guide (page 42) for example images.

3.4.2.2 Environment dome parameters

With these settings, you can control the spatial properties of the virtual environment:

Dome Type
: While most commonly a virtual infinite sphere is used for IBL, Iray provides different dome shapes of either infinite or finite size.

  The supported modes are:

  Infinite Sphere
  : The conventional infinite spherical environment

  Sphere
  : A finite size sphere shaped dome

  Box
  : A finite size box shaped dome

  Infinite Hemisphere
  : An infinite spherical environment, but with a textured groundplane

  Hemisphere
  : A sphere-shaped dome of finite size where the lower part of the environment is projected onto the plane dividing upper and lower parts of the sphere

  Box Ground
  : A box-shaped dome of finite size where the lower part of the environment is projected onto the plane dividing upper and lower parts of the box

  The default setting is Infinite Sphere.

Sphere Radius
: The dome radius for the dome types Sphere and Hemisphere in scene units.

Box Size
: The box size for the dome types Box and Box Ground. The values are interpreted as width (along the global X axis), height (along the global Y axis) and depth (along the global Z axis) in scene units.

Dome Position
: Specifies the origin of the dome in case it is of finite size. Using this parameter, you can offset the environment to match the camera position to create a plausible environment lighting.

Note: You can also enable the Ground parameter to create a shadow catching groundplane which intersects with geometry. However in image-based lighting mode, it will neither be visible as a solid geometric plane nor create diffuse bounce lighting: these effects originate from the Floor property of the Sun and Sky mode.

3.4.3 Ground settings

Both environment modes can make use of an additional Ground. It acts as a *shadow catcher* plane. This plane weights the environment contribution of the ground using a progressive estimation of the amount of shadow on the groundplane, caused by the objects in the scene (from direct light only). As opposed to the *virtual* ground plane of the Sun and Sky mode, it will both receive shadows and intersect with existing geometry. Thus objects reaching through this plane will actually be clipped, just like a proper geometric plane would do.

The groundplane is controlled with the following attributes:

Use Ground
: Controls whether the groundplane is enabled or not.

Height
: With this parameter you can define the Y position of the groundplane. The default value of 0.0 means standard height, level to the world space origin. If you change this height, you might also want to adjust the Horizon Height.

Shadow Intensity
: Controls the intensity of the shadow on the groundplane with a value greater or equal to 0.0. A value of 0.0 denotes no shadow at all and a value of 1.0 denotes regular shadow intensity, thus values between 0.0 and 1.0 denote a lighter shadow. Values above 1.0 increase the shadow intensity. Note that the range of useful values above 1.0 is rather limited and exceeding this range may lead to an unnatural dark shadow that can influence other additive lighting effects, such as reflections of the objects in the scene. The exact useful range is scene dependent. See section Soft physical sky shadow (page 45) for sample images rendered with different values of this parameter.

Reflectivity
: Turns reflectivity of the groundplane on or off. Material properties of the groundplane reflectivity are given by the parameters Color and Glossiness; these are thus disabled, if the Reflectivity option is not set.

Color
: Specifies the intensity and color of the groundplane reflectivity weighted by the fresnel term. This parameter is available when groundplane Reflectivity is turned on.

Glossiness

Here you can adjust the glossiness of the ground reflectivity. Smaller values result in diffuse reflectivity, higher values approach perfect mirror reflectivity. A value of 0.0 effectively disables ground reflectivity. This parameter is only available when Reflectivity is turned on.

The default value is 100,000.

Texture Scale

Use this to control the texture projection of the environment on the ground. This can be seen as an equivalent to the distance from ground where the real life camera was placed to take the captured environment map.

> **Important:** This parameter is only available for the Image-Based Lighting mode of the environment object. Since scaling depends on the environment dome size (as specified by the parameters Sphere Size or Box Size), texture scaling can not be combined with the Infinite Sphere or Infinite Hemisphere dome type.

3.4.4 Animating the environment object

All parameters of the Iray Environment object can be animated, just as you are used to with the native Physical Sky object of Cinema 4D. Smooth changes in the sun direction are calculated by interpolating the parameters between key frames. This also applies to the time and date: if two keyframes have different time or date values, Iray for Cinema 4D interpolates the values for in-between frames correctly. You can thus create animations with realistic environment lighting featuring a smoothly change of sun direction as time passes by.

3.5 Section plane object

Using the Iray Section Plane object, you can divide the scene in two parts: any geometry on the side of the positive Y axis of the plane gets excluded from rendering, the other side is rendered normally. It hence acts as a free clipping plane for geometry. Clipping planes can come in handy for various types of visualization work, for example to show the inner workings of an animated car engine. The plane can be rotated and translated freely; since infinite in size however, the plane has a uniform Size of 0 and cannot be scaled.

Up to eight section planes can be used simultaneously.

3.5.1 Section plane parameters

The Iray Section Plane can be controlled with the following parameters:

Clip Light

When off, this option means that the geometry will be cut off completely. Thus light may fall into the opened parts of the model, just as if the clipped geometry would not exist. When on, lighting behaves as if the *section plane* does not exist: the clipped geometry still casts shadows and prevents exterior lighting from entering the model.

Preview Size

Here you can control the visual size of the Iray Section Plane object in the editor viewport. As the plane which actually cuts the geometry is itself infinite in size, the

Preview Size has no effect on the rendered image. Nevertheless it may be useful to adjust the size displayed in the editor to be convenient with regard to the scale of your scene.

3.6 Camera tag

With this tag you can control camera properties regarding tone mapping, focus, lens type and stereoscopic attributes when rendering with Iray. By adjusting the focal distance, you can control the *depth of field* (*DOF*) effect. Using lens type and stereoscopic properties you can create outstanding renderings for *virtual reality* (*VR*) applications.

The tag can be assigned to a Cinema 4D camera object by selecting the camera in the Object Manager and then clicking Iray Camera Tag from the Iray Tags group. Alternatively, the tag can also be created using the Iray menu (page 3).

The following sections describe the available parameters to control tone mapping, depth of field, lens types and stereoscopic rendering. All these parameters can also be accessed in the Cinema 4D camera object directly.

3.6.1 Tone mapping

Internally, Iray calculates color intensities as the actual light energy, neither restricted to the display device nor to the color depth of the output format. Hence some mapping must be applied to decide which color intensity corresponds to a certain pixel value. *Tone mapping* is a method to map the unprocessed, simulated high dynamic range results to values that current display technology is able to work with and thus also offers optional display gamma correction.

Tone mapping is disabled by default, but it is recommended to use these built-in methods for best rendering performance and output quality.

In the following, a concise description of the available parameters is given; consult section Linear workflow and tone mapping (page 53) in the task-based chapter, for a general introduction on the conceptual basics.

Note: Since tone mapping is an image process, it does not affect the actual rendering (which of course takes place in 3D). Using the live render window, you can therefore observe each of the tone mapping parameters dynamically: adjusting the parameters, the live rendering will not start from anew; instead you can see the tone mapping effects in real-time, while the rendering keeps refining.

3.6.1.1 Photographic exposure settings

Enabled
: Enables or disables tone mapping.

Exposure Value
: In real-world photography, the *exposure value* (often abbreviated *EV*) is a logarithmic measure of a picture's exposure, given by the relative aperture (*f-number*) and the shutter speed (*film ISO*). Adjusting this value, you can thus set the combination of these properties: the f-number and the exposure time.

Increasing Exposure Value corresponds to a shorter exposure time, which yields generally darker images but also increases the details of parts with very bright illumination (for example the mosaic structure of a window facing the sun). Lower values of this parameter increase the exposure time, resulting in overall brighter images and revealing details of darker areas; however regions with high luminance might become over-bright, exposing a glare effect. Tone mapping is all about controlling these effects by mapping a large range of intensities to the limited range of the screen.

Whitepoint

In general, a *white point* is a color that defines "white". It is hence often referred to as *reference white*. The Whitepoint parameter in Iray is a color that will be mapped to "white" on output, which means an incoming color of this hue/saturation will be mapped to grayscale, but its intensity will remain unchanged.

The value of this parameter is interpreted as color temperature, stated in the unit *Kelvin*. Its default value of 6,500 K represents for example a diffuse, overcast but bright sky at daylight, which is typically perceived as a natural "white".

3.6.1.2 Advanced settings

Gamma

The gamma parameter applies a display gamma correction. If you want to display the image as-is, without further post-processing (as in a preview image), and if you did not apply gamma anywhere else in your workflow, this value should be set to match the display characteristics, otherwise disabled by setting to 1.

> **Important:** Be careful not to apply gamma twice in your workflow. For a linear workflow we recommend not only setting the gamma parameter to 1, but to disable tone mapping completely. See section Linear workflow and tone mapping (page 53) for detailed information.

Compress Highlights *and* Crush Shadows

These two parameters guide the actual tone mapping process of the image: they control exactly how the high dynamic range values are adapted to fit into the black-to-white range of current display devices. Both parameter's value range is from 0.0 to 1.0.

If Compress Highlights is set to one and Crush Shadows to zero, the transfer is linear. This means, the mapping behaves like a simple linear intensity scaler only. These values are set by default when adding the Iray Camera Tag.

Compress Highlights can be considered the parameter defining how much "over exposure" is allowed. As it is decreased from 1 towards 0, high intensities will be more and more "compressed" to lower intensities. When it is 0, the compression curve is asymptotic, i.e. an infinite input value maps to white output value, hence over-exposure is no longer possible. A common value for most scenarios is 0.5.

When the upper part of the dynamic range becomes compressed it naturally loses some of it's former contrast, and one often desires to regain some "punch" in the image by using the Crush Shadows parameter. When 0, the lower intensity range is linear, but when raised towards 1, a strong "toe" region is added to the transfer curve so that low intensities get pushed more towards black, but in a gentle fashion.

Saturation
: Compressing bright color components inherently moves them towards a less saturated color. Sometimes, very strong compressions can leave the image in an unappealingly desaturated state. The saturation parameter allows an artistic control over the final image saturation. The value range is from 0.0 to 5.0, where 1.0 is the standard "unmodified" saturation. Higher values increase saturation, lower values decrease saturation.

Vignetting
: In a real camera the angle with which the light hits the film impacts the exposure, causing the image to go darker around the edges. The vignetting parameter simulates this. At 0.0 it is disabled, higher values cause stronger darkening around the edges. Note that this effect is based on the cosine of the angle with which the light ray would hit the film plane, and is hence affected by the field-of-view of the camera, and will not work at all for orthographic renderings. A good default is 3.0, which is similar to what a compact camera would generate.

3.6.2 Depth of field

In photography, there is a certain range of focus in which a viewer recognizes objects as completely sharp. Objects that are closer to the camera and farther away from it, are out of focus and appear blurry. This range of focus is well-known as *depth of field*, often abbreviated as *DOF*. Simulating a depth of field effect can greatly improve the authenticity of a rendering.

With the Iray Camera Tag you can control this effect in the Depth of Field tab with the following parameters. See section Creating depth of field effects (page 50) for a practical guide.

Radius
: Controls the radius of the lens. The analogy in photography is the aperture (usually specified as the f-number, the ratio of lens focal length to aperture diameter). Higher values decrease the depth of field: the area in which objects appear as sharp gets smaller, so objects with a large distance from the focal point get blurred more strongly. This value is interpreted in scene units.

  A radius value of 0 means that the depth of field is infinite; therefore every object is in focus, independent from the distance to the camera.

  Note: While you can also provide negative values for the Radius, this is usually undesired. Even small negative values result in a very strong blurring of the whole image irrespective of the focal distance.

Lens Thickness
: Controls the thickness of the lens.

Use Camera Distance
: When on, the focal distance is taken from the camera object the tag is assigned to. When off, you can set this distance manually using the parameter Focal Distance.

Focal Distance
: Controls the focal distance to which the camera is adjusted. A point at this distance from the camera, is completely in focus; objects nearer to the camera and objects farther away from it will be blurred proportionally to their distance from the focal point. This value is interpreted in scene units, like the focal distance parameter of the camera object itself.

Note that the camera must be the active camera for rendering, to see changes of the focal distance in the final result.

3.6.3 Lens

A real-world lens is always a solid, volumetric object, which means it has a certain three-dimensional shape and a thickness. Differences in lens thickness influence the depth of field (page 18) effect and this setting is hence controlled as described in the previous section. The shape of the lens instead greatly affects the visual distortion of the generated image. Traditional implementations of lenses in digital rendering typically ignore this property, simulating an infinitely thin (planar) lens. While this usually sufficient for image display on screens, especially virtual reality applications demand for other lens shapes. In Iray for Cinema 4D you can control lens parameters in the Lens tab of the Iray Camera Tag.

Lens Type
: This specifies the lens type as given by its three-dimensional shape. The supported modes are:

  Planar
  : The "normal" shape, typically used in digital rendering when the output device is a screen, for example. You can think of a virtual image plane, capturing light on its front side falling in from all directions. The direction of the camera defines the orientation of this virtual image plane. This is the default setting, and a planar lens is also used if there is no Iray Camera Tag present at all.

  Spherical
  : Renders the scene with an angle of view of full 360 degrees. Images generated with this lens can be used for virtual reality applications. This is critical as a specific rendering must not be limited to a predefined viewing direction. Instead, the "camera direction" is given by the VR system and the rendering must include all possible viewing directions.

  Cylindrical
  : Renders the scene like a "panorama view", featuring an angle of view of full 360 degrees only in the horizontal direction.

  Radial *and* Inverse radial
  : Lens types of different orders. These special lens types allow to define the lens distortion manually. Radial distortion is sometimes called *barrel* distortion (as the resulting shape bends outwards, looking similar to a barrel); inverse radial distortion is also called *pillow* or *pincushion* distortion for similar reasons.

Distortion Parameters
: Controls the parameters in the radial distortion models. Note that third order radial types use only the first parameter. The fourth order types are described by the first and second parameter, and fifth order radial types use all three parameters. A value of zero means no distortion.

Distortion Scale
: Controls the relative scale of the lens distortion for radial lens types.

3.6.4 Stereoscopic

Cinema 4D supports stereoscopic rendering (often abbreviated as simply *stereo*), which can be enabled and configured in the render settings. In its picture viewer, Cinema 4D is also able

to display stereo renderings in a variety of ways. Iray for Cinema 4D integrates seamlessly into the native stereo workflow of Cinema 4D. The only setting you manually have to control is the camera offset given by the eye separation, and it can be set in the Stereoscopic tab of the Iray Camera Tag.

Eye Separation
: Controls the distance in scene units between the two separate renderings for each eye.

3.6.5 Backplate

It is often convenient to define a texture or color that is used as the background for rendering, when no geometry is hit. A typical application is an image featuring a rich distant environment that would be complex to create geometrically. Using the Iray Camera Tag, you can define any texture or color as a virtual backplate for the camera.

Note: Iray for Cinema 4D also supports rendering the Cinema 4D Background object directly. Applying any texture to the Background object is effectively identical to using the Backplate properties of the Iray Camera Tag. Note however, that the Iray Camera Tag has priority over any Background object. This way, you can define a general backplate that is used whenever no specific camera backplate is present. Different renderings from cameras that define a specific backplate with the Iray Camera Tag will ignore the general backplate and only the camera-specific backplate is applied.

Use Texture
: If enabled, the given texture acts as the background when rendered with the current camera. Note that you can instead define a simple background color with the Use Color parameter.

Texture
: The texture of the backplate, if Use Texture is enabled.

Offset U *and* Offset V
: A custom offset that is applied to the backplate texture. Using this parameter you can define an arbitrary position of the backplate by offsetting the texture in either direction by a given percentage. Entering negative values, you can invert the direction. These parameters are identical to the respective parameters of the native Cinema 4D Texture Tag.

  The default values are 0center of the backplate.

Scale U *and* Scale V
: A custom scale that is applied to the backplate texture. Using this parameter you can define an arbitrary resolution of the backplate by stretching or shortening the texture in either direction by a given percentage. Entering negative values, you can mirror the texture along the chosen direction. These parameters are identical to the respective parameters of the native Cinema 4D Texture Tag.

  The default values are 100used.

Use Color
: If enabled, the backplate renders a color instead of a texture.

Backplate Color
: Defines the backplate color if Use Color is enabled.

Apply Tonemapper
: Controls whether tone mapping, which is defined in the Tonemapping tab (page 16), should be applied to the backplate, too.

Apply Lens Effects
: Controls whether the lens effects, which are given in the Lens tab (page 19), should apply to the backplate.

Apply Depth of Field
: Controls whether the depth of field effects, which are defined in the Depth of field tab (page 18), should apply to the backplate.

3.7 Object tag

In post-processing, it is often crucial to render individual objects into separate passes. Using the Iray Object Tag, you can label objects that should be treated individually in rendering. The actual definition how the labeled objects are rendered and which ones contribute to which pass, is given in the Iray render settings using light path expressions. For a practical guide on the powerful capabilities of this tag, refer to section Rendering object passes (page 60).

Note: When no object tag is assigned, objects can still be addressed in LPEs with their name.

LPE Handle
: Here you can define the label that the assigned object can be referred to in light path expressions. You can enter an arbitrary name, but note that the following characters must be escaped with a backslash in the expression: \, ', and ".

  A detailed description on light path expressions can be found in section Using light path expressions (page 58).

Shadow Terminator Offset Mode
: With this option you can override the default shadow terminator offset behavior for single objects. In the render settings, you can specify whether to enable or disable slightly offsetting the shadow calculation to prevent shadow terminator artifacts. This setting accounts for every object in the scene. It may be useful to set this option only for specific objects, overriding the global setting. This can be achieved with the Iray Object Tag, the available modes are:

  Inherited
  : Use the global setting given in the render settings. This is the default for any object.

  Off
  : Disable the offset for the shadow calculation, regardless of the global setting.

  On
  : Enable adding an offset for shadow calculation to reduce terminator artifacts, regardless of the global setting.

  For details on shadow termination artifacts see the description of the Shadow Terminator Offset Mode (page 33) option in the render settings.

Matte
: A common task in computer graphics is to faithfully combine artificial objects with an environment given by a real-world photography. Suppose you have a photograph acting

as a virtual ground or floor. Merely placing 3d objects in your scene and rendering them will look unrealistic as there is no floor geometry to catch any shadows of the objects. To manually model the geometry as displayed in the photograph is often not the desired way to go. Instead, an invisible plane that is able to catch shadows could be used as well. A very useful property, an object behaving as such became known as *shadow catcher* or *matte object*. In Iray for Cinema 4D you can turn any geometric object into a a matte object with an Iray Object Tag and turning on the Matte option of the tag.

