CINEMA 4D

CURRICULUM

State 12/2019

MAXON
A NEMETSCHEK COMPANY

Contents

Maxon Cinema 4D R21 Curriculum

1 About the Curriculum

This document was created to give instructors a guide for establishing a Cinema 4D training course. This curriculum's structure is designed to reflect that of lectures at educational institutions. The necessary theoretical backgrounds as well practical examples are used to teach Cinema 4D. This is a curriculum for basic instruction and students are not required to have any previous Cinema 4D skills. However, they must have solid computer skills. This curriculum covers all basic lighting, modeling, texturing, rendering and animation skills in Cinema 4D.

Because Cinema 4D can be run on either Windows or Mac OS, when necessary, references will be made to the different keyboard keys that can be used for a given function. Aside from this, Cinema 4D is identical both in its appearance and in the results achieved, regardless of which operating system is used.

A summary of important features/elements is located at the end of each section that can be used as a handout for students.

The practical examples included in the curriculum are designed to help instructors and students strengthen their knowledge what was taught. Depending on the time available, variations of these example or additional examples can, of course, be introduced.

The examples used in this curriculum are available in the zipped directory and can be opened after the file has been unzipped.

2 Introduction to 3D

Not all students will know what 3D actually is or for what a 3D software is even used. This is especially true with regard to design or general media studies of which 3D is only a small part or merely an optional class.

The first block of instruction should therefore be used to give students an overview of how and where 3D graphics and animations can be used. It's also a good idea to explain the occupational fields in which 3D plays a role. The goal is to spark the students' interest in this multi-faceted field and to generate enough motivation to first study the theoretical end of 3D before starting with the practical application.

It has proven to be very helpful and effective to use short films, making-ofs or still images of current Hollywood films or advertising campaigns during the initial phases of instruction. The Maxon demo reel is also perfect for giving students a quick overview of how 3D can be used and the different styles that can be created.

Generally speaking, the various common fields of use for 3D should be explained. The following categories can be emphasized:

2.1 Technical Visualization

This category includes product visualization for various industries such as auto manufacturers, consumer products and much more. Architectural visualizations also require a high degree of technical renderings, many of which must also be rendered photo-realistically. Product prototypes are also modeled and rendered during the product development cycle.

2.2 Medical Visualizations or „Explanatory" Films

Many microscopic procedures or extremely large processes cannot be demonstrated using traditional methods and need to be visualized using 3D. These types of visualizations are common in the fields of science and medicine.

2.3 Advertising and Motion Graphics

3D is engrained in advertising and for creating effects and abstract television spots. 3D offers artists complete control of objects, lights and cameras, which means that just about anything imaginable can be created.

2.4 Special Effects

Special effects can be used to compensate for a limited budget or to create something that simply does not exist or is not possible in the real world. This includes explosions and dynamics simulations as well as virtual sets or backgrounds to supplement existing environments. This category also includes matte painting and the enhancement of virtual characters. Motion-capture technology or even performance capture, for which real-world actors are used to control virtual characters, are also part of the exciting world of 3D special effects.

2.5 Computer Games

Computer games have always depended on virtual depictions of content. Initially, the content consisted of 2D graphics due to hardware restrictions. Today, almost every computer game on the market has 3D content, with worlds through which gamers can freely move. These games can even be played on modern mobile devices. Computer games will become even more prevalent in the future when devices such as Oculus Rift or Kinect will transfer the gamer's movement directly to the virtual game character for an even more intimate gaming experience. Interactively adding 3D objects is also interesting for those who are not afraid of programming. Game engines such as Unity 3D offer direct exchange with Cinema 4D so Projects can be used directly for games or other interactive applications. This can also include a virtual tour of a building, for example.

3 Cinema 4D's Scope of Features

Cinema 4D is a complete 3D software package, which means that it includes almost all features and functions required for the creation, rendering and animation of 3D objects. Of course applications are available that specialize in individual disciplines such as fluids simulation or cloth and whose corresponding features therefore outperform the standard features in Cinema 4D. More important for a good 3D application is not which effects it can produce itself but its ability to exchange data with third-party applications for a successful and productive integration into a production pipeline. And this is exactly where Cinema 4D's strength lies. It not only offers numerous import and export formats but well-documented exchange possibilities that can be used to easily and seamlessly integrate additional functions – referred to as plugins. Plugins can be used to add numerous functions such as external renderers.

Cinema 4D offers outstanding exchange with Adobe products such as Photoshop for still image editing and After Effects for editing animations.

3.1 Cinema 4D Strengths and Weaknesses

Cinema 4D's strengths lie, among others, in its easy learnability – even though its list of features has grown immensely. The interface in particular is easy to use for beginners. Important functions can be accessed via icons, menus or hotkeys. Advanced users can re-design the interface as they see fit to adapt it to their specific workflows, e.g., when using a multi-monitor setup. Cinema 4D's easy expandability via plugins has already been mentioned, which lets less powerful Cinema 4D functions be enhanced by third-party applications.

Product designers will miss real **NURBS** (**N**on-**U**niform **R**ational **B-S**plines), which make it possible to create production-ready objects. Modeling in Cinema 4D is polygon-based, i.e., objects primarily made up of triangles or quads (four-sided plane surfaces). This missing NURBS functionality can be compensated for by third-party applications such as Rhino 3D but is not really much of a barrier in practical application. High-end objects can be created in Cinema 4D but these models are not suited for use in the product manufacturing process.

4 A First Look at Cinema 4D

After all this general information, it's time to take a first look at the interface and the most important layout elements in Cinema 4D. Depending on the type of installation made in your classroom, your students should now start Cinema 4D. If a license server is being used, it must be started before the clients start their versions.

4.1 A Note About Automatic Updates

Often, a dialog window will appear when Cinema 4D is started that displays updates that are available for download. Cinema 4D is configured to contact the Maxon update server when started. If a newer version of Cinema 4D is available it can be downloaded and installed directly via the dialog window. Since this process can take several minutes depending on the type and size of updates, as well as the available bandwidth, these updates should be installed by the system administrator before or after instruction. If the students do not have admin rights, the updates cannot be installed anyway. Students can close this dialog window when it appears and the update(s) can be installed later. It's also important to know that any additional render clients used must have the identical version number as the Cienema 4D installation in order to render across a network. If Team Render or the Cinema 4D render client should also be used, please make sure you check that the versions match.

4.2 Cinema 4D Interface

By default, Cinema 4D starts with the Standard layout and presents a quick start window in which tutorials, industry news or previously opened projects can be opened. This dialog window will appear automatically each time the application is started but can also be called up at any time via the Window menu in Cinema 4D. You can also disable the automatic opening of the window via the **Preferences / Communication** menu. The default layout contains many of the menus and icons needed to work with Cinema 4D. The layout can be customized and optimized for special interest features such as Sculpt (the deformation of objects with brushes, stencils and stamps for the creation of organic shapes) or painting on surfaces in the BodyPaint 3D layout. At the top right of the Cinema 4D layout you will find a menu that contains several interface layout options. You can also use this menu to save your own custom layouts. Ifa student should accidentally change the Standard layout, for example by closing one of the manager windows they can revert to the Standard layout by selecting the **Start** layout from this menu.

Advise the students to use the **Start** layout initially. This will avoid any confusion when you use your **Start** layout to demonstrate functions from your computer. New commands, features and settings are highlighted in yellow. This new function is designed to help you easily find these items in the interface. After a feature has been used several times its color will revert to the normal interface color. This feature can also be disabled (or enabled again) in the **Preferences** menu under **Interface > Highlight Features**.

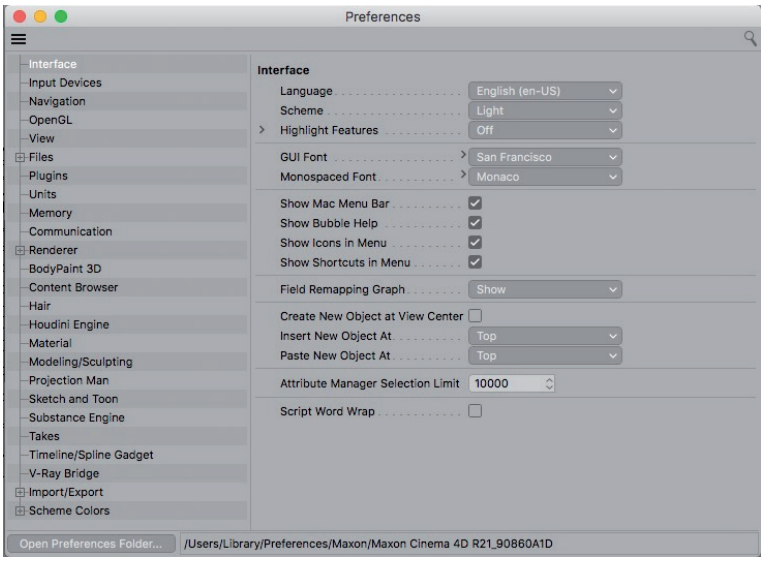
4.3 Useful Cinema 4D Presets

Before individual windows, icons and Managers are explained, many of the **Project Settings** (Edit menu) should be discussed.

You don't need to discuss all available options. At this phase, only some of the settings are of relevance:

4.3.1 Interface

Here you will find the **Scheme** setting.