Shadow Intensity

Controls the visual intensity of the shadow caught by a matte object. This setting only refers to matte objects, it is disabled if the Matte option is not set.

3.8 Materials

Iray for Cinema 4D is built to join your existing workflow with as little adaptations needed as possible. The native Cinema 4D material, you are accustomed to, can be rendered directly with Iray. But Iray for Cinema 4D also brings new possibilities with it: materials based on the NVIDIA Material Definition Language (MDL) can be utilized to create realistic and highly portable shaders.

3.8.1 Cinema 4D material

A convenient feature of Iray for Cinema 4D is that it is capable of rendering the native Cinema 4D materials directly. Most material properties are supported without any kind of adjustment needed from your side. This also applies to broader material usage: Iray for Cinema 4D supports stacked materials (with the native Texture Tag) and materials for polygon selections (with the native Polygon Selection Tag) out of the box.

As Iray is a physically correct renderer, there are however a few things to consider when rendering the native Cinema 4D materials:

- Material settings that violate physical laws cannot produce predictable results. This is for example the case when a material is both very reflective as well as very transparent. Due to the conservation of energy, for real-world material the sum of its reflection and transparency energy can never be greater than the incoming energy. Iray adheres to these laws, so unrealistic combinations of material properties are likely to yield unexpected results.
- Cinema 4D material channels that do not have a comparable equivalent in Iray, are not supported. These are:
  - Environment
  - Fog
  - Glow
- Emissive materials, controlled with the Luminance channel, are highly dependent from overall exposure, and as thus, also from tone mapping.

  Note that the perceived luminance from an emissive material is not merely given by its absolute intensity. It rather depends on the relative difference from the surrounding illumination intensity. Consider this real-world example: if you let a strong blue light source shine on the street during bright daylight, almost no blue will be visible on the street surface. Inside of a room instead, the same lamp may cast clearly visible blue light on the floor, because its luminance now is no longer overwhelmed by the strong

sun. The same principle applies to rendering with Iray: if you use strong environment lighting, with the Iray Environment object for example, you need to enter way higher luminance values of an emissive material as compared to a standard Cinema 4D scene with traditional light sources.

Furthermore, keep in mind that Iray calculates true light energy values, not limited to a distinct number of discrete levels. These energy values must be mapped to a limited range of 256 levels for red, green and blue to be displayed on a screen. This process is called *tone mapping*, described in detail in section Linear workflow and tone mapping (page 53). Tone mapping greatly affects how luminous materials appear in the final result: in a typical scene the sun intensity may be thousands of times higher than other emissive materials. Applying tone mapping with an exposure value so that the sunlight is seen as white, the emissive material is likely to be rendered very dark, if not totally black.

See also section Principles of physically based rendering (page 40) for other hints about scene setup for Iray.

3.8.2 MDL material

Iray materials are based on the NVIDIA Material Definition Language (MDL). MDL materials are constructed from physically based elements and functions that are easily layered to achieve a rich continuum of materials without the need to program or compile. Furthermore, MDL materials can be shared between supporting applications. For example, create an MDL material in Iray for Cinema 4D, save it to your library, then use it in Iray for Rhino, Iray for Maya, mental ray or any other supporting application.

To create a new MDL material, follow these instructions:

1. In the Material Manager, click [Create ▸ Shader ▸ NVIDIA MDL Material].
2. A dialog opens, containing all the MDL materials that were found. They are organized in a tree structure of their source location. Select the material of your choice, for example [Material examples ▸ Asphalt].
3. Double-click the newly created MDL material. In the Material Editor you can now adjust the material properties.

You can change the current base material in the material editor using the browse button next to the Select Material parameter; this opens the MDL material selection dialog again. The dialog contains all MDL materials that could be found. Iray for Cinema 4D searches in several locations to compile this list; these locations are described section MDL search paths (page 24).

A very powerful feature of the MDL integration of Iray for Cinema 4D is that every procedural or numeric material property can be defined by a shader. This resembles the concept of nodal based material systems: the output of a procedural shader or algorithm can act as an input parameter for another shader. Per default, each parameter value can be controlled directly. A color parameter, for example, provides a color picker, and a numeric parameter provides a floating point field to directly adjust the value in the material editor. If you want a specific parameter to be defined by another shader, click the shader icon on the right hand side. The user interface then changes so that you can select a shader instead of a value. Just like for native Cinema 4D materials, you can click the shader to adjust its properties, and use the

navigation controls in the menu bar. Note that the shader icon is a toggle button: clicking it multiple times, you can switch between direct value input and procedural shader input.

Note: When you hover the mouse cursor over the shader icon of an MDL parameter, a tool tip shows up, containing a description of the parameter.

3.8.2.1 Export MDL preset

Having created and adjusted a custom MDL material, you might want to save the material as a preset. This can be done as follows:

1. Double-click the MDL material that you want to save.
2. On the left hand side of the Material Editor, click Export MDL Preset.
3. Type a name for the material preset and confirm it by clicking OK.

The next time Cinema 4D is started, you can now select the exported MDL material from the user_materials group. The preset gets saved to the MDL user path.

3.8.2.2 MDL search paths

I edited this section to try to clarify the operating system differences and the description of material name conflicts. Please review. *[by Andy]*

The MDL materials in the user interface are located in various directories in the file system. A directory in which Iray searches for materials is called a *search path*.

By default, Iray searches for MDL materials in four types of search paths:

Iray MDL search path
: The `mdl` directory in the directory containing the currently running Iray for Cinema 4D installation.

MDL admin search path
: A directory specified by the `MDL_ADMIN_PATH` environment variable. When this environment variable is not defined, an operating-system dependent path is used instead.

Operating system	*Search path*
Windows	`c:/ProgramData/NVIDIA Corporation/mdl`
MacOS	`/Library/Application Support/NVIDIA Corporation/mdl`
Linux	`/usr/share/NVIDIA Corporation/mdl`

MDL user search path
: A directory specified by the `MDL_USER_PATH` environment variable. When this environment variable is not defined, Iray searches a directory in the user's home path instead. This directory differs based on the operating system:

Operating system	*Search path*
Windows	`c:/Users/SampleUser/Documents/mdl/`
MacOS	`/Documents/mdl`
Linux	`/Documents/mdl`

Custom search paths

If you acquire third-party MDL materials or write your own custom materials, you can add their directories to the MDL search paths through the Preferences dialog. See the "MDL search paths (page 37)" subsection of the "Preferences" section (page 35).

To find a material, Iray searches *by its name* through the search paths. Even though more than one material with the same name may exist in the search paths, Iray will only load the first material it finds. The material loaded by Iray is therefore dependent on the order of the search through the four types of search paths.

The search paths are processed by Iray in the order listed above, summarized here:

1. The mdl directory of the Iray for Cinema 4D installation directory
2. The MDL admin search path
3. The MDL user search path
4. Custom MDL search paths

For example, if you have a material called "my_material" located both in the Iray for Cinema 4D installation directory (the first search path) as well as in your home directory (the third search path), the material in your home directory will not be loaded.

Some questions for Andre:

This section contained this sentence:

"The conflict happens when the relative paths are identical, hence you can avoid the conflict by placing materials with the same name into different subfolders."

But this still means that only one material will be loaded and that there is still a "conflict" between two identically named materials located in different directories. Only one material of a given name will ever be loaded.

Can the custom search path contain multiple directories? If so, how are they separated? (For example, with the colon character as is often the case in Unix-based systems.) *[by Andy]*

3.8.2.3 MDL material preview render preset

To get a quick feedback on material property changes, MDL materials display a preview rendering in a similar way to the native Cinema 4D material. For MDL materials, this preview is rendered by Iray, of course. By default, the MDL material preview is generated by using built-in render settings that are optimized to quickly finish rendering, but to still yield meaningful quality. However, based on your available rendering hardware and your material complexity, it might be useful to manually control MDL preview rendering. This can be done by creating and assigning an individual material preview render preset:

1. In the Material Manager, click the material which preview render settings you want to change.
2. In the Attribute Manager, click the Basic tab, if not already selected.
3. Expand Select Render Preset. Here you can now select a different render preset Iray uses for rendering the material preview.

The list to choose from contains all saved render settings presets. This includes the predefined presets that come with Iray for Cinema 4D as well as any saved presets. You can manage the available presets with the Content Browser of Cinema 4D: the folder Presets contains all of them.

For material preview rendering, there is an override regarding the render time: you can define custom end criteria settings, but the rendering will be stopped after two seconds in any case. This can not be changed with the end criterion Time.

Note: The active scene render settings are not included in the list of available material render presets to choose from, as it rarely would make sense to use high-resolution or even production-level settings for material preview. But you can add custom render settings by saving them as a preset. Saved presets are included in the material preset list.

3.8.2.4 NVIDIA MDL shader

Iray for Cinema 4D includes a plethora of MDL shaders, integrated seamlessly in the material user interface of Cinema 4D. These shaders are practicable to define MDL material properties (or other shader properties), as described in section MDL material (page 23). The NVIDIA MDL shader is thus the default shader when switching from value input to shader input in an MDL material. But note that the MDL shader can also be used in a native Cinema 4D material: the MDL shader integrates into the shader list, where all other shaders are found, too. You thus can insert an MDL shader wherever Cinema 4D allows a shader selection. This is typically the Texture property of a material.

3.8.2.5 MDL Handbook

The *MDL Handbook* is an in-depth description of the NVIDIA Material Definition Language, providing useful information on the background of MDL, as well as practical guidelines on how to implement custom materials. The Handbook can be found online at: `http://mdlhandbook.com`.

3.8.3 Automatic shader baking

A feature quite convenient for the general workflow using Iray for Cinema 4D, is that native Cinema 4D shaders can be rendered without any input needed from your side. This is achieved by *automatic shader baking:* for every 2D Cinema 4D shader used in the scene, an image texture is created to be rendered with Iray. Every successive rendering then uses the cached texture, if the shader properties did not change in the meantime. By default, each cached texture is stored in the `iray` folder of your local home directory. If not used for at least 14 days, cached textures are deleted automatically.

These and other parameters of the shader baking process can be configured in the Iray preferences. See Shader baking settings (page 38) for a description of the available options.

Shader baking is optimized on performance: multiple occurrences of the same shader only use the single cached texture once. Shader hierarchies are handled correctly. You can create complex layer shaders, which combine different shaders, and Iray for Cinema 4D will take care of their full caching.

Important: Bear in mind that only 2D shaders can be baked. Shaders that evaluate 3D scene data at rendering time, cannot be baked into an image texture of course. Therefore this feature is not supported for shaders like Ambient Occlusion, Subsurface Scattering or MoGraph shaders and alike.

Note: The automatic shader baking feature of Iray for Cinema 4D creates high dynamic range textures and as thus adheres to a linear workflow (page 53). You do not need to care about pre- or post-processing these textures.

3.9 Render Settings

3.9.1 Organization of render settings and presets

Cinema 4D provides convenient tools to create, adjust and manage render settings. This section briefly outlines the basic concept of render setting presets and gives some information on how they can be used efficiently with Iray for Cinema 4D.

Different render setting configurations can be managed in the lower-left corner of the Render Settings window. Each entry here represents a distinct render settings configuration, which includes all the actual parameter values, such as output resolution or render engine. The entries are organized hierarchically, in a tree-like user control, similar to the Object Manager. You can thus drag and drop render setting entries to change the organization structure, for example the order of them or their parent and child relationship. Double-click a render setting entry to rename it.

A right-click on the settings tree opens a context menu including several functions:

- Creating a new render setting entry or a new child entry.
- Removing a render setting entry.
- Loading a render setting entry from a saved preset.
- Saving a render setting entry as a preset.

The context menu can also be opened with the Render Setting button.

When you save a render setting entry as a preset, it becomes available in the Content Browser, ready to use for other scenes as well. Iray includes three render setting presets, optimized for different tasks:

Iray Live Render
: A preset optimized for physically correct live rendering, useful to interactively exploring the scene with photorealistic light and material evaluation.

Iray Live Remote
: A preset for remote rendering, useful for either streaming or queuing.

Iray Interactive
: A preset that prioritize interactivity over physical accuracy, not evaluating full global illumination, useful to quickly explore scene changes.

The crosshair icon on the left of a render setting entry designates whether it is currently active for rendering or not. When rendering to the picture viewer, the render settings with the highlighted crosshair icon are used (independent of the current selection, for example). For live rendering, you can select the render setting preset in the dialog directly. See section Live render presets (page 6) for more information.

Note: Saved presets can also be used to define how MDL material previews are rendered. This is addressed in section MDL material preview render preset (page 25).

3.9.2 Iray renderer settings

This section describes the renderer-specific settings. They can be adjusted in the Renderer tab of the render settings when NVIDIA Iray is selected.

3.9.2.1 End criteria settings

There are a number of termination criteria used to determine when a progressive rendering operation is considered finished. The progressive rendering operation is restarted, for example, when the camera position changes or changes are made to the scene that would be visible in the rendered image. On the other hand, progressive rendering is ended when the quality cannot be further improved; that is if more iterations would not further refine the final image.

The end criteria defined here count for each single rendering. Thus in animation rendering, these criteria apply to each frame individually: every frame gets the same amount of iterations, maximal time or quality percentage.

If more than one end criterion is given, Iray will stop after whichever criterion is met first.

Important: For animation rendering, it is essential to set reasonable end criteria. Each frame renders until one of these criteria is met; if you turn off all of the end criteria options above, Iray will therefore never finish rendering the first frame.

Iterations
: When enabled, Iray stops rendering the frame after the given number of iterations. Higher values result in more refined renderings but increase the total render time for each image. Use this parameter, if you strive for similar quality of different frames. The render time for an iteration depends on the complexity, which most likely varies between frames; as such, terminating rendering by iterations generally results in different render times for individual frames.

  Commonly, if not set too high, each frame will be able to hit this limit. This means that more iterations would yield an even more refined image. However when set to high values, it might occur that frames with a low rendering complexity (having only a small amount of light interaction) cannot be further refined. Iray handles this internally; if the

next iteration level will not alter the final image, rendering will be stopped, as the quality would not improve by allowing more render time. If this is the case even before your end criterion is met, Iray stops the rendering prematurely. Thus if you enable the Iterations criterion and set it to 1000, for example, but a specific frame cannot further be refined after iteration level 500, the frame stops at its "final" level without spending time on needless calculations.

When this criterion is turned off, rendering is not restricted by a certain number of iterations.

Time

When enabled, rendering a single frame stops after the specified amount of maximum render time. As described just above, Iray recognizes if there is no further refinement gain in allowing more time, so very simple frames may be completed before meeting this criterion.

Note: Counting the time begins with the actual Iray render process. Any preprocessing of the scene, like creating cache files or scene export itself, is not taken into account. The total computation time for a frame does therefore most often not coincide with this value but may take considerably longer.

Important: For animations, we do not recommend restricting the render process by the Time criterion. Different frames are most likely variable in complexity. By stopping each frame after a constant time, the quality naturally also varies per frame, which is not wanted in most animation projects.

Quality

At some point the rendering will have reached a quality where subsequent refinements will not have any visible effect on the rendering quality anymore. At this point rendering may stop. Using this parameter you can utilize the internal refinement measurement, mentioned above in the Iterations description, as a termination criterion. Unfortunately it is not intuitive to control render times using this parameter alone. However, it can be useful to additionally employ this criterion with Iterations, for example, if you find Iray spends too long a time with minuscule refinement at high iteration levels.

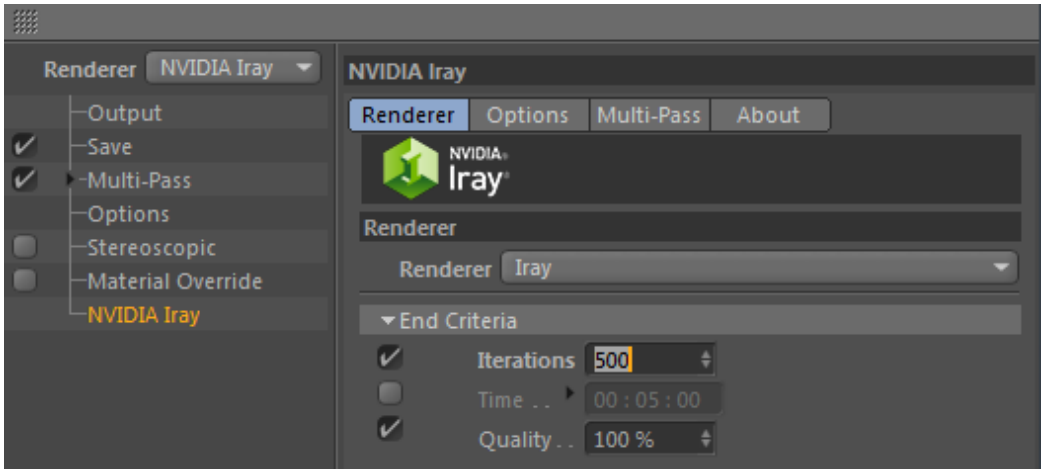


Fig. 3.2 – Typical end criteria settings for animation rendering

In Figure 3.2 (page 29), a maximum number of iterations is specified to achieve the same amount of refinement for each frame. The time criterion is turned off so that differences in frame complexity do not result in different frame qualities (but in different frame render times instead). Additionally, the quality option is set to the maximum of 100 iteration level fewer than 500.

3.9.2.2 General settings

You can optionally enable two additional samplers to improve quality in special scenes, at the cost of render time:

Caustics
: The Iray caustic sampler can be used to improve the quality of caustics. When enabled, Iray will augment the rendering with a dedicated caustic sampler, designed to improve capturing caustic effects.

  See section Improving caustic effects (page 47) for more information on this parameter.

Architectural Sampler
: The Iray architectural sampler can be used to improve the convergence speed of difficult scenes, as such complicated lighting scenarios can then be handled much more efficiently with this sampler. A common scene type that profits from this specialized sampler is indoor architectural visualization, especially if it is mostly illuminated by indirect lighting. One specific example would be a room that is illuminated by light sources placed in neighboring rooms or by outdoor lighting (such as the Sun and Sky model) shining through a small window. As the sampler introduces additional overhead and also only works well when a lot of iterations need to be spent on finalizing the picture it should be avoided for rather simple scenes, or in general mostly directly lit scenes such as outdoor or design visualization.

> **Note:** The Iray live render window does not support the architectural sampler, as stated in Differences between live rendering and production rendering (page 8).