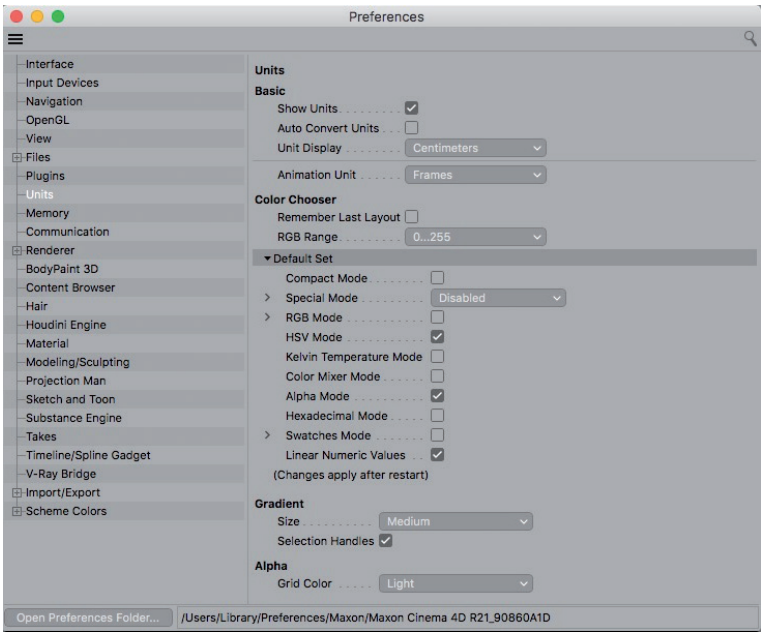


This setting defines which layout tone will be used – Light or Dark. Using the Light setting can be better for use with beamers, for example. The **Highlight Features** function that was previously mentioned should be disabled if you are completely new to Cinema 4D to avoid any confusion.

This menu also contains the **Show Shortcuts in Menu** option. Enable this option to help you get acquainted with the most important keyboard shortcuts (also called hotkeys). These will be displayed at the right of each command in the menus.

4.3.2 Units

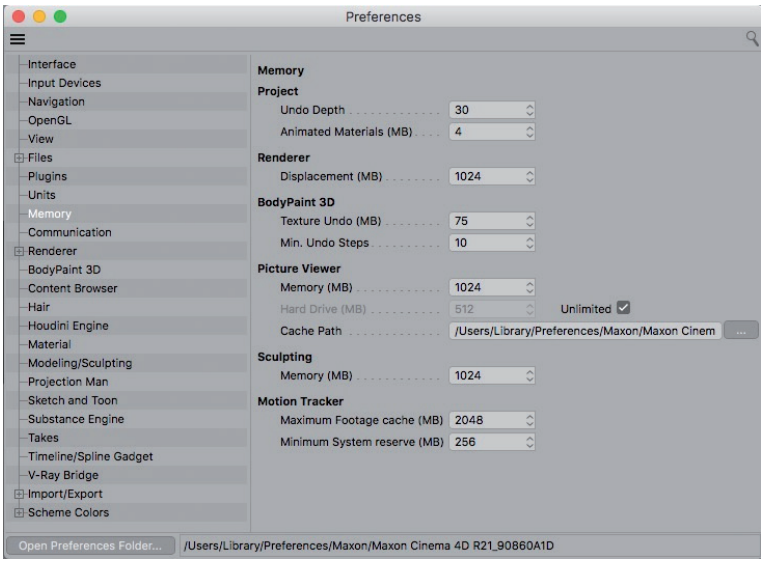
The **Units** menu you can define the units of measure that will be used for work in Cinema 4D. Select the units you are most comfortable using, e.g., **mm**, **cm** or **m**. Make students aware from the very beginning that Cinema 4D uses real-world units of measure. This is very beneficial, for example, when creating materials or when working with a camera, e.g., when setting up realistic depth of field.



You should also check the Color Chooser settings that are used to define the color of certain program elements. You can select from **RGB** or **HSV** values. As a rule, the **HSV** system is easier to configure. Otherwise the color values will be identical for both modes.

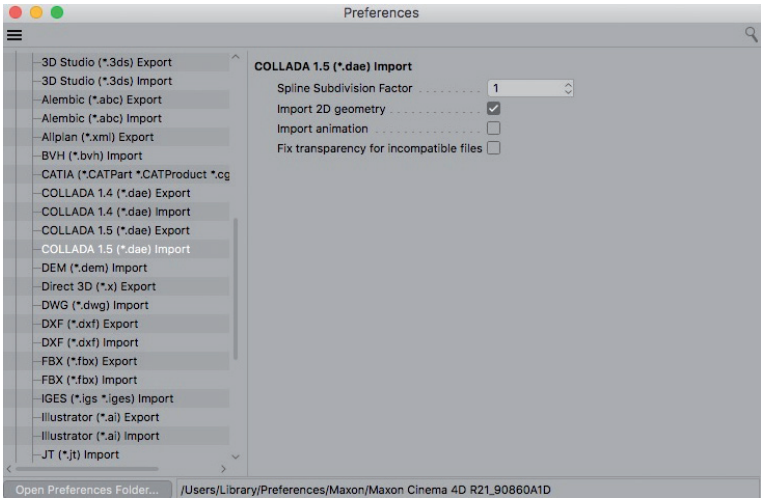
4.3.3 Memory

Explain the **Memory** menu's settings in the **Preferences** menu. Here you will find values for the maximum possible number of **Undo** steps, for example. Beginners in particular feel more comfortable if they know that they can undo numerous steps, if necessary. The default value is 30, which should be enough in most cases.



4.3.4 Import/Export

Next, you should take a look at the **Import/Export** menu to give the students an impression of the available exchange formats that are directly supported by Cinema 4D. Saving in foreign formats can be done in the **File/Export** menu.



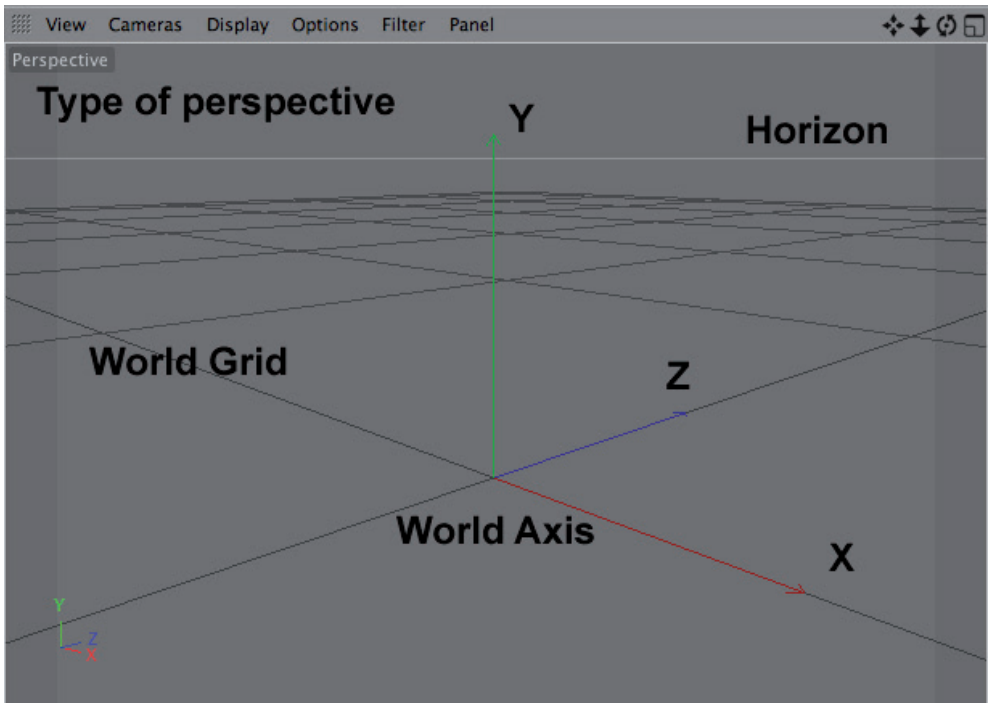
This is a good way to show graphics artists, for example, that a direct exchange with Illustrator is available. Users of other 3D applications can also find out which 3D exchange formats Cinema 4D offers. Let students know that importing into Cinema 4D is done automatically via the **Open** command in the main **File** menu.

4.4 The Standard Layout's Most Important Elements

The Cinema 4D layout is made up of menus along the top of the interface, icon palettes along the top and side of the Viewport as well as various Manager windows. Most of the layout is occupied by the 3D Viewport(s). This is where models are created and scenes are set up. Because being able to work with these windows is important for all subsequent instruction, students should practice using a hands-on example.

4.4.1 3D Viewport

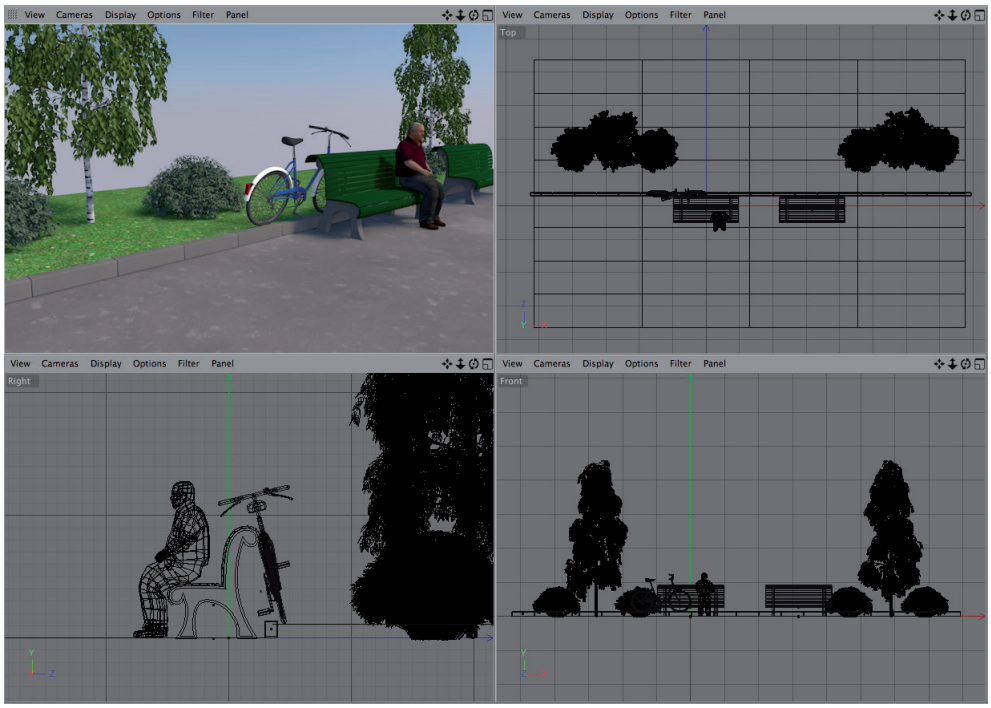
- Explain the visible elements: world grid, world axes, horizon, name of the Viewport at the top left corner, reduced axis display at bottom left.