3.9.2.3 Ray control

Max Pathlength
: Limits the maximum number of bounces of light paths to contribute to the result. Decrease this value if the rendering has to be accelerated at the expense of physical accuracy; increase it to allow more light bounces, which results in higher physical accuracy but longer render times in most cases.

  The default value is 20.

3.9.2.4 Filtering

Here you can control filters to decrease artifacts like overly bright pixels or noise from the rendered image. See section Reducing image artifacts (page 62) for a more elaborate description of these filters. For an introduction on how filters work in general, see section What are Iray filters? (page 61).

Firefly Filter
: Enables or disables a filter to reduce bright pixels in the final rendering.

Degrain Filter: Enable

Enables or disables a filter to reduce noise or subtle grain in the final rendering without sacrificing overall sharpness.

> **Note:** Despite this option, the Iray live render window does not support the degrain filter as stated in section Differences between live rendering and production rendering (page 8).

Degrain Filter: Mode

Modes 1 to 3 are working very conservative and should thus be safe to use in general, modes 4 and 5 are considered to be more aggressive and should be used with caution, especially if the scene features fine details in either geometry or applied materials.

Degrain Filter: Radius

This value should be reduced if the filter smooths out edges, and increased if some noise still remains in the image.

Degrain Filter: Blur

Modes 4 and 5 feature an additional setting that limits the influence of neighboring pixels if the brightness is too different.

3.9.2.5 Motion blur settings

With these settings you can enable and control the rendering of motion blur effects. Section Creating motion blur effects (page 73) contains detailed, practically oriented information on how the following parameters influence the visual appearance of motion blur.

Please note that motion blur effects will not be produced in the Iray live render window as stated in the section Differences between live rendering and production rendering (page 8).

Enable

Turns motion blur rendering on or off.

Shutter Open

The relative time offset between two frames at which the motion blur calculation starts. This usually should be 0, meaning that the blur starts exactly at the previous frame. A larger value would make the motion blur calculation start later. Setting it to 0.5 for example, results in blurring an object beginning with its position at one half of a frame.

Shutter Close

The relative time offset between two frames at which the motion blur calculation stops. The default value is 1, so that the blur ends exactly at the current frame. A smaller value would calculate the motion blur until an earlier position. The value of 0.5 for example, blurs an object only to its position at one half of a frame.

Samples per Step
: Number of iterations rendered before the scene is updated to a different time value. With this value you can hence control the quality level for intermediate steps between two frames.

> **Important:** The value Samples per Step must be a power of two, for example 2, 4, 8, 16, and so on. Values that are not a power of two get rounded down to the nearest power of two.

3.9.3 General render options

This section describes the various general options of Iray. They can be adjusted in the Options tab of the render settings when NVIDIA Iray is selected.

3.9.3.1 Image filter

Here you can specify the filter settings Iray uses for anti-aliasing (AA). Generally, the filters provided here blur the image slightly by averaging each pixel with its neighboring pixels. They can be controlled by the weight of neighboring pixels and the total number of neighboring pixels used for averaging:

Type
: The filter type to be used. This affects the weight of neighboring pixels. Possible types are:

  Box
  : Box filtering adheres equal weight to each pixel in the averaging process. Thus it is very fast to compute but not very accurate; it generally leads to poorer quality when compared to Gaussian blur.

  Triangle
  : Triangle filtering reduces the weight of neighboring pixels linearly with their distance. It is intermediate between Box and Gauss in both performance and quality.

  Gauss
  : Gaussian filtering weights neighboring pixels using the Gauss function (also often referred to as *bell curve*). Quality-wise, it gives the best results, but it is slower than the other options.

Radius
: The radius of the filter kernel. This affects the number of neighboring pixels to be used for averaging. Its value is thus proportional to the blur effect; higher values result in stronger blurring of the image. Recommended values are 0.5 for box filter, 1 for triangle filter and 1.5 for Gauss filter.

3.9.3.2 Bloom settings

Here you can optionally add a glow or bloom post effect to the final image. Consult the task-based section about adding bloom effects (page 64) for an in-depth description of how to use the following parameters to control the visual result of the bloom filter.

Note that bloom effects will not be produced in the Iray live render window or in viewport rendering as stated in the section Differences between live rendering and production rendering (page 8).

Enable
: Enables or disables the bloom effect.

Intensity
: Scaling factor for the blurred bright spots; higher values result in brighter glow.

Radius
: Sets the blur radius of the bloom filter, specified as fraction relative to the output resolution. Small values create intense glow in a small area around bright parts of the image. High values will wash out the glow over a larger area, thus also reducing the visual brightness.

Threshold
: Specifies the threshold to cutoff bright spots in the image and blur these in a second step. Higher values will make bright spots more distinct from their surrounding halos.

3.9.3.3 Shadows

Shadow Terminator Offset Mode
: A common and rather well-known problem of physically based path tracers is the occurring of so-called *terminator artifacts*. These artifacts manifest themselves at shadow borders, when the surface normals of polygons are nearly orthogonal to the incoming light direction. In these situations, sharp dark polygon borders can become apparent, similar to the image below. This is not an implementation problem, but an inherent limitation of combining smoothly shaded surfaces with ray traced shadows. The extent of these artifacts can be reduced through finer tessellation of the mesh. However, in many cases this is either not wanted or practical to do.

  Another common technique to reduce shadow terminator artifacts is to slightly offset the polygon normal for the self-shadowing calculation. This can be enabled by setting the Shadow Terminator Offset Mode option.

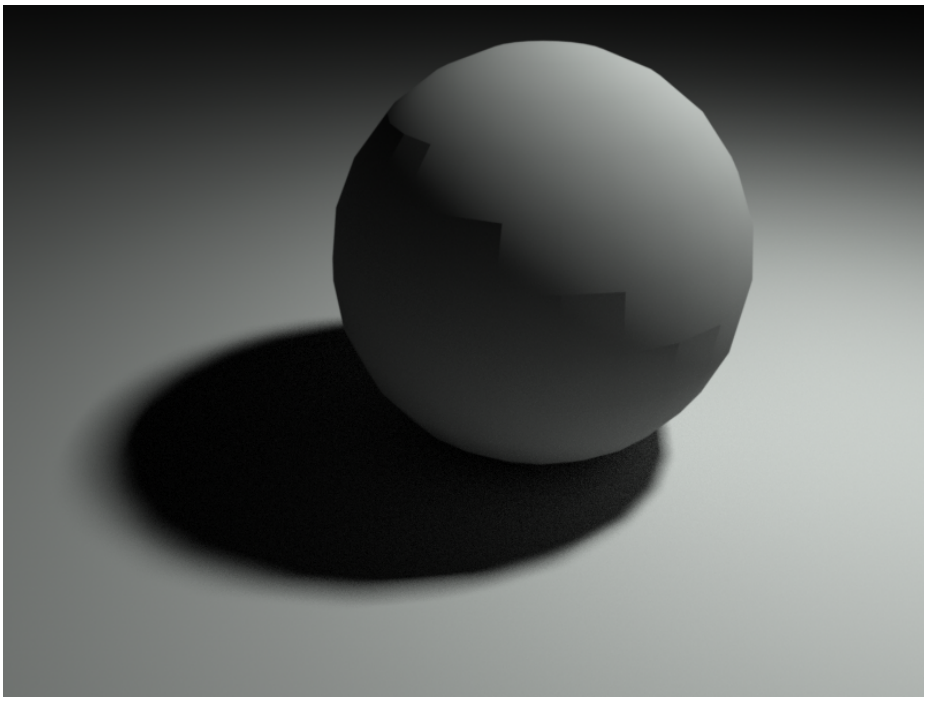


Fig. 3.3 – Terminator artifacts: where polygon normals are nearly orthogonal to the incoming light direction, surface smoothing can fail so that the dark polygon outlines become visible.

3.9.4 Multi-Pass settings

3.9.4.1 Light path expressions

Here you can select, add or remove any number of additional passes that should be rendered individually. Iray for Cinema 4D includes several common pass presets to choose from. Section Pass presets (page 57) contains more information on these.

Additionally, you can define own passes using custom light path expressions. An introduction is given in the LPE guide (page 58). The Edit button opens the LPE Editor to conveniently create custom light path expressions from basic elements.

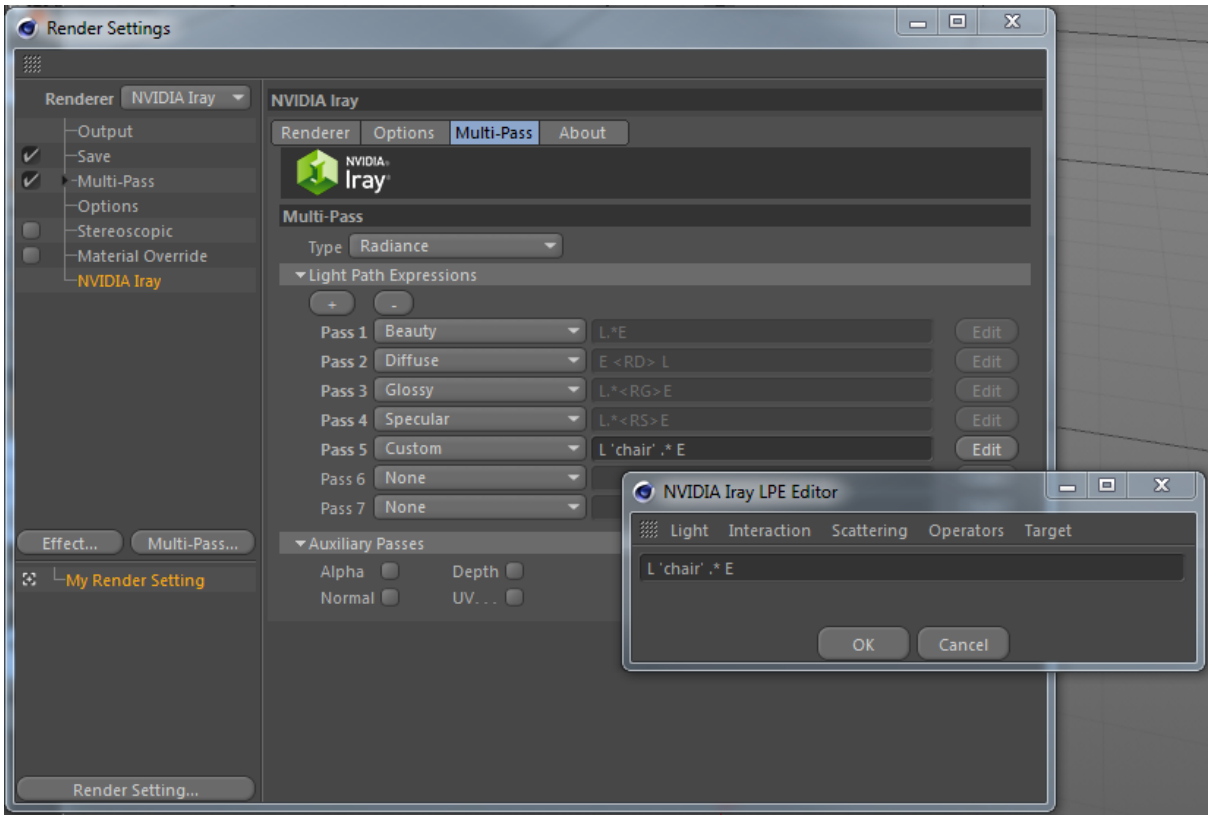


Fig. 3.4 – The user interface for Multi-Pass rendering. Here, a few basic passes are rendered individually, and a custom light path expression to render a "chair" object separately is provided with the LPE editor.

3.9.4.2 Auxiliary passes

Additionally to the LPE pass presets, described in the previous section, Iray for Cinema 4D includes four *auxiliary passes*. These are helpful, often-used passes that can not be formulated as light path expressions. The following auxiliary passes are available:

- Alpha
- Normal
- Depth
- UV

In the task-based section about multi-pass rendering (page 58), we provide more information about these passes.

3.10 Preferences

Application-wide Iray settings, independent of a current scene or render settings, can be adjusted in the native Cinema 4D preferences dialog. The parameters can be found here:

1. Open the standard Cinema 4D Preferences dialog, either by selecting [Edit ▸ Preferences] from the menu or via default shortcut `Ctrl+E`.
2. On the left hand side in the preferences dialog, select NVIDIA Iray.

3.10.1 Resources settings

This section describes the settings for the hardware resources to be used for rendering. They can be adjusted in the Resources tab of the Cinema 4D preferences when NVIDIA Iray is selected.

Render Resources

You can switch between rendering using local resources and using a remote server. Local resources are typically given by the processor (CPU) and the NVIDIA graphics hardware (GPU) of the machine you are currently working at. The available remote resources are:

- An NVIDIA Quadro Visual Computing Appliance (VCA), a network-attached appliance that harnesses the power of the highest-performing NVIDIA GPUs. More information about VCA can be found at: http://www.nvidia.com/object/visual-computing-appliance.html
- Cloud rendering with the NVIDIA Iray Server, a software solution that provides distributed Iray rendering across remote machines, without the need to install any other application. Get information on Iray Server at: http://www.nvidia.com/object/iray-server.html

3.10.1.1 Local render resources

Use CPU

While you will get the most out of Iray by using powerful graphic processors, it is also able to utilize available CPUs for rendering. With this setting you can enable or disable employing CPUs for rendering. Using both CPU and GPU hardware is often called *hybrid rendering*.

Threads

If you need to balance processor load, you can change the number of threads that Iray uses for rendering. The number of threads will be taken into account when you start a new rendering process.

> **Note:** If all available threads would be used for rendering, no resources would be available for the user interface and any input events. The number of threads you can set here is therefore limited to the number of available hardware threads given by your CPU, reduced by one. For example, an 8 core CPU with Hyper-Threading allows to use up to 15 threads for rendering with Iray. This way, it is guaranteed that at least one thread is available for other tasks, preventing the application from becoming unresponsive.

Use all GPUs

Iray can leverage NVIDIA CUDA capable graphic processors (GPUs)to render photorealistic images in a short amount of time. Enable this option if you want to use all available GPUs of the system for rendering with Iray. We highly recommend using Iray in connection with at least one powerful CUDA capable graphics card.

If you have more than one GPU, you can selectively choose which ones to use for rendering. This can be useful to restrict Iray to use only the most powerful GPU for

rendering by deactivating a separate graphics card to speed up viewport display. An example of this setup is displayed in Figure 3.5.

Note: If a system does not have CUDA capable graphics cards it will automatically fall back to CPU based rendering, even it the option Use CPU is turned off.

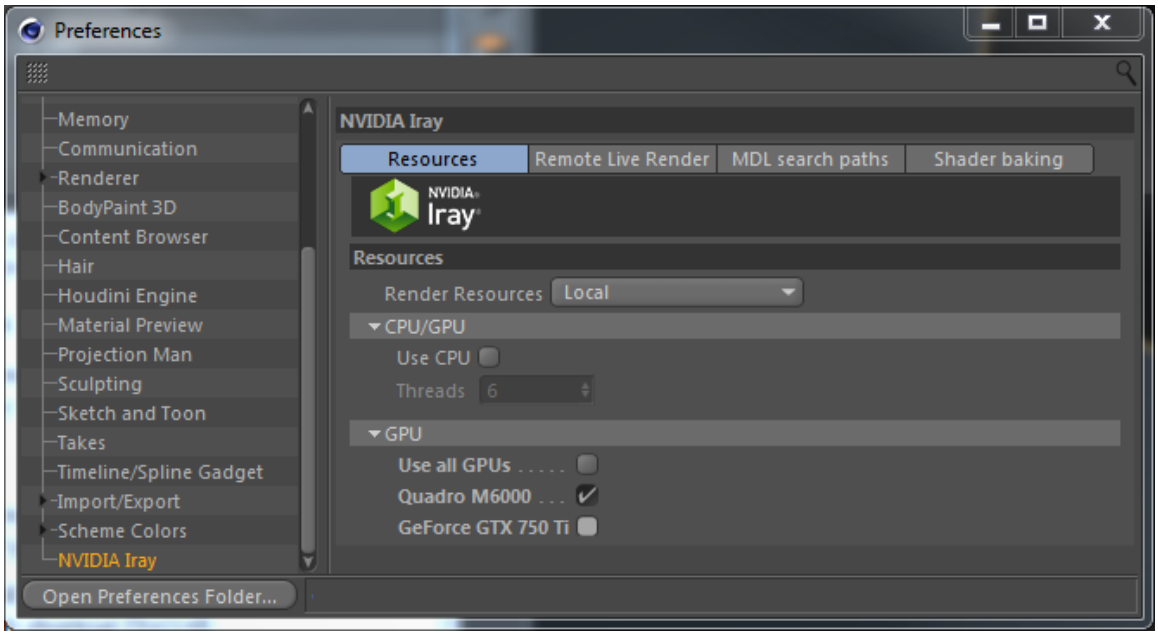


Fig. 3.5 – Individual settings for the hardware resources Iray uses. Here, the most powerful graphics card is dedicated for rendering, and other resources will not be used by Iray.

3.10.1.2 Remote render resources

The different settings of remote render resources and guidelines on how to connect to a render service, are covered in section Iray Streaming (page 77).

3.10.2 Remote live render settings

When using the Iray live render window, you typically prioritize interactivity over almost anything else. For remote rendering, it hence might be useful to provide separate communication settings that affect interactivity. In the Remote Live Render tab of the Iray preferences you therefore have distinct control over these parameters, only valid for live rendering:

Mode
: Streaming or synchronous display

Format
: The output format, video (H264) or image (Lossless, PNG, JPG)

If a video format is selected, you can also define performance relevant parameters: the bit rate, maximum lag and frame rate.

3.10.3 MDL search paths

Iray for Cinema 4D searches in various directories, called *search paths*, for the MDL materials it displays in the user interface. To add custom paths that Iray for Cinema 4D should in-

clude in the search for materials, enter them in the MDL search paths tab of the NVIDIA Iray preferences.

A detailed description on the default search paths, naming conflict precautions and the order of loading priority is given in section MDL search paths (page 24).

See section MDL material (page 23) for general information on the MDL material and its integration into Cinema 4D.

3.10.4 Shader baking settings

Here you can control the automatic shader baking (page 26) feature of Iray for Cinema 4D. The following parameters are available:

Clear cache now
: With this button you can remove the cached data files. All files in the currently specified cache path are being deleted.

Resolution
: Specifies the texture resolution to use for baking Cinema 4D shaders into image files. Higher values increases texture detail at the cost of rendering speed. Lower values can cause texture filtering artifacts to appear.

  The default value is 512, hence textures with a resolution of 512 x 512 pixels will be created.