- Explain the term "Perspective": A display type in which the shape and scale of objects depends on theviewer's position. The closer you are to an object, the larger it will appear. The focal length is simulated. Parallel lines in the scene will not necessarily be displayed parallel. This can be seen on the world grid.
- Advantages of using the Perspective view: Realistic depiction of objects, the angle of view and focal length can be freely selected.
- Disadvantages of using the Perspective view: The depiction of an object can be deceptive, which can negatively affect modeling.

4.4.1.1 Navigating the Perspective View

The Viewport can be navigated using the icons at the top right of the window or directly in the perspective view. Beginners should, however, get accustomed to using hotkeys to navigate directly in the view. To practice this, open the scene **1_ViewportNavigation**.



This and all other example scenes can be accessed in the *Content Browser's* **Presets/Curriculum** menu. The *Content Browser* can be found at the right edge of the layout.

- 1 key + left mouse button: translate view
- 1 key + right mouse button: zoom in or out in the view
- Alternative: use the scroll wheel to zoom in or out
- Alternative: 2 key + left mouse button: zoom in or out
- 3 key + left mouse button: rotate the view
- 3 key + right mouse button: rotate around the visual axis
- If **Cmd/Ctrl** is pressed while rotating, a special **Orbit** mode will be used that allows more fluid movements. This proves to be very advantageous when painting objects and when sculpting. If the **Shift** key is also pressed, the horizon will be kept horizontal.

Demonstrate how the cursor can be positioned in the Viewport so the rotation and zoom can be affected individually relative to the object. Using the icons at the top of the Viewport does not offer this feature and the rotation and zoom will always be centered to the Viewport.

4.4.1.2 Introducing other Views

Clicking on the 4th icon at the top of the Viewport or pressing **F5** will switch to a 4-panel layout. In addition to the **Perspective** view, this layout also includes **Front**, **Top** and **Right** views without perspective. Demonstrate how the navigation works in these views. The differences, e.g., when rotating, will be obvious. Clicking again on the 4th icon at the top of the Viewport will switch to a single view layout of the view for which the icon was clicked. Alternatively, the **F1** to **F4** keys are used to switch views. Also make students aware of the views' **Panel** menu, which is used to switch Viewport layouts. The middle mouse button (or scroll wheel) can be used to quickly switch from a single Viewport to a 4-panel layout.

The Viewports can be scaled by clicking and dragging on the respective outer edges. Double-clicking on the point at which all views meet will restore them to equal size. Alternatively, the **Panel** menu can be used to define the Viewport **Arrangement**.

4.4.1.3 Introduce the Filter Menu's Options

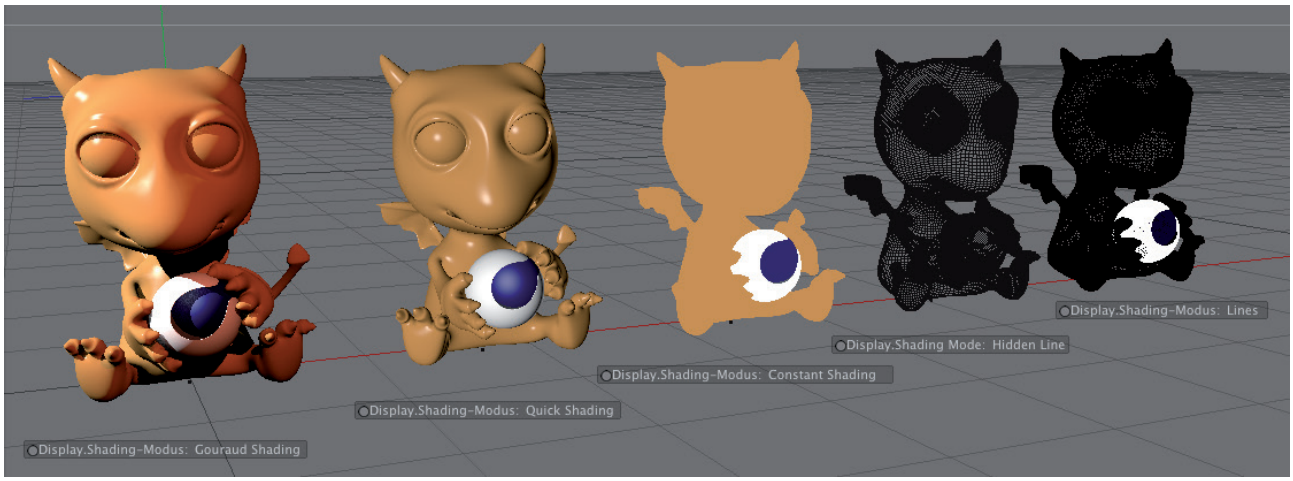
You don't always want all objects to be displayed in the Perspective view. The **Filter** menu lets you define which elements should – or should not – be displayed.

4.4.1.4 Demonstrate Shortcuts for Navigation

Each view's **View** menu offers commands that let the view be centered to a selected object or world coordinate system. Selecting the **Frame Default** command, the view can be reset to its original position in the world coordinate system. Selecting the **Frame Selected Objects** command (hotkey **O**), will center the view on the selected object. Furthermore, a blue arrow at the edge of the Perspective view shows the direction in which a selected object lies if it is not visible in the Viewport. Clicking on the arrow will center the Viewport on this object.

4.4.1.5 The Various Quality Levels in the 3D View

Objects can be displayed at various levels of quality in the Viewport.



It's not always advisable to use the best quality display option, particularly during the modeling phase where a lower quality display is generally faster. Objects in Cinema 4D are made up of **Polygons**. Polygons are simple surfaces with either three or four corners. The polygons' corners are connected by straight lines called **Edges**. The area bordered by these edges is the surface. One of the primary differences between the various display modes affects how these surfaces and their edges are displayed.

As a matter of principle, the realistic display of a surface produces a solid look. Edges or surfaces that lie behind other surfaces from the point of view of the camera will not be visible. The lower-quality display of surfaces as a wireframe can on the other hand be useful when modeling in order to be able to view an object's entire structure.

To compare the various types of display in the Viewport, open the Project **2_DisplayQuality** via the ***Content Browser***. Point out that the **Gouraud Shading** mode is the only mode that can also realistically display lighting. **Quick Shading** is similar but only uses the Viewport's **Default** light. Briefly introduce the Default light's settings in the Viewport's **Options** menu. Demonstrate how the display of edges or isoparms can be combined with shadows and show that the display quality can be defined individually for each view

4.4.1.5.1 OpenGL

The display quality can be improved by using the **Enhanced OpenGL** option in the Viewport's **Options** menu.

Brightness and highlights as well as shadows and the quality with which materials are displayed can all benefit from this option. The degree of improvement depends on the graphics card and corresponding drivers that are used. A quick test can be made by selecting the **Show OpenGL Capabilities** command in the **Project Settings' OpenGL** menu. Briefly explain that additional material and light properties can be displayed directly in the Viewport if **Enhanced OpenGL** is enabled. This includes reflections, displacement and shadows as well as more specialized effects such as Ambient Occlusion and blurriness. We will discuss this in more detail when working with light sources and materials, etc.

Point out that the display quality in the Viewport is not quite as good as the rendered result, despite its quite extraordinary OpenGL capabilities. This particularly affects the display of shadows, refracted materials and correct reflections between objects. Nevertheless, there are numerous situations in which the display quality in the Viewport will be good enough for accurately assess a scene for final rendering.

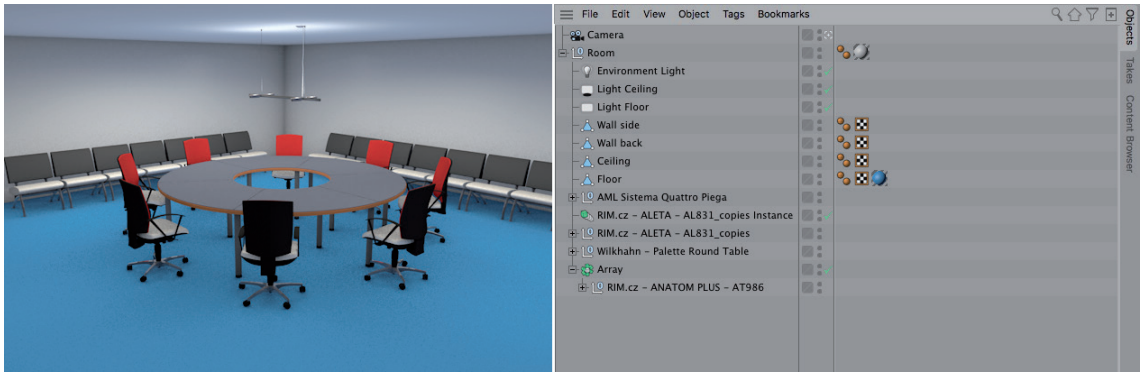
SUMMARY: LAYOUT

- In the Project Settings menu you will find fundamental settings that affect units of measure, color schemes and import/export settings.
- Displaying hotkeys (optional) in the menus makes learning these easier.
- Cinema 4D uses real-world units of measure for modeling and scene layout.
- Inadvertent modifications of the layout can be undone by selecting the Standard layout.
- The Perspective view offers a view of the scene through a camera and simulates real perspective.
- The keys 1, 2 and 3 as well as the mouse wheel can be used to navigate within the Viewport.
- If **Ctl/Cmd** is pressed while rotating, the **Orbit** mode will be used; also pressing **Shift** will lock the horizon.
- Display menu commands make it easy to center the view to an object or world coordinate system.
- The F5 key or clicking the MMB or mouse wheel switches to a 4-panel layout.
- The 4-panel view contains 3 additional views: Front, Top and Right. These do not simulate real-world perspective and are especially well suited for use while modeling.
- The number and arrangement of views can also be defined via the Panels menu.
- The display quality can be defined individually for each view via the Display menu. The Gouraud Shading setting offers the best quality and the wireframe settings are the simplest display types. Many display types can be combined with the wireframe display.
- Enhanced OpenGL should generally be enabled in the Viewports' Options menu to increase display speed and quality.
- The quality of the OpenGL display can be increased using reflections, shadows, depth of field, displacement and Screen Space Ambient Occlusion (SSAO) to make it resemble the final rendering as closely as possible.
- The Viewports' Filter menu can be used to hide specific object types or elements.