Clear cache on exit
: Turns automatic deletion of unused cached shaders on or off. Iray for Cinema 4D automatically evaluates how much time elapsed since each of the cached shaders were used the last time. For shaders that were not rendered in at least a given number of days, the baked textures are deleted upon exiting Cinema 4D. You can set the number of days with the parameter Keep files (in days), as described below.

Keep files (in days)
: With this setting you can define the number of days after which the textures of cached shaders will be deleted automatically. This accounts for each shader individually: if you have a Cinema 4D noise shader and a gradient shader, they each have their own date for the last usage. Hence if only the noise shader is unused for the given days, Iray for Cinema 4D will delete the cached texture for the noise shader but keeps the gradient shader cache.

  Note: This setting is only available if Clear cache on exit is turned on. Otherwise, cached shaders will never be deleted.

  The default value is 14 days.

Use custom cache path

When this option is turned on, you can define a custom directory where to store cached shader textures. When turned off, Iray for Cinema 4D stores the cache to the `iray` folder in the home directory of the local system.

> **Note:** This setting can be useful for collaboration work: you can set the cache path to a network directory, so that multiple users have access to a global cache repository. Working on the same project or employing a common set of Cinema 4D materials, users will experience an increase in rendering performance.

4 Working with Iray for Cinema 4D

In this chapter we address the most common tasks using Iray for Cinema 4D. We explore the underlying concepts that should be understood to get the best results with Iray, and give guide lines on how to control various effects and hints where to look at if the results still turn out unexpectedly.

4.1 Principles of physically based rendering

A physically based render engine like Iray strictly adheres to the physical laws of light and materials. As such, Iray is capable of generating truly photorealistic images. A convenient benefit of a physically correct renderer is that creating even complex real-world lighting effects, like full global illumination or caustics, does not require expert knowledge or high scene setup effort.

Nevertheless, to get good results, there are few basic rules about creating scenes for physically based rendering:

- Your virtual scene must reflect the actual size, given in real-world units of measurement. This is discussed in the next section.
- Setting up a scene is less about specifing the information what you want to see, but much more about creating a scene how it would be in reality. It is often helpful to think of a real set with real lights and cameras, as in photography, when creating scenes for physically based rendering.
- Material properties must be physically plausible. While the built-in NVIDIA MDL Material ensures this automatically, you need to respect this when working with the native Cinema 4D materials. More information can be found in section Cinema 4D material (page 22).
- Simulating actual light distribution, Iray calculates the interaction of virtual photons with the scene. The accuracy of this simulation depends largely on the amount of photons and the number of allowed bounces for each photon. Higher numbers of these parameters of course increase the calculation effort and thus also the render time. Iray renders progressively, which means that as more and more photons contribute to the result, the more refined it gets. As there is no strict upper limit on photon count, it is up to you to define end criteria (page 28), which specify when rendering is considered as complete and stopped. To control the number of allowed bounces of a photon, Iray provides the Max Pathlength (page 30) parameter.

4.1.1 Units of measurement considerations

Internally, Iray works with virtual "Iray units", not rigidly correlated to real units of measurement like meters. Setting up your scene however, you provide both absolute sizes during modeling, and the unit of measurement settings of Cinema 4D. While conventional renderers are typically rather independent of absolute scene scale and units, this is crucial for physically based rendering. Iray for Cinema 4D automatically converts the scene scale to Iray units,

respecting your scene unit settings. This accounts for all unit-relevant quantities, for example the size of an area light, or the length light enters into an object with a subsurface scattering material (SSS). The only thing you need to keep in mind during scene setup is to provide the actual real-world units and sizes; a correct scene scale is important for Iray, as it calculates the light interactions using these values to produce realistic results.

There are several ways how you can influence units or global scene scale; these methods are described in the following.

To change in which unit of measurement sizes are displayed, follow these steps:

1. Click [Edit ▸ Preferences] to open the Cinema 4D preferences.
2. Select Units on the left hand side to display the unit settings.
3. Set the attribute Unit Display to the unit that you want to use.
4. Optionally, set or clear the option Auto Convert Units. If turned on, Cinema 4D does not strictly adhere to displaying the unit you chose, but will convert it automatically if the value is several orders of magnitude greater or smaller. If Unit Display is set to kilometers, for example, but an object has a size of 2 centimeters, its displayed size would be 0.0002 km. In the user interface, Cinema 4D shows floating point numbers with a precision of three digits after the decimal mark, resulting in a displayed size of 0 km. When Auto Convert Units is on instead, Cinema 4D converts values like this to keep the value meaningful, with a precision of two digits after the decimal mark. Thus in this case, the value is displayed as 0.02 m, despite the unit display setting.

Note that by changing the unit display with the procedure above, the actual sizes do not change. Thus if you have an object with a size of 2 cm and then change the Unit Display to meters, the object size is now displayed as 0.02 m; so this is just a change in units, not in actual spatial size. The render result is hence not influenced by this setting.

To instead globally scale your scene, maintaining the current unit of measurement, follow these steps:

1. Click [Edit ▸ Scale Project]
2. Enter the scale conversion you want to apply globally, by specifing a factor and a unit you want to convert from, as well as a destination unit and its factor.

Note that by using this approach, the scene size gets scaled (but the displayed units are still dependent from the Units settings above). A simple example: you have an object with a size of 2 m. If you apply Scale Project with a current scale of 1 Meters and a target scale of 1 Centimeters, the object now has a size of 0.02 m. This makes a great difference for a physically correct renderer like Iray: a two meter thick object made of glass, for example, of course renders quite different from a two centimeter thick one.

There is another way to change the project-based scale: in the project settings, accessible in the Edit menu or by pressing Ctrl+D, there is an explicit Project Scale parameter. It works identical to the previous procedure, actually scaling the global size and thus affecting render results.

4.2 Creating realistic lighting effects and shadows

In this section you will learn how to control common lighting effects to achieve authentic results using Iray. We show how using real-world images for environment lighting (page 42) in-

creases quality, and how you can improve caustic effects (page 47), itself contributing strongly to the perceived realism of renderings. Two additional sections provide information about how you can purposely deviate lighting effects from true physical accuracy. It can, for example, be useful to have full artistic control over environment shadow properties (page 45) or to simulate the ubiquitous ambient occlusion look (page 44).

4.2.1 Image-based lighting

Image-based lighting, often abbreviated as *IBL*, is a common approach to illuminate a scene by a high dynamic range image (*HDR*) taken from a real-world environment. Such an image is able to capture and transport the original light energy and colors from the physical world to the renderer. Most commonly, this is realised by mapping the image onto a virtual infinite dome or sphere surrounding the scene.

The Iray Environment provides a native solution for image-based lighting. It delivers results with superior quality and optimized performance compared to traditional ways, like using global illumination (*GI*) techniques combined with explicitly modeled geometry surrounding the scene with the IBL image texture assigned to its surface.

Follow these basic steps to set up a scene that makes use of image-based lighting:

1. First, if your are starting with an existing scene, be sure to disable any other sources of environment lighting. These may include the Cinema 4D objects like Sky and Physical Sky as well as traditional light sources such as Sun and Infinite. These can alter the illumination in an undesired way when combined with image-based lighting. If your IBL texture includes a bright sun for example, having an additional sun light which contributes to the environment lighting, will reduce the credibility of a realistic rendering. We thus recommend to use only a single source of environment lighting when employing IBL.
2. From the Iray menu, click Iray Environment. This adds an object that controls environment lighting to your scene.
3. Select the Iray Environment object in the object manager.
4. In the attribute manager, set the Environment parameter to Image-Based Lighting mode.
5. Choose an Environment Texture for the environment by selecting an image from the file system. We highly recommended to use a high dynamic range image (HDR) for environment lighting.
6. You should now see your scene lit as if it were sourrounded by the image content, similar to what is shown in Figure 4.1 below.
7. The overall intensity of the environment lighting can be adjusted using the Intensity parameter. Figure 4.2 (page 43) and Figure 4.3 (page 43) show how varying the intensity affects illumination.

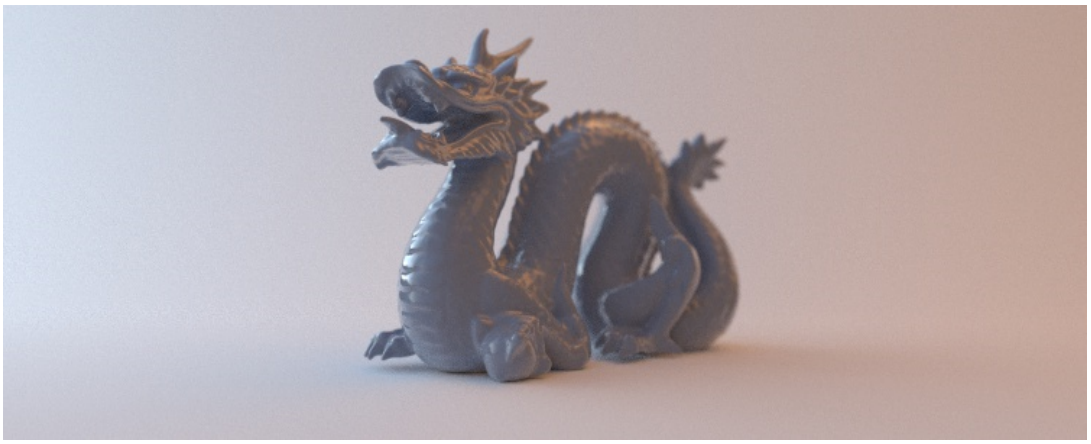


Fig. 4.1 – Using image-based lighting to illuminate a scene

Fig. 4.2 – Decreasing IBL intensity reduces luminance

Fig. 4.3 – Raising IBL intensity increases luminance

However an infinite dome is used most commonly, Iray features a flexible environment dome as an extension over the conventional infinite spherical environment. Depending on the parameter Dome Type, the dome can also be of finite size, where camera movements inside the dome provide a different environment lookup.

For the finite size domes, the projection is computed by assuming a virtual camera position that is defined by the Dome Position parameter. So, as an example, if the real environment is an approximately box-shaped room and the camera position to capture the environment

map is known, then those values can be used directly to set up a faithful representation of the environment. Also, the finite size domes support a scale factor for the texture projection on the ground. This can be controlled with the Texture Scale parameter of the Ground settings (page 14).

Consult the IBL parameter descriptions (page 12) above for additional information on the finite size dome settings.

Good HDR environments are likely to contribute lighting information that results in very realistic final results. When using image-based environment lighting, we recommend the following:

- Avoid HDR images where the sun is not as bright as in real life. Otherwise the resulting image looks washed out. To determine whether the sun is bright enough, open the HDR with an image tool and take a look at the value of the sun. A sun of realistic size should be a few thousand times brighter than other parts of the image. If its intensity value is anywhere below 1000 then the sun is too dark, and the resulting light simulation will look dull and lack the necessary contrast in the rendered image.
- Avoid low resolution HDRs, as the aliasing of small bright spots in the environment map will appear in the final picture and yield artifacts, especially in the shading or lighting of objects. Apply the optional blur settings to low resolution HDR images.

Note: The mode Sun and Sky provides realistic environment lighting without an image texture. Hence if you just need authentic outdoor illumination, you can find a detailed description on this in section Sun and sky settings (page 9).

4.2.2 Faking an ambient occlusion effect

A very common and well known technique to enhance an object's perceived plasticity is *ambient occlusion* (*AO*). A non-physical rendering technique, ambient occlusion approximates global illumination only very roughly. The basic concept is to sample a given surface point, measuring how much of its surrounding space is occluded by geometry within a specified radius. This measure then acts as a weight, estimating how much light can reach this surface point.

Physically correct renderers like Iray calculate full global illumination accurately and hence do not include AO approximation. However, for artistic reasons, it might be of interest to create the distinctive AO look anyways. Since the physical correctness originates from the simulation of actual light ray bounces, limiting the number of bounces can effectively reduce realism to create an AO like effect.

To prepare a scene setup for faking an ambient occlusion effect with Iray for Cinema 4D, follow these steps:

1. Create or load a geometry object that should be rendered.
2. Create a ground plane, able to catch shadows.
3. Create a new NVIDIA MDL Material.
4. Double-click the new material to open the Material Editor. Click the browse button to select a material from the available MDL materials. From the nvidia::core_definitions, choose Simple Diffuse.

5. Change the material color property to 100 % white.
6. Assign this diffuse material to both the ground plane, as well as any other object in the scene: drag the material and drop it on the objects either in the viewport or in the object manager.
7. For the ambient occlusion look, a uniform white environment light is needed. Create a sphere and increase its size so that it surrounds the whole set. Your scene (including the camera) should now be completely inside the sphere.
8. Since Iray respects normal vectors, the sphere will only be rendered as expected if its normal vectors point inwards. Convert the procedural sphere to a polygon object by selecting the sphere and pressing C. With the sphere still selected, reverse it normals in the menu: [Mesh ► Normals ► Reverse Normals].
9. To use the sphere as a light source, create another MDL Material. From the nvidia::core_ definitions choose Diffuse emission for the new material. Set its intensity parameter to 3.
10. Assign the new emissive material to the surrounding sphere.
11. The spherical environment needs to be the only light source, so make sure that the scene does not include other light sources. Also, from the render settings, select Options on the left and turn off the Default Light setting.

The key for faking the AO effect is now to actually reduce the light bounces, which create physically accurate global illumination:

1. Open the Iray Render Settings.
2. On the left hand side, select NVIDIA Iray to edit the Iray settings.
3. Click the Renderer tab.
4. Max Pathlength limits the number of light bounces. With it, you can control the trade-off between physical accuracy and needed render time. While the default value of 20 is intended as a reasonable compromise between the two, reduce the max pathlength to 2 to fake AO.
5. Render the scene to the picture viewer. You should now see the scene illuminated with an effect quite similar to ambient occlusion, displayed in Figure 4.4.

Fig. 4.4 – Faking ambient occlusion with Iray

4.2.3 Soft physical sky shadow

To conveniently illuminate a daylight scene in a realistic way, Iray includes the Iray Environment object. With its Sun and Sky mode, you can define the lighting situation with the parameters outlined in section Sun and sky settings (page 9). Initially, the shadows may appear somewhat too hard, when the physically correct values are used. Figure 4.5 shows an object, rendered using Sun and Sky with a Size value of 1, which is the physically correct sun disk size.

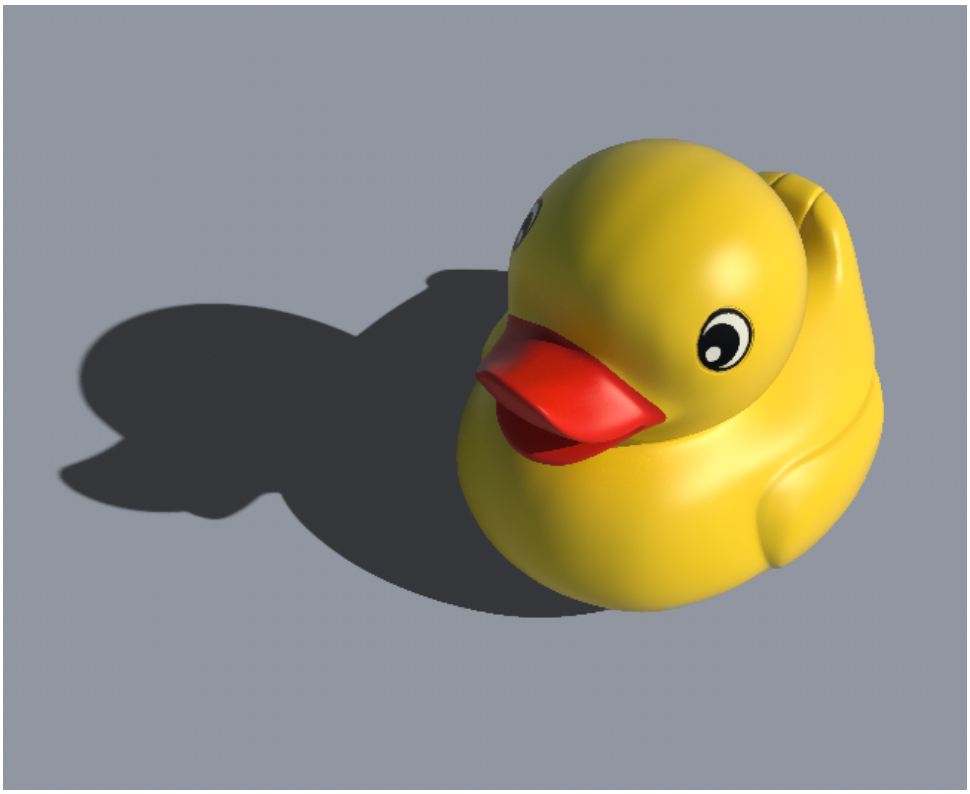


Fig. 4.5 – Environment lighting with physically correct sun and sky settings

Note: A shadow catching ground plane is needed to see the shadows in the rendering. But you do not have to model it with actual geometry: the Iray environment object includes a Ground precisely for that.

To soften the shadows, increase the Size value; similar to an area light, a larger sun disk size also yields softer shadows:

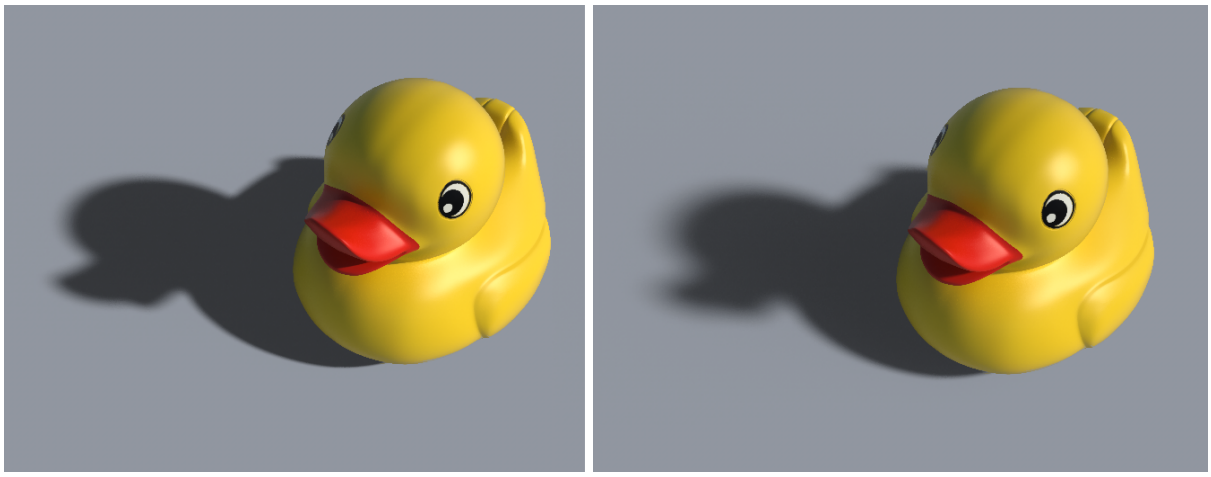


Fig. 4.6 – Higher Size values create softer shadows. The left image was rendered with a sun disk size of 5, the right image with an exaggerated size of 10.

Besides the softer shadows, you can notice as well:

- The overall light intensity remains the same: a larger sun disk neither simultaneously increases the luminance, nor does it reduce it (as distributing a constant intensity over a larger area would do). Instead, Iray will still calculate the overall environment light intensity accurately, just as in the previous image. The Size parameter controls the area of the sun light, which affects the visible sun disk size, the softness of shadows and to a certain extent also the variance in incoming light direction (depending on the Dome Type). To actually adjust the intensity, use the Intensity parameter.
- Very soft parts of the shadow are of course less dark than the full shadow itself. However, the intensity at fully shadowed parts is still the same as before. This general shadow color can be adjusted with the Shadow Intensity parameter of the Ground settings (page 14) See the Figure 4.7 for a comparison between different shadow intensity values.

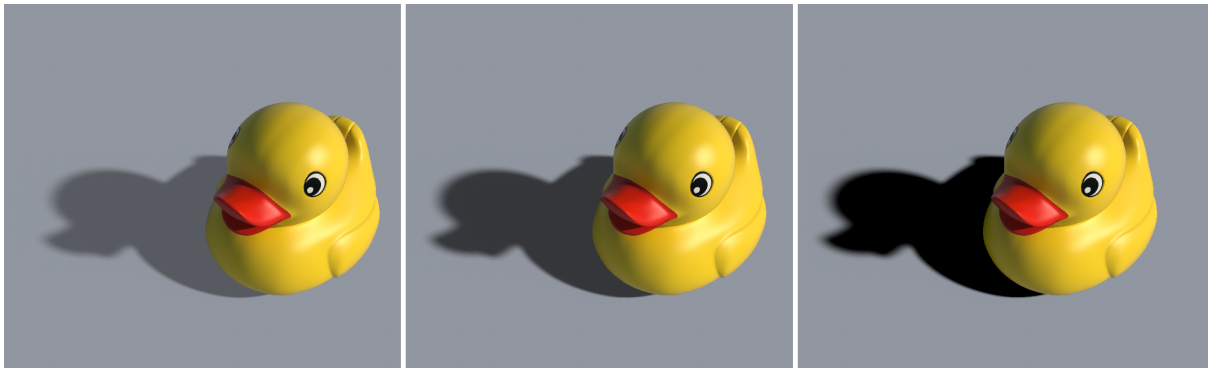


Fig. 4.7 – Control the shadow intensity with the Ground object. The images were rendered with Shadow Intensity values of 0.7 on the left, the default value of 1.0 in the middle, and 1.3 on the right.

Note: Adding Haze also affects shadow properties. A high value of haze simulates a rather diffuse lighting, resulting in less intense shadows.

4.2.4 Improving caustic effects

In reality, light can gather locally when either reflected or refracted by a curved surface. The distinctive bright light patterns caused by these effects are called *caustics*. Typical examples include objects made of glass, or the irregular water surface in a pool. Due to a change in the materials *index of refraction* (*IOR*), the light rays get deflected as they enter and leave the glass or water medium. According to the IOR and the surface geometry, some rays may get scattered when they leave the object, while others gather to form bright caustics. Another common example is a curved reflective surface, like a ring on a table. Light reflects from the inside of the ring onto the table, creating a distinctive nephroid pattern of high brightness. When the bright patterns result from refracted light, they are called *diacaustics*; if they originate from reflection instead, they are called *catacaustics*.

Evaluating physically based global illumination, Iray is able to render authentic caustic effects, greatly improving the credibility of rendered scenes that would also feature caustics in reality. However, Iray generally goes for a compromise between caustic detail and reducing noise, which may result in somewhat poor caustics.

If you want to create images with a focus on detailed caustics, you can use the built-in caustic sampler. The Iray caustic sampler can be turned on in the Renderer tab of the Iray render settings by setting the Caustics option. It was mainly designed for typical turntable scenes, but is not limited to them. It thus performs best under the following conditions:

- The scene geometry is (mostly) in view.
- The caustics are (mostly) in view.
- The scene is positioned on top of a groundplane or large surface.

Figure 4.8 shows a typical application of the Iray caustics sampler. Figure 4.9 (page 49) shows the same scene, rendered with the caustics sampler turned off:

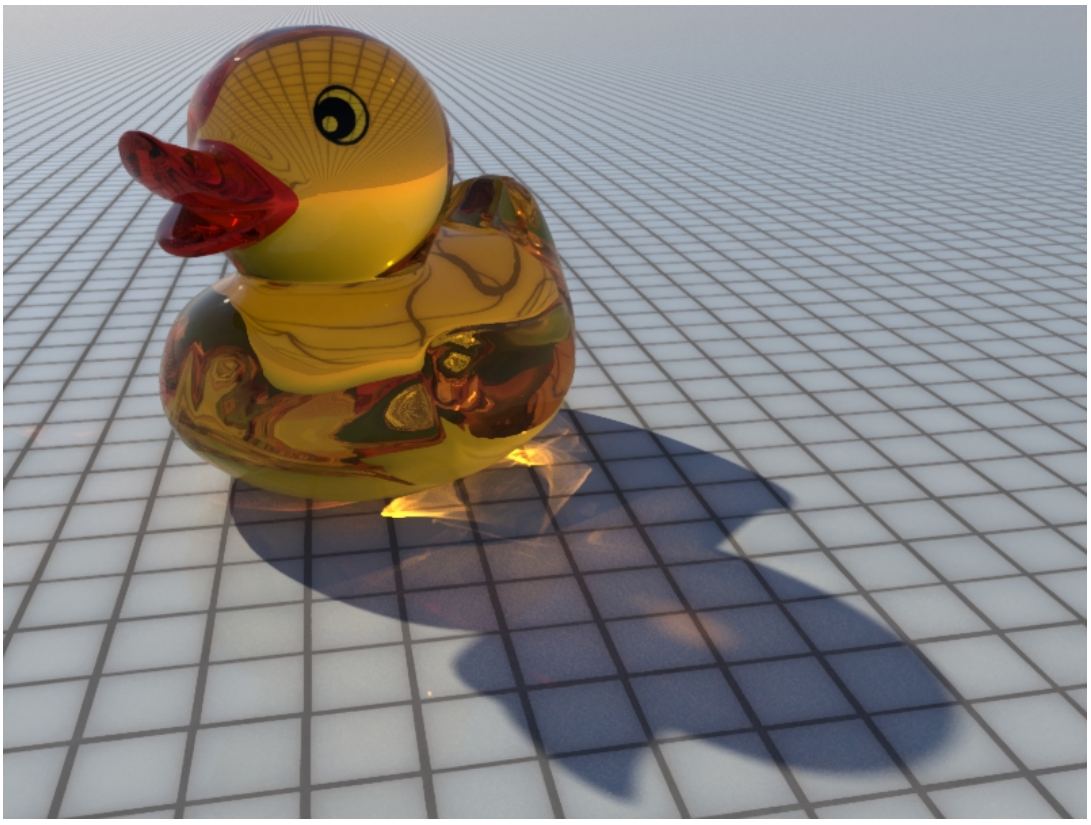


Fig. 4.8 – High quality caustics using the Iray caustics sampler

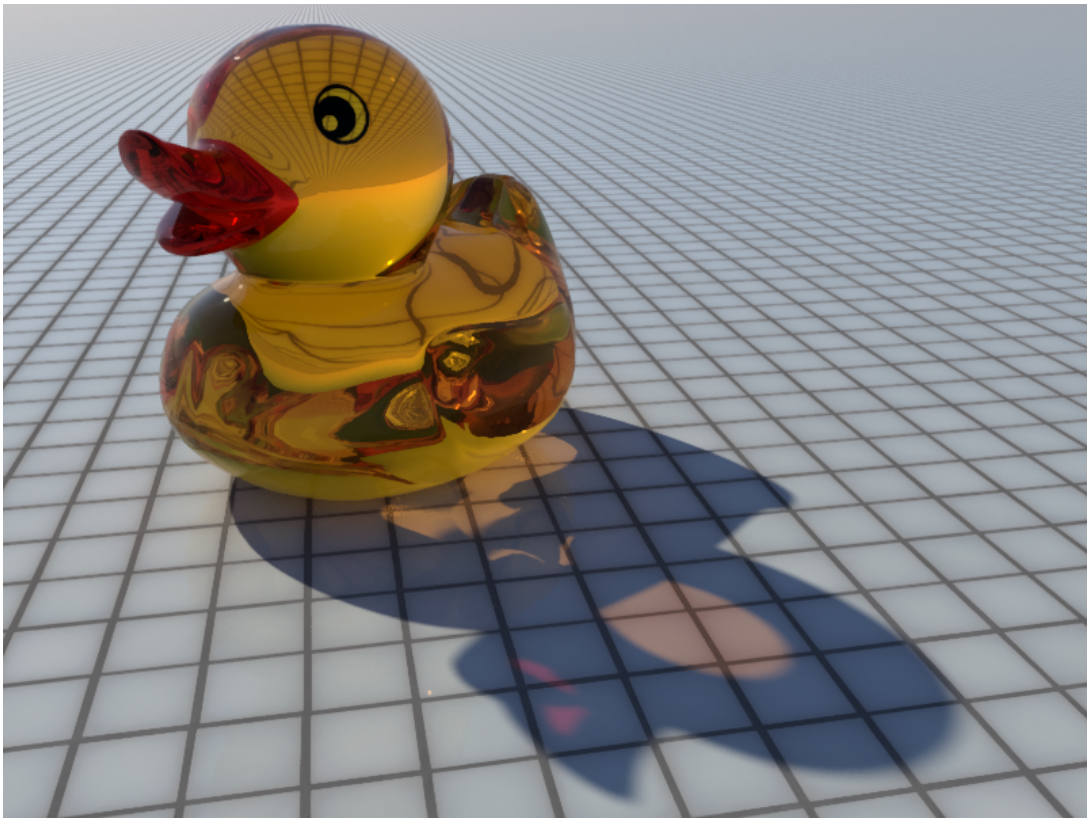


Fig. 4.9 – Without the caustic sampler, the rendered image lacks the distinctive caustic patterns

Note: To catch caustics on the Iray Environment Ground (page 14), you must enable Reflectivity in the Ground parameter settings.

4.2.4.1 Caustic sampler performance considerations

Since the caustic sampler will generally increase render time, it is disabled by default. With respect to rendering performance, a few things should be kept in mind when employing the caustic sampler. First of all, it should only be enabled in situations where the caustics can actually be improved. The caustic sampler can fail to improve caustics under the following conditions:

- Only a small part of a larger scene is in view, for example if the camera zooms in on a single item in a detailed scene.
- The camera zooms in on a small part of a much larger caustic.
- The caustics are seen through mirrors or windows; these can not be improved. The caustic sampler only improves directly visible caustics.

Second, the caustic sampler can be used in combination with the Architectural Sampler (page 30) to help rendering caustics more efficiently in difficult scenes and lighting situations. However, it is not always beneficial to enable both samplers at the same time as there is additional computational overhead involved that can outweigh the benefits.

4.3 Camera

In real-world photography, camera properties greatly affect the captured image. Trying to create authentic images, a physically based renderer like Iray must thus supply settings that account for the real camera properties to produce the same visual effects. Iray for Cinema 4D provides the Iray Camera Tag to control these properties. In this section we describe the possibilities of the camera tag and give a general introduction on both the concept underlying depth of field and tone mapping, as well as guidelines on how these effects can be produced and controlled with Iray for Cinema 4D.

4.3.1 Creating depth of field effects

Strictly speaking, a lens of a real camera has only one exact distance, at which an object truly is in focus. Any deviation from this distance will cause the object to be out of focus. The percieved blurryness of a defocussed object increases proportionally to the distance from the camera's focal distance. However, as both our visual perception as well as the display device are far from infinite in resolution, there is a tolerance range beyond the exact focal distance, in which objects still appear in sharp focus to us. This range of focal tolerance is commonly known as *depth of field* (*DOF*). A larger depth of field therefore yields sharp objects, even at long distances, whereas a small depth of field considerably blurs very near or very distant objects.

Striving to create credible images, one also wants to simulate DOF effects in computer graphics. Iray for Cinema 4D comes with the Iray Camera Tag with which you can adjust this effect to simulate a variety of camera properties. Follow these steps to create a rendering that features DOF effects:

1. Create a scene (or use an existing one) in which objects are distributed among a considerably varying distance along the viewing direction. In terms of a new scene with Cinema 4D default settings, place some objects near the world origin and some objects along the positive Z axis from the origin up to a distance of at least 6,000 cm.
2. Create a standard Cinema 4D Camera object, either via the tool bar or the Create menu.
3. Assign the camera to the active view, either by double-clicking the icon of the camera object in the Object Manager or by selecting the camera in the viewport menu at [Cameras ► Use Camera].
4. Set the camera object position to X: 400, Y: 300, Z: 0 and the camera rotation to H: 20°, P: -10°, B: 0°. The field of view of the camera should now cover many objects with a wide variety in their distance to the camera.
5. Add a light source to the scene, for example an Iray Environment object from the Iray menu.
6. In the Render Settings, select NVIDIA Iray as renderer.
7. Render the scene to either the picture viewer or by using the Iray live render window. You will notice that currently every object appears completely in focus, despite the huge difference in their respective distance to the camera.
8. Select the camera in the Object Manager.
9. From the Iray menu, select Iray Camera Tag to assign it to the selected camera object. Alternatively, all the Iray tags are also available in the context menu of the Object Manager.
10. Select the Iray camera tag in the Object Manager.
11. In the Attribute Manager, click the Depth of Field tab.
12. As you can see, the default value of the lens Radius parameter is 0, hence the depth of field is infinite. This means that every object appears in focus when rendering. Set this value to 10, for example, and render the scene again. You should now see that objects near the camera and objects very far away from the camera are somewhat blurred, as they now are out of focus, similar to Figure 4.10. With the lens radius value you can influence the strength of the depth of field effect; the analogy in photography is the aperture (usually specified as the f-number, the ratio of lens focal length to aperture diameter). We recommend to keep this value in the order of a couple of centimeters (expressed in the current scene units), otherwise the scene may come out unrealistic and be perceived as a minature. Figure 4.11 (page 52) shows how a change in lens radius affects the DOF effect.

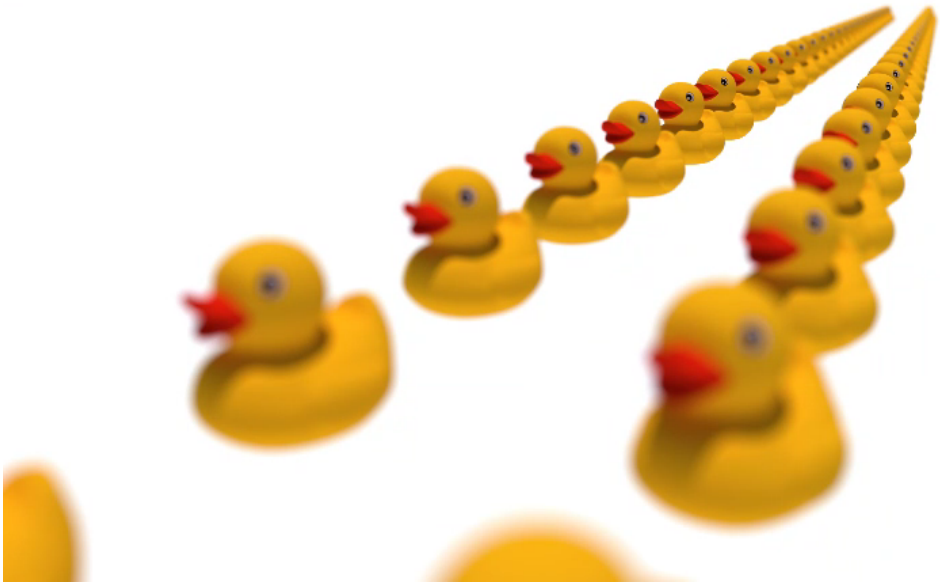


Fig. 4.10 – Limiting the depth of field blurs objects with a large distance from the focal point. Simulating this effect can be achieved with the Iray Camera Tag.

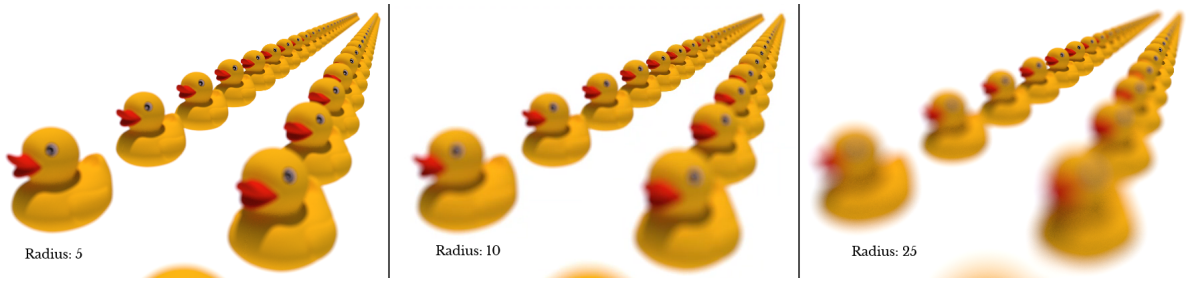


Fig. 4.11 – With the lens Radius parameter you can influence the strength of the DOF effect.

Per default, the Iray camera tag evaluates the DOF effect using the Focal Distance of the camera object it is assigned to. You can also specify a custom focal distance for a given tag:

1. In the Object Manager, select the Iray camera tag.
2. In the Attribute Manager, click the Depth of Field tab.
3. Disable the option Use Camera Distance.
4. Enter the desired Focal Distance, given in scene units.

Figure 4.12 and Figure 4.13 (page 53) visualize different focal distances when rendering with DOF effect.