4.4.2 The Object Manager

The ***Object Manager*** gives an overview of objects and elements in the Project. It also makes it possible to create object groups and, for example, to control the visibility of objects. The ***Object Manager*** can also be used to arrange and assign **Tags**. Tags are properties or expressions with which the look or behavior of objects can also be affected.

Open the scene ***3_ObjectManager*** to demonstrate how to use the ***Object Manager***.



- Double-click on an object name to rename it.
- Objects can be re-arranged via drag & drop.
- Object groups can often be created automatically when a new object is created. For example, if the **Alt** key is pressed when a new object is created, the existing and previously selected object will automatically be made a Child object of the newly created object. This is useful when creating multiple Generators. If the **Shift** key is pressed when a Deformer is created, the Deformer will automatically be made a Child object of the currently selected object in the ***Object Manager***.
- Click on an object name to select the object – press Backspace or Delete key to delete the object.
- Various options for selecting multiple objects are available:
  - Click and drag a selection box around the object you want to select
  - **Cmd/Ctrl** + click on individual objects
  - Click on an object and **Shift** + click on a second object to also select it and all objects in-between
- **Cmd/Ctrl** + drag & drop to duplicate objects.
- Add objects to layers via the first symbol in the column next to the object name. Layers can be used to achieve a better overview of scene elements when working with complex Projects. For example, the visibility of objects can be controlled independently of their grouping in the ***Object Manager***.
- Control object visibility using the small dots next to the object name.
- Demonstrate the effect that changing object visibility within the object hierarchy has.
- Refer to the alternative **Solo** function with which selected objects and their hierarchies can automatically be made visible in the Viewport – and all other object are hidden.
- Point out the function of the green check mark next to the objects. Introduce the term ‘Generator’ object.
- Generally speaking, Generator objects are objects that create a certain type of geometry. Formulas and programs are used to “generate” the shape. If a green check mark is turned off, the Generator object will not be calculated. This can save render time for complex objects and has a different effect than hiding an object (turning off its visibility).
- Tags are displayed in the right column next to the object. Some tags are assigned automatically and many more can be assigned individually. One tag that can be applied to almost any object is the **Phong** tag. This tag controls the shading on the object’s surface. Shading is the combination of light and shadow on the surface. This effect can be effectively demonstrated using a sphere displayed using Quick or Gouraud Shading. Decreasing the Phong Angle affects the shading transition between polygons.

4.4.2.1 Additional Object Manager Options

Several symbols are located at the top right corner of the ***Object Manager***. Clicking on the **Magnifying Glass** will open a search bar in which a search term can be entered. Clicking on the **House** symbol will display a path bar. If an object is selected and the **Set as Root** command is selected in the ***Object Manager's*** **View** menu, only those objects will be displayed that are grouped under the root object.

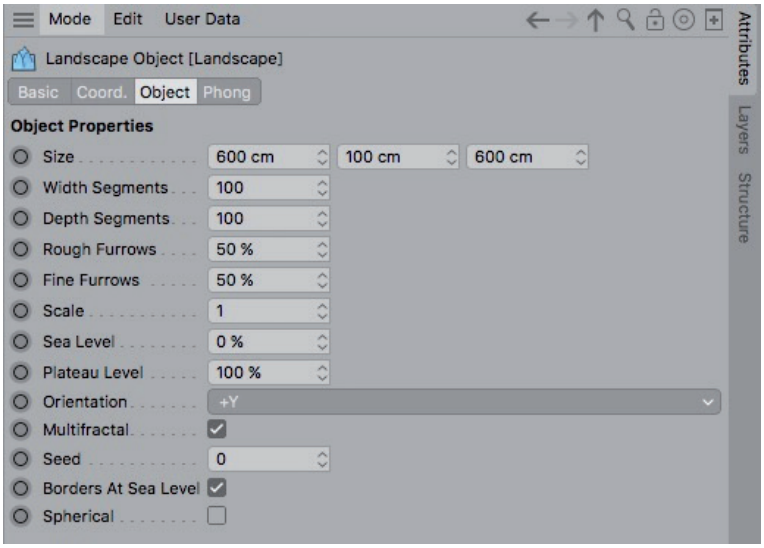
Click on the black **arrow** in the path bar to move the root stepwise upwards in the hierarchy. Click on the **House** symbol to reset the display of all objects in the ***Object Manager***.

Click on the **Eye** symbol to open a new area in which the type and number of all objects, layers and tags is shown. Double-clicking on an item in the list will select all elements of that type in the ***Object Manager***.

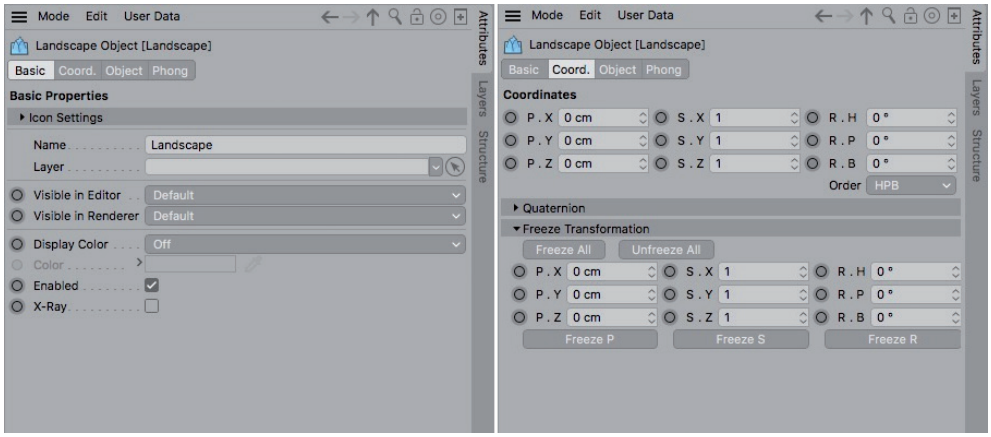
Click on the **Plus** symbol to open or close the ***Object Manager***. This makes it possible to use several differently configured ***Object Manager's*** at the same time. Alternatively, Cinema 4D's **Window** menu can be used. Here you will find a sub-menu for opening additional Managers, through which additional ***Object Manager's*** can be opened. However, a single ***Object Manager*** is enough in most cases.

4.4.3 The Attribute Manager

Whereas the ***Object Manager*** is mainly designed for organizing objects and hierarchies, the ***Attribute Manager*** lets you access all of a selected object's or element's settings.



Because of the great number of settings they are divided into tabs. Many tabs are always present, e.g. for objects while others appear according to the type of object they serve. Tabs that are always present for objects are the **Basic** and **Coordinates** tabs.



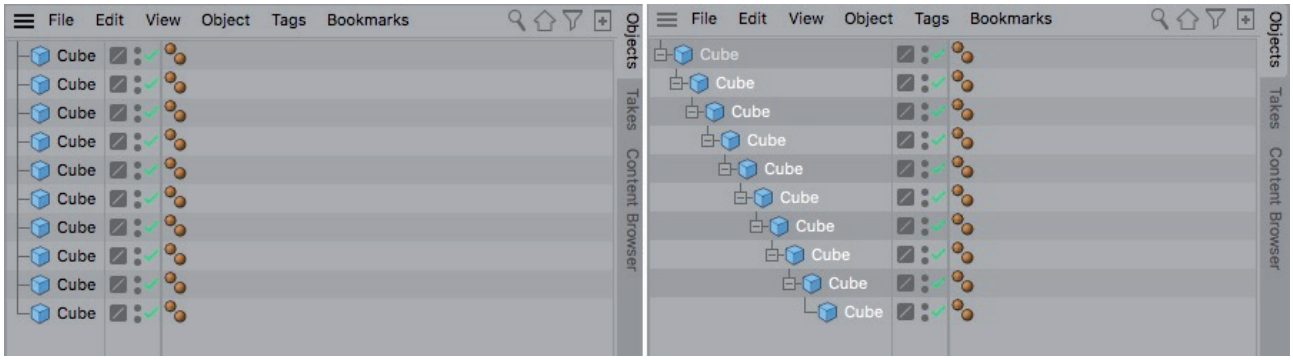
4.4.3.1 The Basic Tab

Here, many settings already mentioned in the ***Object Manager*** section are also present. In addition, icon settings, for example are also available that can be used to select a custom icon image for the object in the Object Manager. A bitmap can be loaded or the ID number of an existing icon within Cinema 4D can be used. These IDs can be ascertained in the Customize Commands (Window/Customizaton). A separate color is also available that can be used to color the object icon. The **Use Color** setting lets you assign a custom color to the object. Depending on what's selected in the Use Color menu, this color will be visible as long as no material has been assigned to the object. When the object is rendered, any material assigned to it will receive preferential treatment. The **X-Ray** option can be used to display the object with 50% transparency. This makes it easier to work with objects that lie behind or even within this object. However, when rendered, this object will be rendered opaque. Actual transparencies for rendering can only be created using the **Display** tag or by applying materials with transparencies.