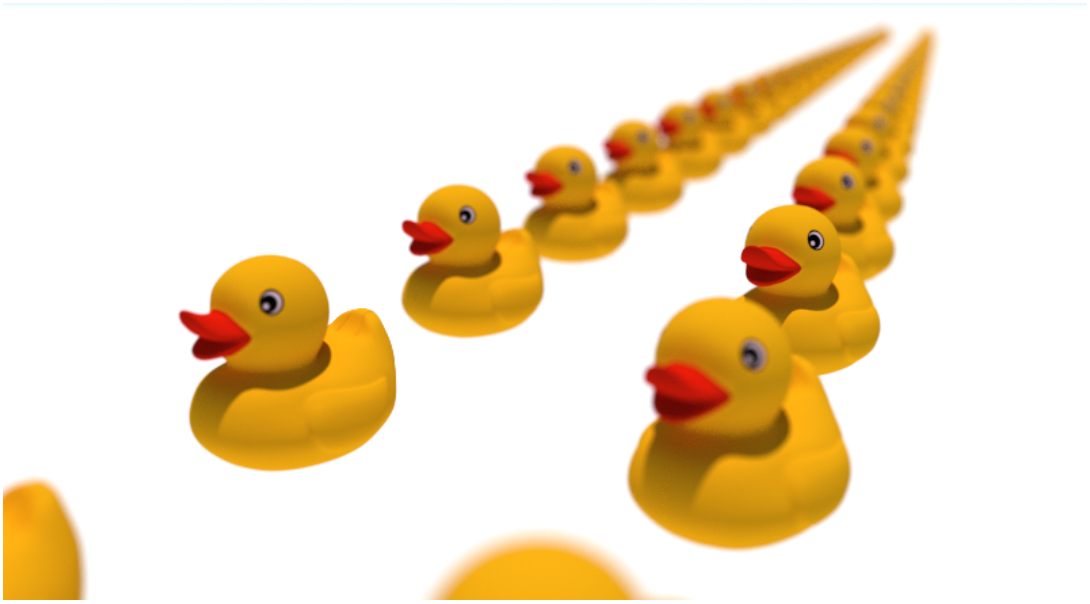


Fig. 4.12 – Reduced focal distance with a custom value specified in the Iray Camera Tag. Objects near the camera appear in focus while very distant objects get blurred considerably.

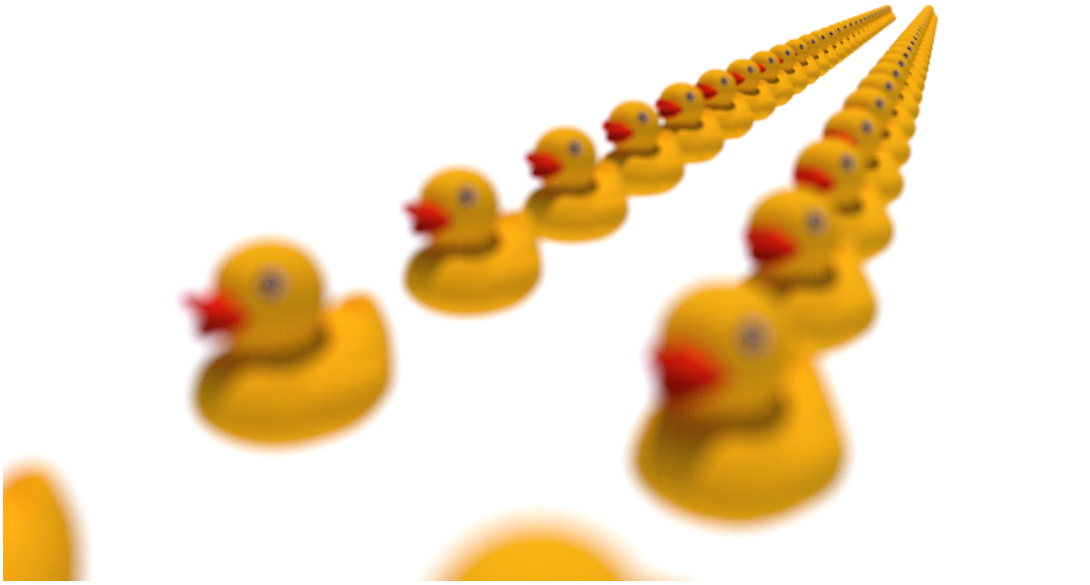


Fig. 4.13 – Increased focal distance with a custom value specified in the Iray Camera Tag. Distant objects appear in focus while very close objects get blurred considerably.

For a description of the available parameters, also see section Depth of field (page 18) in the Iray user interface chapter.

Note: A common workflow is to add depth of field effects to the rendering in a post-production environment instead of rendering it in the beauty pass directly. To achieve this, a separate depth pass should be rendered, which can then be processed in compositing. For this, do not use the native DOF map settings of the Cinema 4D camera object (accessible in the Details tab), but rather Iray multi-passes (page 57).

4.3.2 Linear workflow and tone mapping

Basically the term *tone mapping* just means to map one set of colors to another color set. Most commonly, this is done to define how a large set of distinct colors (high dynamic range) can be converted to a smaller set of colors (low dynamic range).

To make you familiar with the general concept of tone mapping, we here provide an introduction and some background information on why it is needed in the first place, and how you can integrate it into your production workflow.

4.3.2.1 A conceptual introduction to linear workflow

You might expect that a pixel with RGB value 200 200 200 is twice as bright as a pixel with RGB value 100 100 100. This would be true, if your screen would output light intensity in a linear way, such that a certain increase in RGB brightness matches exactly the amount of additional brightness of the screen.

However, the color space of a normal off-the-shelf computer screen is not linear. This is not a "bug" because due to the fact that our eyes see light in a non linear way, the former color (RGB 200 200 200) is actually perceived to be about twice as bright as the latter. This makes the color space of a normal computer screen roughly perceptually uniform. This is a good thing, and is actually the main reason 24 bit color (with only 8 bits, 256 discrete levels, for each of the red, green and blue components) looks as good as it does to our eyes.

The problem is that physically correct computer graphics operate such that a color value represents actual light energy. If one simply maps the color output of the renderer naively to the 0-255 range of each RGB color component it is incorrect.

The solution is to introduce a mapping of some sort. One of these methods is called *gamma correction*. Most computer screens have a gamma of about 2.2, but a renderer like Iray internally calculates color intensities linearly, as just pointed out, which corresponds to a gamma of 1.0. Displaying a thus rendered image on a screen makes everything (especially mid-tones) look too dark, and light will not "add up" correctly. Applying a gamma of 2.2 to the final image makes the physically linear light inside the renderer appear in a correct linear manner on screen.

However, in every non-trivial CGI project you will employ a workflow in which the actual rendering process is just one part among many others. These may include the creation and preparation of textures, compositing and other pre- and post-production tasks. Hence a common demand is to apply image corrections that are specific to the characteristics of the display hardware, only once in the workflow. This requires every image resource, as well as colors itself, to be free of these corrections — which means for them to have no gamma applied, or, equivalently, a gamma value of 1.0. As stated in the previous paragraph, a gamma value of 1.0 weights differences in light intensity linearly. Using images and colors without applied gamma correction throughout the production pipeline has thus become known as *linear workflow (lwf)*. This assures a consistent handling of color intensities before applying a gamma correction only once to the compound image.

Another method to map the physical energies inside the renderer to visually pleasing pixel values is tone mapping. This can be done either by rendering to a high dynamic range file format and using external software, or by using the built-in methods. Both approaches are described in detail in section Tone mapping and gamma in Iray for Cinema 4D (page 55).

4.3.2.2 Practical guidelines on color space workflow

To adhere to a linear workflow, you have to ensure that the data going into and coming out of the renderer is unmodified.

On the input side, this affects texture images and colors you explicitly provide in your applications, for example by using color pickers. In most cases, low dynamic range images are already in sRGB color space with an applied gamma of 2.2. That is so they look correctly when viewed directly on the screen. But when those images are used as a texture, their color information will be interpreted as linear. Applying gamma to your final rendering would result in double gamma for the rendered pixels that show the image texture, and single gamma for the rest. To remedy the mixed-gamma situation you have these options:

- De-gamma input values: applying a gamma of 0.455 to input colors inverts a gamma value of 2.2. *Linearizing* a texture image like this, you remove the gamma compensation and supply the "pure" color to Iray. This must only be done to color data that have a gamma applied to them already. In digital photography, cameras record light linearly. However, upon writing the image to a low dynamic range format, JPEG for example, they apply gamma directly. Raw image formats contain the unmodified sensor data without gamma correction instead.
- Use textures in linear color space: while low dynamic range images are mostly in sRGB color space with applied gamma, high dynamic range images are generally not pre-processed with gamma. This is the very purpose of HDR images: to remain proportional to light and to pertain the light intensities captured by the camera linearly. A common file format to store floating point image data is OpenEXR. Using textures in linear space, you need not to worry about gamma at this point.

Having linearized input data, you receive a rendered image with actual, "pure" light intensities. Displayed as-is on the screen, it will not look correct though: for the output device, a gamma correction or tone mapping must be applied, as described in the lwf introduction (page 54). Therefore consider the following cases:

- If you want to view your fully linearly rendered image on the screen, you have to employ tone mapping. This is mostly useful for previewing render results, to be able to judge the percieved color and brightness of rendered images.
- If you want to create renderings for post-production, it is most often useful to keep the linear output. This way, you can post-process the true colors and have maximum control without any colors being clipped before post. You then have to apply tone mapping in the very last step of your post-production pipeline.

How these two different goals can be achieved and controlled with Iray for Cinema 4D is addressed in the following section.

4.3.2.3 Tone mapping and gamma in Iray for Cinema 4D

Having linearized color input as described above, you now have to decide whether or not to apply tone mapping or gamma to the render result Iray produces. This usually depends on the goal you want to achieve with the render result. We here provide instructions how both of the possible goals, discussed in the previous section, can be achieved using Iray for Cinema 4D.

If you want to view your rendered image on the screen, you have to employ tone mapping. While this could be done in the Cinema 4D picture viewer, we recommend using the tone mapping techniques Iray supplies. To enable tone mapping for an existing scene, follow these steps:

1. Create a standard Cinema 4D Camera object, either via the tool bar or the Create menu.
2. Assign the camera to the active view, either by double-clicking the icon of the camera object in the Object Manager or by selecting the camera in the viewport menu at [Cameras ▸ Use Camera].
3. Select the camera in the Object Manager.
4. From the Iray menu, select Iray Camera Tag to assign it to the selected camera object. Alternatively, all the Iray tags are also available in the context menu of the Object Manager.
5. Select the Iray camera tag in the Object Manager.
6. In the Attribute Manager, click the Tone mapping tab.
7. By default, tone mapping is disabled. Check the Enabled option, to enable tone mapping.
8. Expand the Advanced group. The default value for Gamma is 2.2, which is the correct value for most screens.
9. Rendering the scene, you should now see tone mapping applied to the final rendering, resulting in a way more natural appearance with respect to contrast and color intensity variance. Use the Exposure Value to adjust the exposure intensity to your specific scene.

As you can see, there are several parameters to control tone mapping and other camera properties. These are explained in detail at the tone mapping user interface descriptions (page 16).

Note: Since tone mapping is an image process, it does not affect the actual rendering (which of course takes place in 3D). Using the live render window, you can therefore observe each of the tone mapping parameters dynamically: adjusting the parameters, the live rendering will not start from anew; instead you can see the tone mapping effects in real-time, while the rendering keeps refining.

A second common goal is to render images for post-production. Often, one wants to have full control over colors in post-production, without having them clipped by the renderer or otherwise pre-modified. Follow these steps to render to an unmodified output file:

1. First of all, to keep the full range of color intensities, it is important to specify a high dynamic range output format. Common low dynamic range formats like PNG are not able to store more than 256 discrete levels for red, green and blue. Open the render settings and click Save on the left hand side.
2. Choose OpenEXR as output format. As you can notice, the parameter Image Color Profile is now fixed to Linear Color Space.
3. We assume you already have a camera object with an assigned Iray Camera Tag. If not, follow the instructions for tone mapping just above.
4. To write unmodified, linear color data, it is important not to apply any sort of tone mapping or gamma. Therefore, select the Iray Camera Tag and disable tone mapping completely by clearing the Enabled option.

Rendering to the specified output file, Iray now does not clip or modify any color values but instead writes the linear energy values. In your post-production software, you now have full control over these true values. As you typically want to create final low dynamic range images, able to be displayed using common image or video software, you should apply tone mapping and gamma only at the very last stage of your production workflow.

Note: The automatic shader baking (page 26) feature of Iray for Cinema 4D creates high dynamic range textures and as thus adheres to a linear workflow. You do not need to care about pre- or post-processing these textures.

4.4 Practical Multi-Pass rendering with Iray

A common task in 3D graphics production is to render specific output data into individual passes. Iray for Cinema 4D allows to conveniently manage render passes and contains flexible tools to control multi-pass output.

Rendering multiple passes can be enabled and controlled as follows:

1. In the menu, click [Iray ▸ Iray Render Settings].
2. Select NVIDIA Iray on the left hand side.
3. Click the Multi-Pass tab.
4. Here you can now select the set of passes you want to render separately. As long as there is no pass preset set, or at least one auxiliary pass enabled, Iray just renders the beauty pass to the regular image output. When at least one pass is given, Iray for Cinema 4D automatically activates the Cinema 4D Multi-Pass option and creates multiple pass outputs.

To inspect the different pass results in the Cinema 4D picture viewer, follow these steps:

1. On the right hand side of the Cinema 4D picture viewer, click the Layer tab.
2. This list shown here should contain your different pass outputs. To view a single pass, set the layer option to Single-Pass.
3. Now you can select any pass from the list and the picture viewer just shows the respective individual pass result.

A very powerful feature of Iray for Cinema 4D is its ability to render different scene objects into individual passes. This can be done by employing LPE labels, covered in section Rendering object passes (page 60).

4.4.1 Multi-Pass capabilities and auxiliary passes

Iray is not restricted to just render different scene information into separate passes. Instead, it allows to query exactly the path that light takes through a scene. Using a syntax based on regular expressions, you can build any number of passes, each including precisely the render information you need. Creating such passes on your own, you of course are most flexible; however, you have to understand the syntax and grammar of these light path expressions. For many applications, a limited set of regularly used passes might suffice, so Iray for Cinema 4D includes some ready-to-use presets.

4.4.1.1 Pass presets

The built-in pass presets can be accessed in the Multi-Pass tab of the Iray render settings. Using the buttons with the plus and minus sign, you can dynamically add or remove the number of available passes. Clicking the minus sign deletes the last entry in the pass list. Previous selections get remembered though: if you choose Emission for pass 10, for example, then remove it and add a new pass, it will default to previously selected Emission.

As you can see, each of the preset internally represents a light path expression. As long as the preset is selected, you cannot modify this expression. Select Custom to either modify an existing preset or to enter a completely new light path expression.

By clicking Edit, you can open the LPE Editor to help you construct expressions in a more convenient manner than by mere typing.

Note: Pass presets that depend from the viewer (or camera) position might not be meaningful in the irradiance mode. These are the Glossy and Specular preset.

4.4.1.2 Auxiliary passes

Additionally to the LPE pass presets, Iray for Cinema 4D includes four *auxiliary passes*. These often-used passes that can not be formulated as light path expressions are convenient to work with in post-production. The following auxiliary passes are available:

Alpha
: This pass includes only the level of transparency of the final image, also called its *alpha channel*. Its color information represents the per-pixel value of transparency. As this is a single gradual quantity, the result is a grayscale image, where completely black represents full transparency and completely white means full opacity. The Alpha pass is typically used to mask out specific parts of the rendering in post-production, or to weight the influence of image-based methods according to transparency.

Normal
: This pass renders the surface normals used for shading at each point. Thus created normal maps are useful to bake geometric information of high-poly objects into a texture. Applied to a low-poly version of the model, the lighting effects of the geometric details are simulated, typically yielding good enough results with a significant reduction in polygonal geometry.

Depth
: This pass indicates the distance from objects to the camera. A single gradual quantity, the depth is rendered as a grayscale image where darker parts are nearer to the camera and brighter parts are farther away from it. The Depth pass is useful to create depth of field effects in post-processing.

UV
: This pass displays the texture coordinates used for UV mapping.

4.4.2 Using light path expressions

Light Path Expressions (LPEs) describe the propagation of light through a scene, for example starting from a source of light, bouncing around between the objects of the scene and

ultimately ending up at the eye. The paths that light takes through a scene are called *light transport paths*.

LPEs are a powerful tool to extract only specific light contributions into separate passes to use in compositing and post-production. In essence, LPEs are regular expressions. The alphabet of LPEs consists of event descriptions, that is, of interactions between light particles and the scene.

In the following, we provide an introduction to light path expressions, its alphabet and grammar.

4.4.2.1 Camera and light sources

As an LPE describes the path of light, this path naturally has a start and an end. The path either starts from a light source and ends at the camera, or vice versa. These are examples of simple *events*. The camera, or eye, can be addressed in LPEs with `E`. For light sources, there are different identifiers, depending on the type of light source:

LPE identifier	*Light source type*
`L`	Any light
`Lp`	Point light
`La`	Area light
`Le`	Environment or background light

To render a beauty pass, every light interaction of any kind of light source should contribute to the rendered image. Just as in regular expressions, the dot character (`.`) designates "match anything" and an asterisk (`*`) means "zero or more" or, equivalently, "any". The LPE for the beauty pass is hence: `L.*E` To get just the lighting that originated from environment, the LPE `Le.*E` matches exactly that.

4.4.2.2 Interaction events

To query for the interaction between materials and light, you can filter by the interaction *type* and *mode*:

LPE identifier	*Event type*
`R`	Reflection
`T`	Transmission
`V`	Volume interaction

The mode describes the kind of scattering:

LPE identifier	*Event mode*
`D`	Diffuse
`G`	Glossy
`S`	Specular

An interaction event is enclosed in angle brackets, so `<RD>` designates a simple diffuse reflection.

You can combine multiple events by enclosing them in square brackets (`[]`). For example, `<[RT]>` matches all reflection events and all transmission events. Analogously, `<[RT][DS]>` includes reflection and transmission, with diffuse and specular scattering.

4.4.2.3 Operators

A number of common operators can be used to combine sub-expressions. The operators are identical as in normal regular expressions, if you are familiar with regular expression, you should be able to use them right away. The available operators are (`A` and `B` represent sub-expressions; `m` and `n` represent integer values):

Operator	*Description*
`AB`	Matches `A`, followed by `B`
`A|B`	Matches `A` or `B`
`A?`	Optionally matches `A`. `A` might not be present in the scene.
`A*`	Matches zero or more occurrences of `A`
`A+`	Matches one or more occurrences of `A`
`A{n}`	Matches `n` consecutive occurrences of `A`
`A{n,m}`	Matches from `n` to `m` occurrences of `A`
`A{n,}`	Matches `n` or more occurrences of `A`

Sub-expressions enclosed in parentheses (`()`) are evaluated first.

4.4.3 Rendering object passes

In post-processing, it is often crucial to render individual objects into separate passes. A common application is, for example, to mask certain objects to be excluded in compositing. Iray for Cinema 4D provides the Iray Object Tag to label objects that should be treated individually in rendering.

Follow these instructions to mark an object for multi-pass rendering:

1. Either start with an existing scene that has multiple objects or create a simple test scene with at least three polygon objects that are visually distinct when rendering. For demonstration purpose, three primitive objects with a bit of distance to each other will suffice.
2. Select an arbitrary polygon object in the Object Manager.
3. From the Iray menu, click Iray Object Tag to create a new tag, getting assigned automatically to the currently selected object. Alternatively, all the Iray tags are also available in the context menu of the Object Manager.
4. Select the newly created object tag. In the Attribute Manager, type `mask` in the LPE Handle attribute text field.
5. In the Render Settings, select NVIDIA Iray as renderer.
6. Click NVIDIA Iray in the list on the left hand side of the render settings dialog to access its properties.
7. Click the Multi-Pass tab.
8. Under Light Path Expressions, choose a pass (for example Pass 2) and set it to Custom.

9. Enter `E 'mask' .* L` in the text field of that pass. This expression will mask this object, so that only the light interacting with this object contributes to the render result of this pass. For detailed information on the wide range of the possibilities that light path expressions offer, and on the structure and syntax of it, consult the previous section. Rendering this pass now, you should now only see the objects that are marked by the label "mask" with an Iray Object Tag. The rest of the rendering should be black. With this approach you can easily create convenient passes for post-production compositing.

Having set the custom label of "mask", you can now assign or copy this tag to any number of other objects. All the objects with this tag are then included in the rendering, since the relation between the pass and each object is maintained through the custom label.

Note: If not given a custom label with the LPE Handle attribute, the Iray Object Tag automatically uses the name of the object it is assigned to. An object tag on a polygon object "Chair", for example, initializes with the LPE handle "Chair", too. So long as you do not define the LPE Handle manually, the tag keeps up the relation to the assigned object dynamically: when you rename the assigned object or move the tag to another object, the Iray Object Tag updates the LPE handle to the new name accordingly.

When no object tag is assigned, objects can still be addressed in LPEs with their name.

Note: Object hierarchies are supported: when assigned to a parent object of a hierarchical group of objects, all of its child objects are being evaluated in light path expressions with the tag's label as well.

4.5 Improving image quality

In this section you will learn how to use filters to improve the image quality of your final rendering. It covers both improvements by reducing artifacts as well as by adding visual effects. In an introduction (page 61) we briefly point out the principles of filters in Iray. Following sections contain information on how to minimize the most typical artifacts (page 62) in physically based rendering, and how you can enhance visual intensity of bright parts (page 64) of the rendering. A final section mentions other possibilities to control image quality (page 70) that are covered elsewhere in this document.

4.5.1 What are Iray filters?

In the scope of this document, the term *filter* denotes an image processing technique. As such, a filter operates in two-dimensional image space and not, like rendering itself, in 3D. Hence a filter can be applied to an image, for example the final rendering or a texture. In general, the filter alters the image in a specific way, using only image information (which is given by the color of its pixels, of course). A simple yet common example is modifying the brightness of an image: each pixel of the image is brightened or darkened, regardless of the actual illumination at that point in 3D. Some consequences that result from the very nature of a filter, should thus be kept in mind:

- Since filters only process image data, in most cases they cannot conform to physical correctness. If you think of the illumination being computed by Iray in a physical correct way, given by the scene's geometry, lighting and materials, it is clear that a change in

the final image contrast is independent of physical rules. This should be considered when applying filters; we recommend using them for rather subtle changes, to keep the overall authenticity of a physically correct rendering.

- As multiple filters may act upon an image, the order of application is important. This is something you normally should not need to worry about: Iray applies its own filters in the correct order. For example Iray will first reduce unwanted bright pixels with the firefly filter and then use the bloom filter to enhance bright parts of the image. Applying them vice versa would result in the unwanted effect of intensifying image artifacts. However, if you intend to use other post production filters outside of Iray, the order of filters can be of importance.

4.5.2 Reducing image artifacts

In physically based rendering, a common rendering artifact is the appearance of so-called *fireflies*. They can arise under some difficult lighting conditions and are visible in the final image as very bright pixels.

Fireflies are to be distinguished from an overall noise, evenly distributed over the whole image. Regular noise most often results from to little rendering iterations (and can thus be reduced by allowing more render time). Instead, fireflies are single overly-bright pixels that are more frequent in some specific parts of the image.

Iray offers two filters to reduce render artifacts resulting from fireflies or noise: the Firefly filter and the Degrain filter, both of which are described below.

4.5.2.1 Firefly filter

Iray includes a built-in firefly filter to reduce these artifacts: fireflies are automatically detected and masked out in the early rendering stage when variance is high but eventually these missing contributions are faded back in over time. Thus, the results at a low sample count may appear slightly darker than the final, converged image. The filter works best in combination with the built-in tone mapping (page 16). If tone mapping is not enabled, the filter estimates can be too pessimistic for some scenes, resulting in some fireflies to still appear.

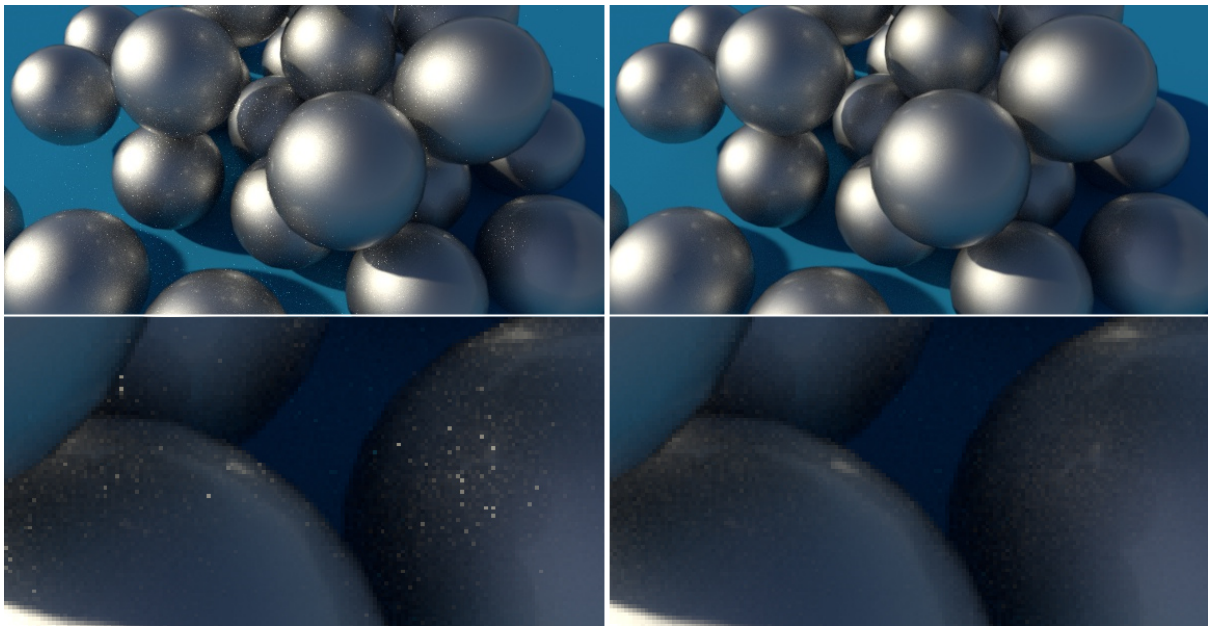


Fig. 4.14 – Using the Iray firefly filter, you can reduce rendering artifacts resulting in unwanted bright pixels. The left side shows a rendering with disabled firefly filter. When the filter is turned on, the artifacts are reduced significantly as can be seen on the right.

The firefly filter is enabled by default; it can be turned on or off in the Iray renderer settings (page 28).

4.5.2.2 Degrain filter

The degrain filter can be used to reduce low frequency noise without sacrificing overall sharpness, mainly to reduce remaining subtle grain in difficult areas of a scene. How a degrained rendering compares to an unmodified rendering, is shown in Figure 4.15.

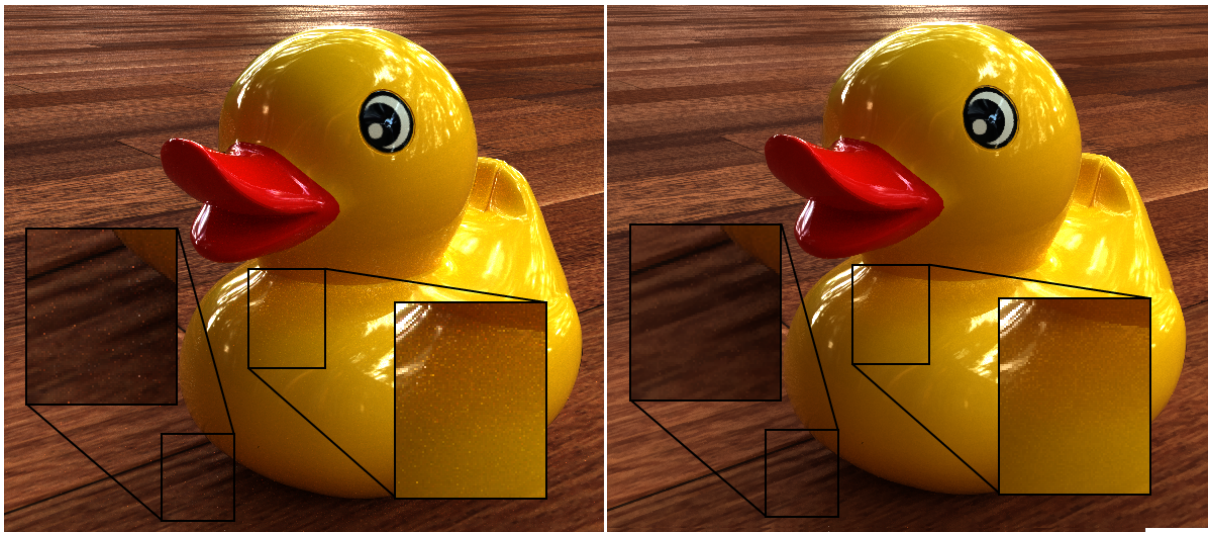


Fig. 4.15 – The degrain filter can remove noise from the final image. In the left image, no filtering was used. In the right image, the Iray degrain filter was applied, using the default settings (Mode 3, Radius 3). Note that for illustration purposes, the renderings are shown "as-is", without any kind of tone mapping.

Note: To increase interactivity, the Iray live render window does not evaluate the degrain filter as stated in section Differences between live rendering and production rendering (page 8).

Degrain filtering is disabled by default; it can be turned on or off in the Iray renderer settings (page 28). Here you can also fine-tune the filter process with the following parameters:

Mode

Using this parameter, you can select one of five different filtering modes. Modes 1 to 3 are working very conservative and should thus be safe to use in general, modes 4 and 5 are considered to be more agressive and should be used with caution, especially if the scene features fine details in either geometry or applied materials.

Unfortunately there is no general recommendation on which degrain filter mode will perform best on a specific scene, thus it is necessary to cycle through the different modes manually to receive the best results for a given scene. As a rule of thumb, modes 3 and 5 are usually the most reliable ones, with 5 being the more agressive of the two. Also note that the overall rendering performance can be reduced noticably on low-end CPU cores, thus the filter should only be enabled in the final phase of the rendering process.

The default value is Mode 3.

Radius

The "area of influence" of the noise reduction depends on this parameter. The value should thus be reduced if the filter smoothens out edges, and increased if some noise still remains in the image.

The default value is 3, but we recommended to experiment with different radius settings per scene to achieve best results.

Blur

Modes 4 and 5 feature an additional setting that limits the influence of neighboring pixels if the brightness is too different. Higher values create a more blurry effect.

The default value is 0.05.

4.5.3 Adding a glow or bloom effect

Glare is the phenomenon you experience when looking at a very bright light source. It is caused when the incoming luminance overwhelms the light receptors of an eye or a camera. As such, humans are accustomed to this effect, and it can increase a rendering to be perceived as realistic. In computer graphics, this effect is usually called *bloom* or *glow*, and it is typically realised as a filter that bleeds bright parts of the image into the darker surroundings.

As Iray is a physically correct engine, it renders bright light sources without glare, since there is no limit on light receptors of course. Instead you can use the included bloom filter, which allows to simulate a glare effect in the final rendering. Follow these basic steps to add a glow or bloom effect to your rendering:

1. Prepare a scene with a considerable amount of concentrated luminance emitted towards the camera (for example a sphere with a light-emitting material).
2. Open the render settings and choose NVIDIA Iray.

3. In the Options tab, enable Bloom.
4. Rendering the scene to the picture viewer, you should now see bright parts of the image to be extended outwards and blurred, similar to the glare effect in reality. Figure 4.16 shows how adding glare affects the visual perception of bright parts of an image.

Note: To increase interactivity, the Iray live render window does not generate bloom effects as stated in section Differences between live rendering and production rendering (page 8). Be sure to render to the picture viewer to see bloom effects in the rendering.

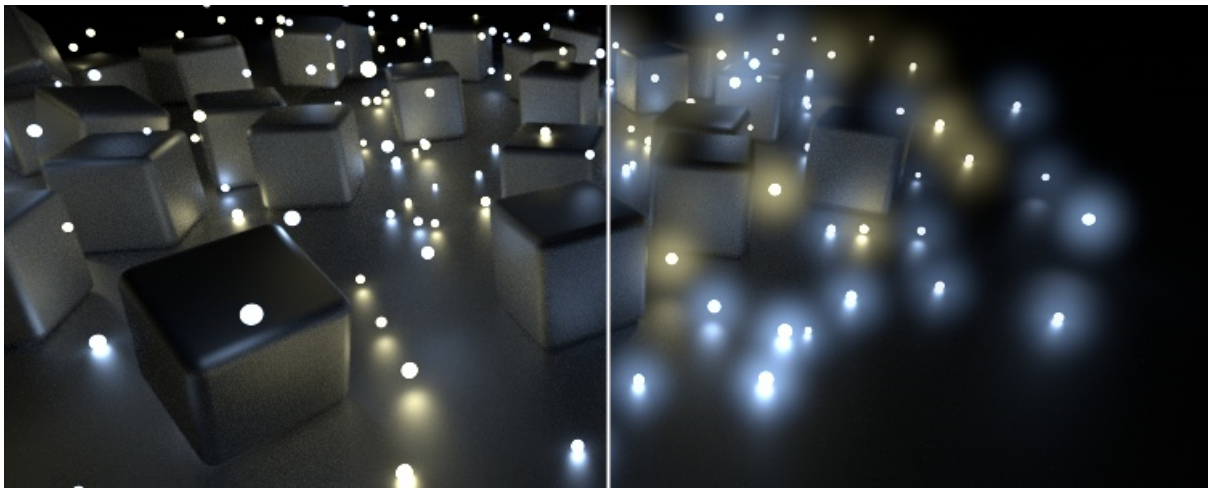


Fig. 4.16 – Adding a bloom effect intensifies the perception of overly-bright light sources. The left side is rendered without bloom; the right sight is rendered using the Bloom filter of Iray for Cinema 4D.

You can see that the filter increases the perceived brightness of highly luminous spots. The different colors of the light sources are now clearly visible; without bloom, they are very subtle due to the overall high intensity of the light sources.

Iray provides three parameters to control the visual appearance of the bloom filter:

Intensity

Using this parameter, you can adjust the intensity of the bloom effect. Note that the final visual intensity also depends on the bloom size (which is specified with the Radius parameter, see below). Think of this parameter as the total intensity available for a single bloom: a larger bloom results in less density, as the total intensity gets distributed over a larger area. Hence if you set a low value for Intensity and a high value for Radius the bloom effect proportionally vanishes.

The default value is 1.

Fig. 4.17 – A highly glossy object, rendered without bloom effect. While some of the reflected light may be recognized as bright, the overall reflected intensity is still rather weak as the visibile reflections have sharp borders.

Fig. 4.18 – The same object, rendered with the default settings of the bloom filter. The bloom intensity is 2, here resulting in a subtle glow effect, best seen on the left and very top of the object. Very bright parts of the image are now blurred into darker surroundings, giving an overall brighter impression for reflections.

Fig. 4.19 – A bloom intensity value of 5 here results in an exaggerated glow effect. The reflections are intensified too much, which a viewer perceives as artificial.

Radius

The size of the bloom effect depends on the Radius. With a small radius you get very intense light sources and a subtle glare effect. Note that the parameter value is specified as fraction relative to the output resolution. This way, the bloom effect scales with the size of the final rendering. Very high values can be used to create a fog-like effect as can be seen in the following comparison images with different radius values:

The default value is 0.01.

Fig. 4.20 – A bloom radius of 0.01, which means 1 % of the output resolution, is the default value and typically yields a subtle glow size.

Fig. 4.21 – Higher values spread the glow over a larger area while keeping the total bloom intensity of the area constant; here, the specified radius value is 0.05.

Fig. 4.22 – The radius value of 0.1 distributes the glow over a large area; when the total intensity is not raised simultaneously, this results in an attenuated bloom, but with a greater space of influence.

Threshold

This specifies the threshold to cutoff bright spots in the image and blur these in a second step. Higher values make bright spots more distinct from the surrounding halo, whereas smaller values reduce sharp boundaries of bright spots. You can notice the difference comparing the following images.

The default value is 0.9.

Fig. 4.23 – The default threshold value of 0.09 here blurs most of the bright reflections.

Fig. 4.24 – A higher threshold of 1.5 now only blur the very brightest reflections on the left and top side of the objects. The intensity of the other reflections is below this threshold, so they will not be included for the glow creation.

Fig. 4.25 – A lower threshold value of 0.5 makes Iray add glow to even somewhat darker parts, so the overall image gets blurred more.

As described in the introduction to filters (page 61), we recommend using the bloom subtly, like adding a slight halo to emissive objects. Very distinct, strong bloom effects usually rather result in reducing the authenticity of a realistic rendering.