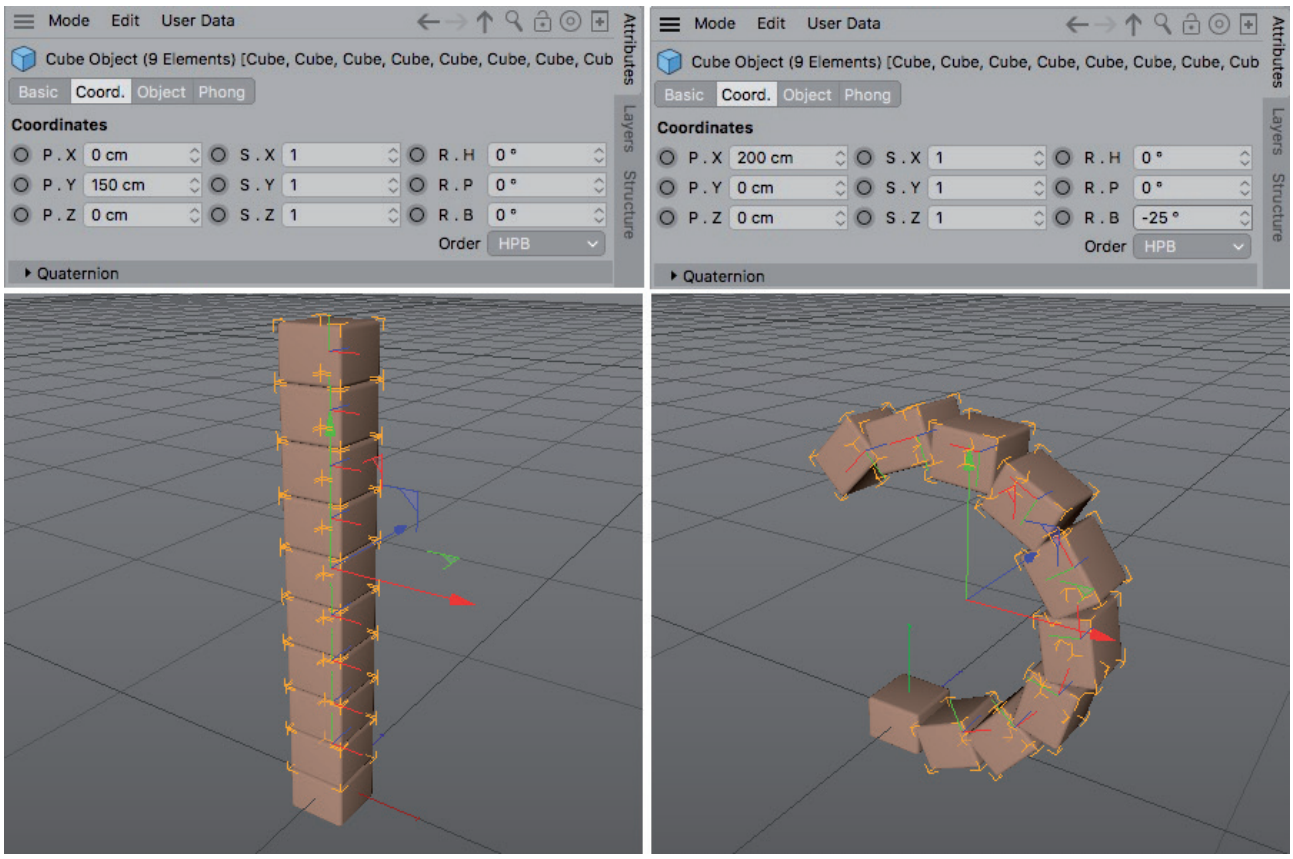
4.4.3.2 The Coordinates (Coord.) Tab

All objects have their own axis, which is called the **object axis**. This axis is oriented along the world coordinate system but can be modified freely. Object axes can be moved, rotated or even scaled. These modifications directly affect the object., which means that the object can be positioned anywhere in 3D space.

If precision is required, the axes' values can be modified by manually entering exact position, scale or rotation values in the designated fields. These values can be accessed in the ***Attribute Manager's*** **Coordinates** menu. Note that these values always apply only to the Parent object. Hence, the object hierarchy in the ***Object Manager*** affects the coordinate values. Use the scene **4_AttributeManager** to demonstrate how this works.



This scene contains several cubes as well as other items. Group these cubes in the ***Object Manager*** so each cube, except for the top-most cube, is arranged beneath another cube. Then select all cubes, except for the top-most cube, and modify the **P.Y** value in the ***Attribute Manager***. The cubes will stack on top of each other to form a column. Set the P.Y value back to 0 and this time modify the **P.X** value to position the cubes in a row. Finally, modify the **R.P** value to curve this row like a scorpion's tail and explain how this effect was created.



Interject the following information about the Color Chooser after the individual coloring of objects in the Viewport has been discussed.

- The HSV system is the standard selection system and its color slider makes it very intuitive to use. The other sliders are used to define the saturation and brightness, respectively.
- The RGB system is useful when exact color values are needed. RGB values are often used.
- The color selection via the Kelvin color temperature is a physical type and is useful in particular in conjunction with light sources since the Kelvin value of light sources is often noted on the light's packaging.
- The Mix Mode makes it easier to define two colors between which a gradient should be created and from which a color can be taken at any point along the gradient
- The pipette can be used to grab a color from any location on the monitor, including locations outside of Cinema 4D itself.
- The Color Wheel is an alternative element for the HSV system and can be scaled by right-clicking on it, or switched to Artistic mode in which those colors used most often are given more space on the wheel. In addition, various modes are available, which, for example, make it easier to select color values or complimentary colors.
- A color Spectrum displays the saturation and brightness of a selected color value.
- The Color Form Picture Mode lets you load an image onto which any number of Color Choosers can be placed.
- Selected colors can be placed in folders, saved or loaded for use elsewhere in the Project or in other Projects.

4.4.3.2.1 Standard Tools for Moving, Scaling and Rotating

If precision is not required, objects can also be positioned manually. To do so, use the **Move** tool with the hotkey **E** in **Use Model Mode**.

The hotkeys **T** and **R**, respectively, are used to scale or rotate. Also point out the corresponding icons in the top icon palette.

Mention how these tools can be restricted to specific axes:

- Click directly on an axis or rotation band
- Use X, Y, Z icons/keys
- Use the colored corners of the object axes' coordinate system
- Manipulate grid/quantize by pressing the **Shift** key

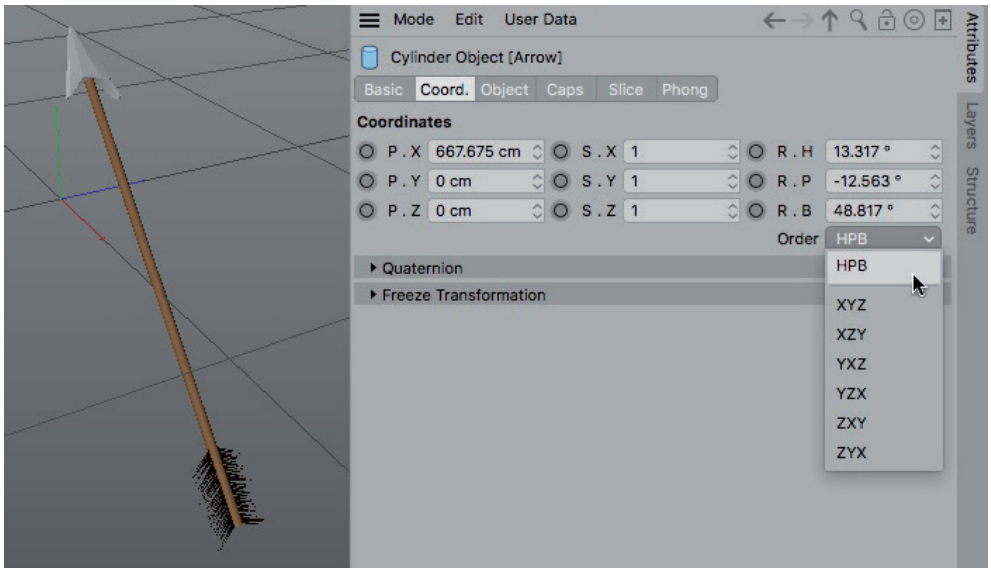
Using the **Per-Object Manipulation** option when using the **Move**, **Scale** or **Rotate** tools, the same effects can be achieved as was done when we created the scorpion's tale.

Explain the meaning of **H**, **P** and **B** (Heading, Pitch, Banking) for rotation. Initially, H is the rotation around the Y axis, P around the X axis and B around the Z axis. Clearly explain that multiple values can change when an object is rotated, even if it's rotated around only one axis. The **Freeze Transformations** settings in the *Attribute Manager*'s **Coord.** tab helps rotate an object in several directions if specific rotations are only supposed to occur around a single axis and at specific angles.

Frozen Position, Scale and Rotation values will set the values in the Coordinates settings to 0 and makes it possible to again enter specific values for **R.H**, **R.P** or **R.B** for a specific axis. The value can be frozen or reset at any time without affecting the object's position.

Briefly discuss the **order of rotation**. The HPB rotation system offers the advantage that the order in which rotation is applied to an object has no meaning for the final position. When using the other rotation systems, a fixed order of rotation must be adhered to. This can, for example, be advantageous when animating if an object rotates between keyframes when it's not supposed to. The **Rotate** tool offers a preview of the rotation bands. Use the scene **5_RotationOrder** to demonstrate.

The scene shows an object that needs to be rotated.



Enable the **Rotate** tool's **Gimballing Rotation** option and select the object to be rotated so you can see its rotation bands. Use the different rotation types in the *Attribute Manager*'s **Coord.** settings and observe how the rotation bands change. Some modes are better suited for certain rotations than others.

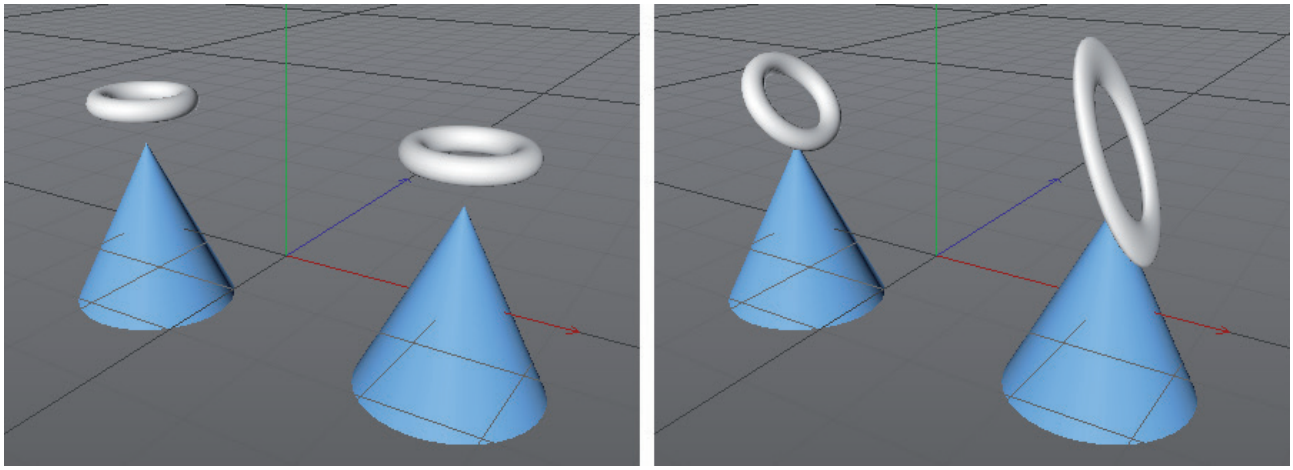
Nevertheless, problems can still occur when rotating objects from unique positions, in particular if the object's Z axis is perfectly vertical, such as a character's upper or lower legs. Unwanted pivots can occur during animation. This is referred to as Gimbal Lock. In these cases, the **Quaternion Rotation** can be applied to object rotations, which always calculates the shortest movement between two states. This function can be enabled in the object's **Coordinates** tab.

4.4.3.2.1.1 Scaling Objects

Explain both scaling options. Objects can either be scaled by scaling the respective axes or by scaling the shape (points, edges and surfaces). The scaling values in the ***Attribute Manager***'s **Coord.** settings only display the length of the object axes.

When scaling the axes, the object's points, edges and polygons remain in place – the scale of the object is increased or decreased because the reference system was scaled. When using the **Scale** tool, the length of the axes stays the same but the object's points, edges and surfaces will be modified.

Point out that scaled axes can lead to problems in the object hierarchies because scaling an object's axis system also affects that object's subordinate objects. If these are then rotated, unwanted distortion can occur. Use the scene **6_ObjectScale** to demonstrate. This scene contains two visually identical object hierarchies. The differences only become apparent when the subordinate objects are rotated.



Use this negative example to demonstrate to students that they should check the scaling of their objects precisely. If axes should purposely be scaled, explain the **Make Editable** mode. Use the **Mesh / Axis / Scale** command to reset inadvertently scaled axes to their default length.

4.4.3.3 Object Tab

This menu contains all object-specific settings. For a cube, for example, these include its dimensions and additional options for rounding its edges. Use a **Cube** primitive to demonstrate how a cube can be modified. Explain the special case of parametric primitives, which can only be scaled proportionally using the **Scale** tool. Individual scaling can only be done using the settings in the ***Attribute Manager***'s **Object** tab.

4.4.3.4 Additional Attribute Manager Options

There are several symbols at the top right of the ***Attribute Manager*** window. The ***Attribute Manager*** always displays the settings of the object or that was last selected. The **arrow** icons can be used to navigate between previously or subsequently selected items like you would in a browser window.

If an object is selected, the upwards **arrow** will jump up one hierarchy. The properties of an object that is not even selected can be modified. The upwards arrow can be clicked until the **Project Settings** menu is displayed. This can otherwise be accessed via the Cinema 4D Edit menu.

Clicking on the **magnifying glass** symbol will open a search bar. As soon as a single character or a series of characters is entered, all elements containing this series of characters will be filtered out and displayed. For example, entering the letter **P** for the **Coord.** tab will display all settings that contain the letter p.

Clicking on the **padlock** will lock the ***Attribute Manager*** and maintain its settings, even if another object or tool is subsequently selected. The last symbol works in a similar fashion, which opens a new ***Attribute Manager***, which is also locked using the padlock symbol.

The **target** icon restricts the ***Attribute Manager*** to the currently displayed parameter type. For example, if a cube was first selected and then the corresponding icon activated, only the object settings will be displayed in the ***Attribute Manager***. A tool's settings or a modeling function will no longer be displayed. This type of mode locking only makes sense if multiple ***Attribute Managers*** are being used that are configured differently.

An ***Attribute Manager***'s mode or the type of settings that it displays can be configured in its **Mode** menu.

4.4.4 The Coordinates Manager

The ***Coordinates Manager*** contains the same values as the **Coord.** tab in the ***Attribute Manager***. However, it also has an additional menu below the Position column, which lets you switch between **Object (Rel)** (relative),**Object (Abs)** (absolute) – depending on whether or not frozen transformations are used) and **World** coordinate systems. The center **Size** column can display either the axis length or the actual dimensions of a given object or even that of a hierarchy (**Scale +**). The **Return**/**Enter** key or the **Apply** button must be pressed to confirm changes made to any values in the ***Coordinate Manager***.

▶ *See: Exercises for object axes, coordinate systems and coordinates*

SUMMARY: OBJECT AND ATTRIBUTE MANAGER

- The ***Object Manager*** lists all objects contained in the scene and allows them to be grouped or arranged individually.
- Double-clicking on the object name to rename the object.
- The first symbol next to the object represents its layer system. This makes it possible to work with the object independent of the hierarchy.
- Click on the small dots in the center column to change their color. If the dots are green, the corresponding object will be visible. A gray dot is neutral and the visibility settings of Parent objects, if present, will be assumed. The top dot controls the object's visibility in the Viewport and the bottom dot controls the object's visibility for rendering.
- Objects that generate shapes have an additional green check mark next to them in the ***Object Manager***. If this is clicked upon and switched to an X, the object generation will be deactivated. This can help to reduce render times.
- The ***Attribute Manager's*** right column is reserved for tags. Tags contain properties and additional information that can be assigned to objects. Several of these tags, like the Phong tag, are assigned to many objects automatically. Other tags can be added manually, e.g., via the ***Object Manager's*** Tag menu.
- The ***Attribute Manager*** lies below the ***Object Manager*** and contains all parameters and options for the selected object. The settings in the Basic and Coord. Tabs are always present by default; other settings will be made available, depending on the object selected.
- The Basic menu's settings are almost exactly the same as the options described for the ***Object Manager***, which are, for example, used to define the name and visibility. In addition, display colors or special transparency effects can be activated.
- The Coord. menu contains information regarding position, scale and rotation of a given object. These values are always relevant to the object's axis system and are calculated relative to the Parent object.
- By using the Freeze functions in the Freeze Transformation menu, values can be frozen/saved temporarily, which can be very helpful when rotating an object to define exact rotation angles for specific axes.
- The order in which rotations are made has an effect on the object's final rotation position. Also, switching the order of the rotations can simplify an object's animation.
- The Rotation tool's Gimballing Rotation option can be used to find out which order should be used for rotation.
- The Move, Scale and Rotate tools can be used to manipulate objects. These tools can be restricted to one or more axes by clicking and dragging on one of the axes or axis bands, or by using the X, Y and Z icons in the top icon palette.
- If possible, scale objects using the object settings or by scaling in Use Model mode. Scaling axes in Use Object mode or by modifying the scale values can negatively affect subordinate objects.
- In the ***Coordinate Manager***, absolute and frozen object coordinates can be output or edited. The coordinates can also be made relative to the World coordinate system, for example to ascertain or define an object's position independent of its position in the hierarchy.
- The ***Coordinate Manager*** can also be used to display an object's or object group's actual dimensions.

4.5 Snapping

Early examples have shown that it can be quite laborious positioning objects precisely or on top of one another. This is where snapping can help. The Snap options can be accessed by clicking and holding on the horseshoe icon on the left icon palette. Alternatively you can press the **P** key to call up the options.

Various Snap modes are available.

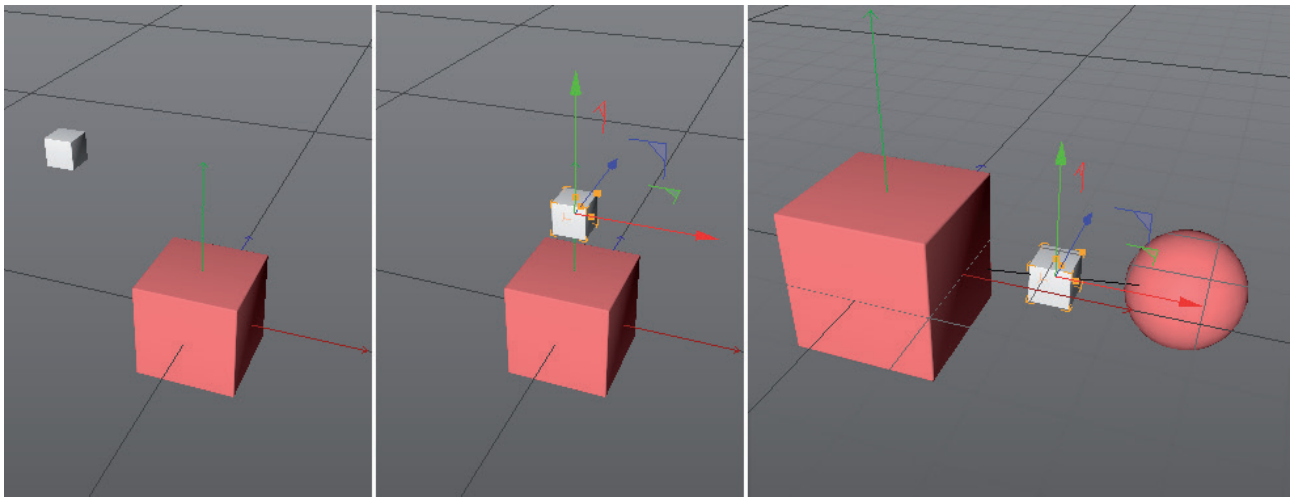
3D Snapping is the "real" snapping, with which objects snap to each other in three dimensions.