4.5.4 Other methods of improving image quality

Besides adding bloom or enabling the firefly or degrain filter, there are numerous other methods of different nature that can improve image quality. We here briefly list and link to the approaches that are covered elsewhere in this manual.

4.5.4.1 Image-based methods

These methods that work in image space can be used to improve results:

Anti-aliasing
: To smoothen the jagged, stairstep-like appearance of diagonal lines in the rendered image, Iray provides an image filter (page 32). You can control the anti-aliasing in the Iray render settings (page 32).

Tone mapping and gamma
: Iray calculates colors without any restrictions of the output format or the output device. However, when displayed on the screen, one must apply some sort of tone mapping or gamma so that the image is perceived in a natural way. Section Linear workflow and tone mapping (page 53) contains detailed information on this and how it can be accomplished with Iray for Cinema 4D.

4.5.4.2 Non-image-based methods

There are also other techniques, not limited to image-space, that can improve render results:

Caustics
: When your rendering features distinctive caustics, the built-in caustic sampler (page 47) can help to improve these effects.

Architectural scenes
: for indoor scenes, mostly illuminated by indirect lighting, consider using the architectural sampler (page 30).

Motion blur
: To authentically visualize motion, you should simulate the motion trails that also appear in photography. Section Creating motion blur effects (page 73) covers how to achieve this with Iray. This is of special importance for still image renderings.

Depth of field
: Real cameras exhibit a limited range in which objects appear in focus. To defocus objects beyond that range is a feature of a renderer, strongly increasing realism. Iray for Cinema 4D includes the Iray Camera Tag to control the depth of field effect (page 50).

Image-based lighting
: Using real-world images for environment lighting can result in very realistic final renderings. Section Image-based lighting (page 42) describes how you can achieve this with Iray for Cinema 4D.

4.6 Rendering Cinema 4D Hair with Iray

Cinema 4D includes powerful tools to conveniently create procedural hair. A description of the wide variety of hair tools, Cinema 4D provides, is well beyond the scope of this document. In this section we instead show how Cinema 4D hair can be added to an object and be rendered with Iray.

When adding a hair object to your scene, Cinema 4D automatically also adds and enables a Hair Render effect to the render settings. This effect handles rendering the procedural hair with the standard Cinema 4D renderers. To render hair with Iray instead, you need to create polygon hair. Follow these steps below, to set up a simple scene for rendering Cinema 4D hair with Iray:

1. First of all either import an object that should be augmented with hair, or create a new one. In both cases, keep an eye on the total size of the object: the hair simulation may cause problems when the object is overly large or very miniscule. In this guide we assume the object to be of a size comparable to a standard polygon sphere when created anew.
2. Select the geometry that should grow hair. This might be the whole object or a selection of polygons of the object.
3. From the menu, select [Simulate ▸ Hair Objects ▸ Add Hair]. This adds a hair object with an already assigned Cinema 4D hair material. The previously selected geometry is automatically linked to this hair object; the linked geometry defines where hair is generated.
4. Select the hair object in the object manager. In the attribute manager, click the Guides tab. Adjust the hair length with the Length parameter. In this example, a length of approximately 5 cm should fit. Also, you can control where hair is generated, using the Link parameter.
5. To render hair with Iray, the hair must be made of true polygons. With the hair object still selected, go to the Generate tab in the attribute manager. Using the Type parameter, you can convert the hair to polygons. Flat would create flat polygonal stripes, so some hair would not be visible when oriented unfavorably towards the camera. Hence, select Triangle here; this creates thin cylinders with a triangular cross section. The total polygon count of the created hair depends on the number of hairs but also on the segments of these cylinders. Both parameters can be controlled in the Hairs tab.
6. Add a light source, for example an Iray Environment Object from the Iray menu.
7. The hair can now be rendered with Iray. In the render settings, select NVIDIA Iray as renderer. Rendering to the picture viewer, you should now see generated hair in the rendered image.

Note: For viewport performance, it may be useful to define how the hair is displayed in the editor. The Editor tab of the hair object contains numerous settings to control this. Practical settings are:

- Setting Generate to As Render displays the actual polygon hair in the editor. This is useful for an accurate preview, but slows down the viewport considerably when using a high number of hairs.
- Setting Generate to None enables the Display setting. Here you can define an alternative hair type for the editor. Guide Lines, for example, is a good compromise between preview accuracy and viewport performance.

The hair properties can be modified with the hair material. But one instantly notices that the rendered hair is plain white; the color given in hair material color is not respected when rendering with Iray. You can overcome this easily by using a separate material:

1. Create a new material; this can be either an MDL material or a standard Cinema 4D material.
2. Configure the new material properties, like adjusting its color or reflection.
3. Assign the new material to the hair object, together with the original hair material.
4. Using the original hair material, you can control the hair attributes like Frizz, Thickness and Density, for example. Just omit the color and specular channels of the material, as they are now defined by the other material.
5. Rendering to the picture viewer with Iray, you should now see the hair rendered with the hair material properties as well as the properties of your new material, similar to Figure 4.26.

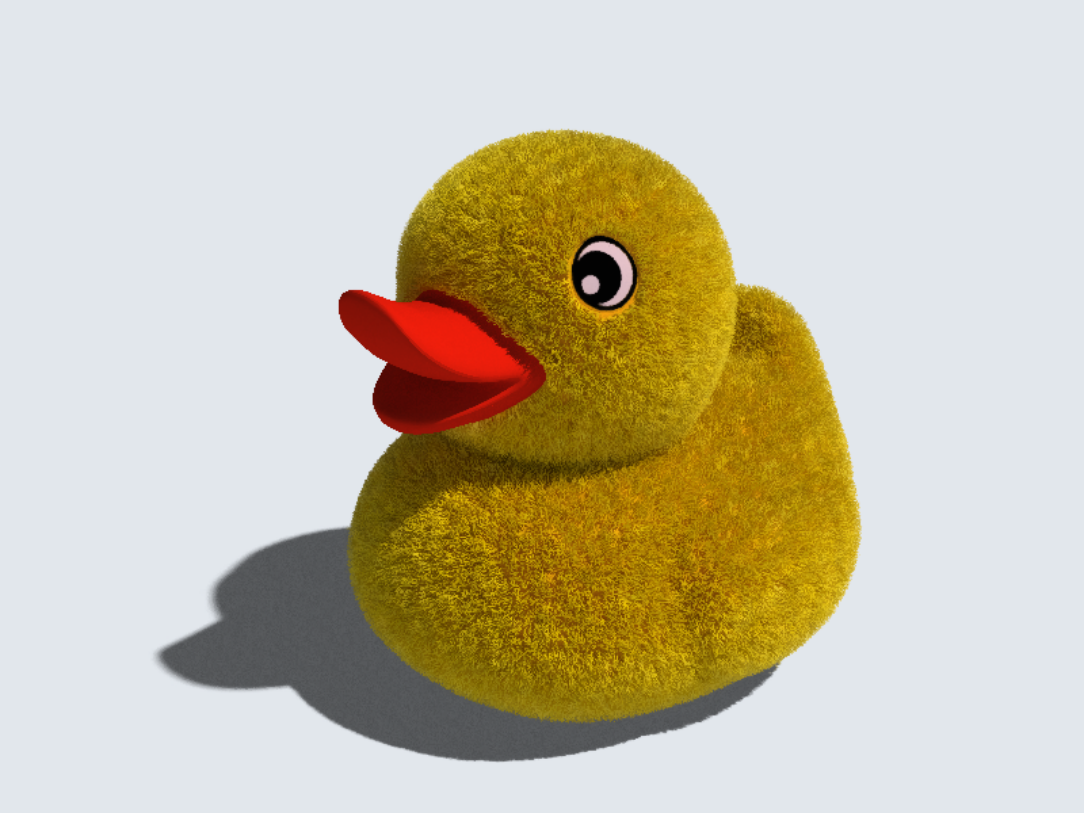


Fig. 4.26 – Cinema 4D hair rendered with Iray

4.7 Creating motion blur effects

In this section you will learn how to create motion blur effects with Iray for Cinema 4D. We point out best practices regarding performance and quality. A brief introduction (page 73) will make you familiar with the general concept of motion blur and how it can be simulated in rendering software like Iray. We also provide some information about how Iray can be used to create renderings to use for post-production motion blur (page 76). The Troubleshooting (page 77) section contains hints to look at if the motion blur effects are not showing up or turn out to be unexpected in their visual appearance.

4.7.1 What is motion blur?

Motion blur is an effect in photography or film that occurs when objects are moving during the exposure of the film or sensor. The movement of an object becomes apparent in the form of visual trails along the motion during the exposure. A faster exposure hence yields a less blurry image. Conversely, a long exposure results in more motion blur. In the real world, this is controlled by the shutter speed. The perceived blur thus depends on the shutter time of the camera and the amount of motion that happens within this timeframe.

In computer graphics, there of course is no inherent concept of shutter speed. So without explicitly simulating this effect, a digital rendering is free of any motion blur. Contributing greatly to the perception of motion, there is however a great demand to render this effect.

Iray is capable to simulate a shutter effect of real cameras, and is thus able to create the blurred appearance of moving objects and light sources. Also movement of the camera itself causes a blur of the rendered scene.

4.7.2 How to create and control motion blur effects with Iray

With Iray, adding motion blur effects to your rendering is simple and straightforward. Follow these basic steps to render motion blur:

1. First of all make sure that the frame you want to render contains an object moving fast enough. The position or orientation of the object should change considerably between two frames, to make motion blur appear distinctively.
2. Open the render settings and choose NVIDIA Iray.
3. In the Renderer tab, enable Motion Blur.
4. For now you can keep the default values of the other parameters.
5. Render to the picture viewer. After the render finishes, you should now see the motion blur effect in the resulting image, similar to the picture below.

Fig. 4.27 – Motion blur rendered with Iray: visual motion trails allow for a credible perception of speed, even in still images.

In order to control the visual appearance of the motion blur, you can use the Shutter Open, Shutter Close and Samples per Step parameters in the render settings:

Shutter Open and Shutter Close

With these values you can adjust the amount of distance over which the object is blurred, along its motion between two frames. You can figuratively think of the units of these to values as 0 being the previous frame and 1 being the current frame. In this regard, a shutter open of 0 and a shutter close of 1 creates inbetween-renderings evenly throughout the complete motion between two frames. This also yields the most natural and authentic result, hence in most scenarios you should not need to modify these two values. Setting shutter open to 0.33 and shutter close to 0.67 for example results in blurring an object beginning with its position at one third of a frame and stopping the blur at two thirds of a frame. Please note that shutter close must be strictly larger than shutter open, otherwise no motion blur will be produced.

For experienced users, here is a technically more accurate description on these paramaters: Iray uses motion vectors to compute motion blur. The shutter open and shutter close parameters specify how the motion vectors should be clamped on each side along their direction. The default range of (0, 1) is therefore equivalent with using the full length of the motion vectors.

Samples per Step

Using this parameter, you can adjust how many render iterations should be performed before a new sub-frame will be calculated. Adjusting this value, you are therefore able to control the trade-off between the quality level of the intermediate steps versus the number of intermediate steps evaluated per time. This will become more clear with a simple example: between frame #1 and frame #2, an object moves a considerable distance from position A to position B. If you render frame #2 with enabled motion blur settings, Iray needs to calculate intermediate positions of the object along the path between A and B. Each intermediate position is rendered separately; all of them combined give the final image of the frame. There are hence two crucial quantities that influence the final rendering:

- The number of intermediate position renderings (sub-frames). More sub-frame renderings will reduce "motion steps", and thus result in a smoother transistion from A to B.
- The quality of intermediate renderings. This is determined by the number of iterations for each intermediate rendering. More iterations per sub-frame will reduce noise and result in a more refined image.

As you can see, these two quantities influence each other inversely, if we want a constant render time for the whole frame. If your end criteria is set to 5 minutes, there must be a trade-off between the number of steps along the object's motion and the quality with which each step is rendered: you could for example define a quality of just 4 iterations per sub-frame but let Iray calculate many steps between A and B. Iray then evaluates as much steps as possible, until the given time of 5 minutes runs out, while only using 4 iterations for each step. This would favor the first quantity, the number of steps, over the quality of each step. Another approach would be to define a quality of 64 iterations per sub-frame. As now way more time is spent to render each step, there will fit less sub-frames into 5 minutes. This would favor the second quantity, the quality of each step, over the total number of steps. The parameter Samples per Step lets you define exactly that number: the quality of each step (given in the number of iterations each sub-frame is rendered with), so you can control the described trade-off.

In the next section we provide some tips on how to optimize this value with respect to render performance.

The default value of this parameter is 8.

4.7.3 Performance considerations

It can be useful to adjust the Samples per Step parameter depending on your scene's simulation complexity. A higher value of Samples per Step optimizes render performance at the cost of a higher total render time (as specified in the end criteria) required before the separate motion steps become invisible in the image. With a lower value, the motion will appear in the image more quickly, but it reduces total render performance for multiple frames.

In general we recommend increasing this parameter if the dominant source of simulation complexity of the scene is the illumination. If, on the other hand, motion is more difficult, smaller values will yield better performance.

4.7.4 Post-production motion blur

Using the actual 3D motion of objects to create motion blur, like Iray does, often results in the most realistic result. However in some cases it might be of interest to apply motion blur in post-production, as an image based technique. If so, it can be useful to create renderings inbetween two frames, with a constant time offset. These renderings can easily be created with the motion blur settings of Iray for Cinema 4D :

1. Open the Render Settings and choose NVIDIA Iray (you optionally might want to create a new render setting for post-production motion blur)
2. In the Renderer tab, enable Motion Blur
3. Set both Shutter Open and Shutter Close to the same value which should be between 0 and 1 (most commonly 0.5 is useful). This value represents the constant offset between two frames, so a value of 0.5 computes scene data at an offset of one half of a frame.
4. You can ignore the Samples per Step parameter as this value will not be taken into account when Shutter Open and Shutter Close are equal.
5. Rendering the scene, Iray now evaluates the position of objects (and every other property) between frames at the given offset.

Note that it is important to set both Shutter Open and Shutter Close to the same value. This effectively disables motion blur but creates renderings at a discrete time given by the current frame minus the specified offset.

4.7.5 Troubleshooting

If your rendering unexpectedly lacks motion blur effects even though motion blur is enabled in the currently active render settings, here are a couple of things to check:

- At least one visible object must move fast enough inbetween two frames.
- If Shutter Open and Shutter Close are equal, motion blur is disabled but the scene is rendered with a discrete constant offset in time. See the previous section (page 76) for details about this.
- Samples per Step must be greater than 0.
- Shutter Close must be larger than Shutter Open, otherwise no motion blur will be calculated.
- Make sure you allow enough render time for your motion blur settings. The number of intermediate motion steps depends on both the Samples per Step parameter and the total render time a frame is given (determined by the end criteria settings). The end criteria must at least allow a greater number of iterations than the number of Samples per Step to create any motion blur. For reasonable motion blur rendering, we recommend to make sure the end criteria will not be met until at least four times the iterations of Samples per Step. For an in-depth treatise on the Samples per Step parameter, see section How to create and control motion blur effects with Iray (page 74).

Also please note that motion blur effects will not be produced in the Iray live render window as stated in the section Differences between live rendering and production rendering (page 8).

4.8 Iray Streaming

With Iray Streaming, you can employ a remote server to render your scene and stream back the results to you dynamically. The available remote resources for streaming are:

- An NVIDIA Quadro Visual Computing Appliance (VCA), a network-attached appliance that harnesses the power of the highest-performing NVIDIA GPUs. More information about VCA can be found at: http://www.nvidia.com/object/visual-computing-appliance.html
- Cloud rendering with the NVIDIA Iray Server, a software solution that provides distributed Iray rendering across remote machines, without the need to install any other application. Get information on Iray Server at: http://www.nvidia.com/object/iray-server.html

4.8.1 Setting up the connection to the server

Before actually using a remote render resource, you first must establish a connection:

1. From the Iray menu, select Iray Streaming.
2. If you did not connect to a server before, click the Add new Server button.
3. In the credentials dialog, enter the URL of the server you want to connect to, as well as the username and password of your account.
4. If you want to connect to a VCA, set the VCA option.
5. Optionally, set the Auto Login option, if you want to automatically connect to this server with this account when you start Cinema 4D.
6. Click OK to confirm the connection information.

You can add multiple server connections, either to an Iray Server or to a VCA. With the dropdown field in the Iray Streaming dialog, you can choose the server to connect to.

When the connection can be established, the dialog changes to display information about status, nodes and load. Turning on Node details you can inspect, which nodes are reserved by other users and some additional information.

If you are connected to a VCA, you may now reserve available render nodes:

1. In the Iray streaming dialog, use the Reserve Nodes dropdown box to specify the amount of nodes you want to reserve. Reserved nodes are only available to your account, even if currently no render task is performed or pending.
2. Optionally, set the Auto Release option, if you want that your reserved nodes should become available to others again, when you close Cinema 4D.
3. Click Reserve.
4. Wait for the system to process reserving the nodes. This can be cancelled by pressing Esc.

The streaming dialog should then update the node information, and the Iray Streaming icon in the Iray menu shows the number of your currently reserved nodes.

Note: The connection setting dialog can also be opened with the Connection settings button when choosing remote server resources in either the Iray render settings or the Iray preferences.

4.8.2 Rendering with the remote server

Having reserved at least one node, as described in the previous section, you are now ready to employ the server as a render resource:

1. Click [Edit ▸ Preferences] to open the Cinema 4D preferences.
2. On the left hand side, select NVIDIA Iray to edit the Iray preferences.
3. Click the Resources tab, if it is not already selected.
4. Select Remote Server from the Render Resources dropdown.

Rendering the scene should now be processed on the server you are connected to, so generally you will now experience a significant boost in rendering performance.

Note: The resource settings apply globally, as they are defined application-wide in the Iray preferences.

4.9 Iray Queuing

Iray Server can be used to submit multiple render jobs to the server. The system then acts as a distributed render farm, supporting user accounts and central queue management.

Iray for Cinema 4D includes a user interface to connect to the render server and submit render jobs directly:

1. From the Iray menu, select Iray Queuing.
2. The Server list displays Iray Server connections you previously connected to. If you want to add a new server connection, select Add new server and enter your account credentials.
3. Provide a Job Name under which the batch render job will be known in the render queue.
4. Specify the file format for the rendered output images.
5. Click Add to Queue to submit your render job to the Iray Server scheduler.

You can inspect and control the queue in a central management console, accessible with any web browser. Click Open Browser to directly navigate to the Iray Server management console directly from within Iray for Cinema 4D.