If **2D Snapping** is selected, objects will only snap on the current viewing plane.

With **Auto Snapping**, objects will snap either on the 2D viewing plane or in 3D space, depending on the Viewport in which you are currently working. When in the Perspective view, 3D snapping will be used and 2D snapping will be used in all other default views.

The following Snap menu options define to which elements objects will snap. Elements can also be combined, e.g., **Edge Snap** with **Mid Point Snap** to snap exactly on the center of a polygon edge.

Generally speaking, an object's rotation will not be affected by these snapping methods. Snapping occurs by moving objects relative to their axis system. To demonstrate, open the file **8_ObjectSnapping**.



1st Assignment: Center the small cube on top of the large cube without affecting its Y position.

Solution: Use 2D or **Auto Snapping** in the top view along with **Polygon Snap** and **Center Point Snap** options.

2nd Assignment: Position the small cube exactly in the center of the other cubes. Make the sphere visible in scene **8_ObjectSnapping**.

Solution: This can be more easily solved with the help of **Guides**. The Guide object can be found in the Parametric Primitives menu.

Step 1: Activate **3D** or **Auto Snap** in the Perspective view.

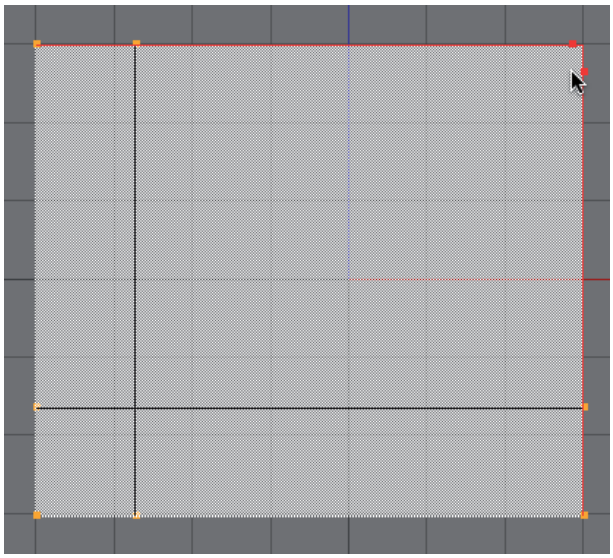
Step 2: Activate Axis Snap only and drag the ends of the guides to the axis of the sphere and the large cube.

Step 3: Several **Line Modes** are available in the ***Attribute Manager***. Select the Segment mode so the guide's ends snap to the end points.

Step 4: Combine the Snap options Guide Snap and Mid Point Snap to snap the small cube to the guide.

Point out the various modes for the Guide object, which also include Line and Plane types. When using multiple Guide objects, using the Guide tool is suggested. This tool can be found in Cinema 4D's main **Create** menu. Click to first create a line's end points and with the third click a surface is created. If you only click and drag, a line will be created. If a guide plane is created, a line can be created with the mouse that runs parallel from the plane's edge.

Red dots signal this function as soon as the cursor approaches the edge.

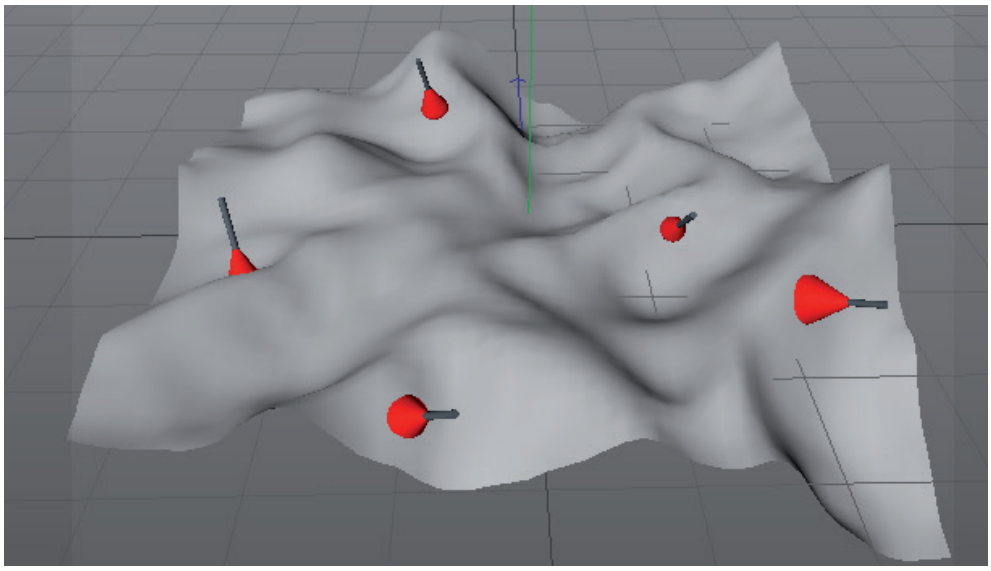


Intersecting guide lines offer additional snapping options for **Intersection Snap** and can, for example, be used to trace a building's outline.

You can also snap to the Workplane. This can be done using the **Workplane Snap** option in combination with the **Grid Point** or **Grid Line Snap** option. In **Workplane** mode, the Workplane can be moved or rotated using the **Move** and **Rotate** tools. Options such as Align Workplane to X, Y or Z can be used to position the Workplane parallel to the world coordinate system.

The Workplane can also be positioned and rotated interactively. Scene **9_WorkplaneSnapping** shows an example of this. Open the file, switch to **Use Model** mode and activate **Planar Workplane**. Also enable **Interactive Workplane**. As soon as the cursor lies over the object's surface, the Workplane will jump to the center of the corresponding polygon and will assume its angle. Select **Locked Workplane (Shift + x)** to lock the Workplane to this position.

If a new object is now created it will automatically be positioned perpendicular to the Workplane.



Other Workplane modes can be used to position objects perpendicular to the camera (**Camera Workplane**), parallel to the world axis according to the angle of view (**Planar Workplane**) or centered and parallel to the XZ plane of the currently selected object (**Axis Workplane**).

4.5.1 Quantizing

Quantizing is the rounding off of values, e.g., when moving or rotating an object. Quantizing is done by pressing and holding the **Shift** key *after* clicking with the left mouse button to rotate or move an object. Alternatively, the **Enable Quantizing** option in the Snap menu can be activated. This has the same effect only that the **Shift** key must no longer be pressed.

Quantizing is set to 10° intervals. Moving will take place in steps of 10 units and scaling in steps of 10% each. Custom intervals can also be used.

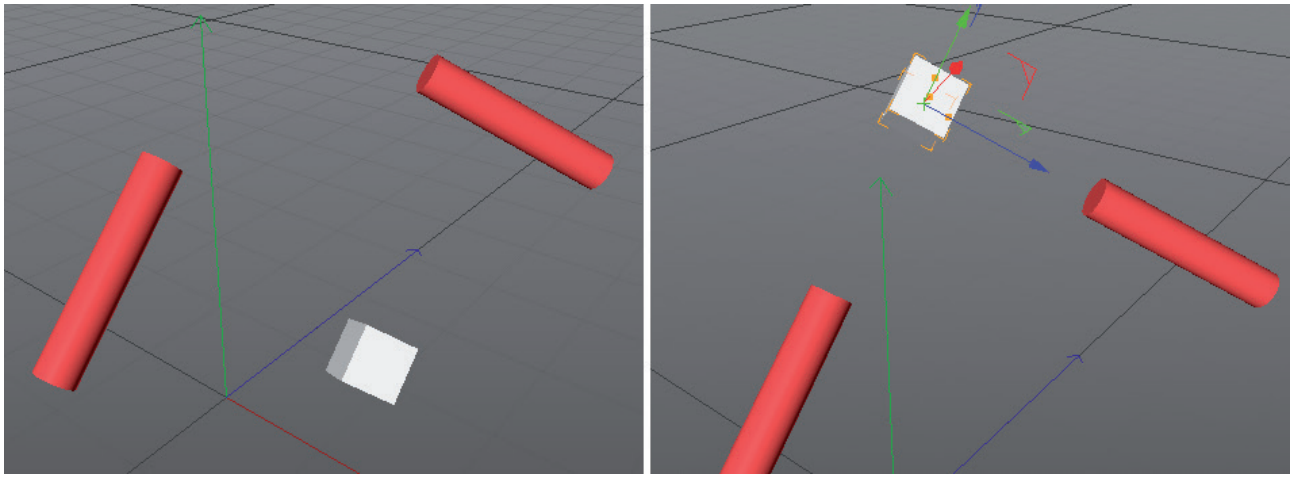
To do so, open the **Mode**/**Modeling** menu in the ***Attribute Manager***. In the **Snap** tab you will find several settings, including one that defines the snapping radius.

In the Quantize tab you will find various interval settings that can be defined manually.

4.5.2 Dynamic Guides

If **Guide Snap** is combined with **Dynamic Guide**, snapping will be possible in directions not specified by guide. Open the scene **10_DynamicGuides**.

Assignment: Position the cube at the intersection of the direction specified by the cylinder.



Step 1: Activate **Axis Snap** and **Intersection Snap**

Step 2: Move the cube to the center of the cylinder until it snaps into place and hold the mouse button pressed for half a second.

Step 3: Without releasing the mouse button, move the cube to the second cylinder until it snaps to its center.

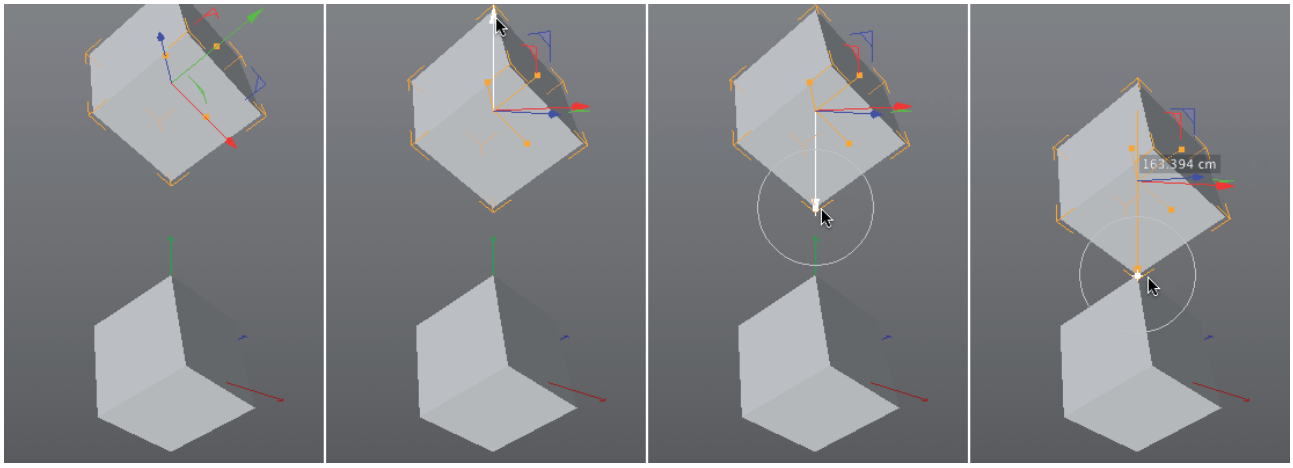
Step 4: Continue pressing the mouse button and, after waiting a moment, drag the cube in the direction of the cylinder extension until a small star is displayed in the Viewport. This is the intersection point that was calculated by the dynamic guides. Snap the cube to this point and release the mouse button.

4.5.3 Hidden Snapping Feature

Object axes can quickly be lengthened or even dragged in the opposite direction in order to use special snapping functions. Take a look at the file **11_Snappingspecial** to see an example.

Assignment: The scene contains two rotated cubes that are positioned so that their corners meet.

Solution: Normal snapping methods will not provide a solution in this case because they do not take an object's outer shape into consideration. However, a key combination can be used that lets the object axis be used for snapping.



Step 1: Since both cubes are rotated, neither of the object axes lies in a good position. This can be corrected if the world coordinate system is activated for the Move tool. The orientation of the objects' axes will be parallel to the world axis.

Step 2: Activate **3D** or **Auto Snapping**

Step 3: **Cmd**/**Ctrl** + **right click** on the top cube's **Y** axis. This object axis can now be scaled at will and dragged downward using the mouse until it snaps to the bottom corner of the top cube

Step 4: Keep the left mouse button pressed while the top cube is dragged downward until it snaps to the top point. The tips of both cubes now lie at exactly the same location.

SUMMARY: SNAPPING, HOTKEYS, GUIDELINES

- Snapping of elements and objects to one another can be activated via the Snap options.
- Various Snap options are available, e.g., Point, Edge, Polygon, object axis or guides.
- Additional options offer more specific snapping such as snapping to the mid point of an intersection.
- Guides can be added individually from the Primitives menu.
- Multiple guides or guide layers can be quickly created using the Guide tool.
- The Guide tool can be used to get additional guides from the edges of guide layers.
- Dynamic guides can be created by snapping to an element and keeping the mouse button pressed for about half a second.
- The world coordinate system or Workplane can be used for snapping.
- A Workplane's position and orientation can be edited manually in the Workplane mode using the Move and Rotate tools.
- Alternatively, various standard positions and automatic positioning based on the angle of view are also available.
- When in Interactive Workplane mode, the Workplane will position itself on the polygons underneath the cursor. To set the Workplane to this position it must be locked. This command should be assigned a hotkey in the Customize ***Commands Manager***.
- Quantizing can also be activated in the Snap menu or by simultaneously pressing the **Shift** key. Elements can then only be modified in uniform steps.
- The quantizing steps can be defined in the ***Attribute Manager***'s **Mode**/**Modeling** menu.
- **Cmd**/**Ctrl** + right clicking on an axis it can be scaled or used in conjunction with snapping.

5 Modeling

Modeling is the creation of shapes, to which colors or materials can be assigned, and which are also illuminated by lights in the scene. Various basic shapes and tools are used for modeling. The following sections introduce various primitives and helper objects for modeling. Subsequent sections will introduce modeling tools and various modeling techniques.

5.1 Parametric Primitives

Parametric primitives in Cinema 4D are basic shapes that can be modified using various settings/parameters, i.e., numeric values and options. Primitives bear the advantage that they are easily accessible, commonly used shapes that are easy to control, e.g., when scaling. The disadvantages only become apparent when a shape other than the object's standard shape must be created, which cannot be created using the object's available settings. In such cases, the object must be "made editable". However, results in the loss of its previous settings – but also gives us complete access to the object's points, edges and polygons.

5.1.1 Working with Parametric Primitives

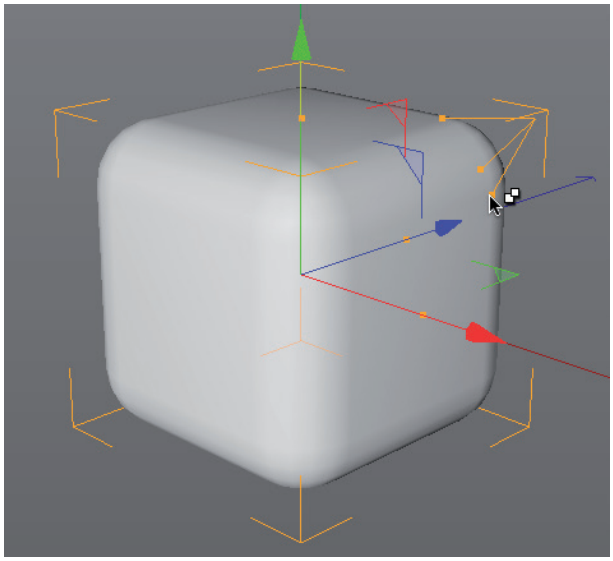
Primitives can be created via the top icon palette or the **Create/Primitives** menu. This object group contains three objects that are different from the others:

1. The **Null** object generates no visible shape and only consists of an axis. This is a helper object that can, for example, be used to group objects. Multiple objects can easily be grouped within a **Null** object by selecting all objects to be grouped and then pressing **Alt** + **G**. This will create a Null object as a Parent object for all selected objects. Null objects will not be visible when an image is rendered.
2. The **Guide** object is only designed for use with snapping and not for modeling shapes. This object will also not be visible when an image is rendered.
3. An empty polygon object can be filled with points and surfaces but is initially completely void of these and is in fact a **Null** object in this state. The **Polygon** Pen, for example, can be used to create points, edges and polygons for such an object. This object does not make an **Object** menu available in the ***Attribute Manager*** and therefore also doesn't offer any settings that relate to modeling.

Work with all other Primitives can take place using the mouse in the Viewport or using the object's settings in the ***Attribute Manager***'s **Object** tab, which generally offers more modification options than are available when working in the Viewport.

To edit a Primitive object, e.g., a cube, interactively, switch to Use Model mode and select the Primitive to be modified. The object will have small orange points (handles) on each axis, which can be dragged using the **Move** tool along the X, Y or Z axis to change the cube's size accordingly. Modifying the **Size X**, **Size Y** and **Size Z** settings in the ***Attribute Manager*** has the same effect.

Additional handles are made available when the **Fillet** option is enabled. The cube's edges can then be rounded by modifying the **Fillet Radius** value or by dragging the handles.

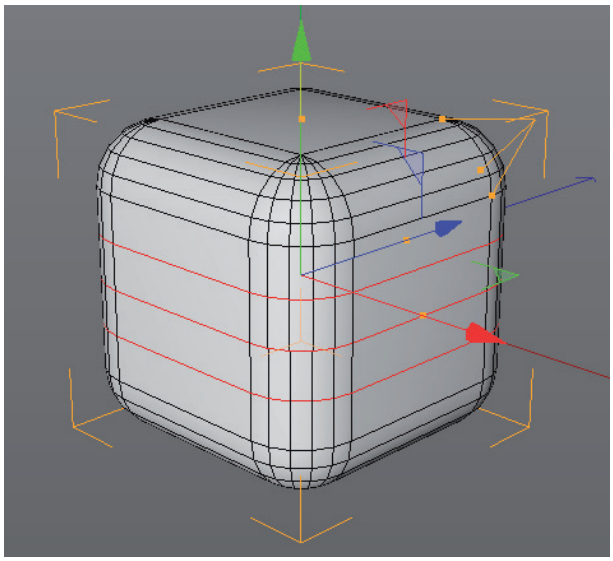


5.1.1.1 Segments

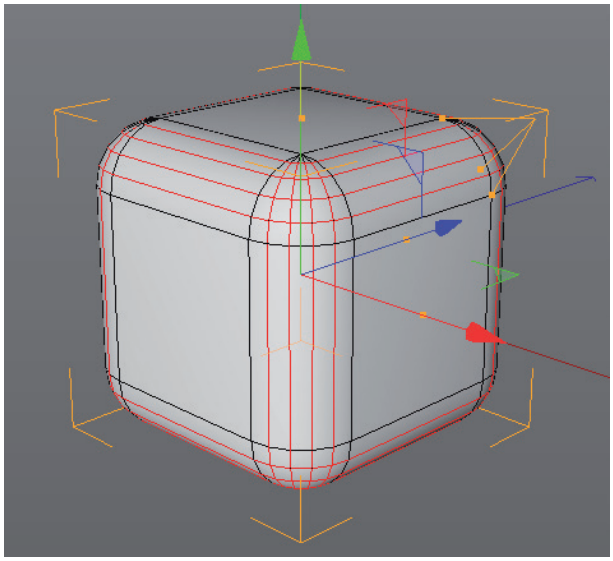
Segments represent the number of subdivisions, i.e., the number of points, edges and polygons that make up an object's surface.

Because more segments lead to correspondingly larger file sizes, the number of segments for each Primitive should not exceed the actual number needed.

The settings **Segments X**, **Segments Y** and **Segments Z** can, for example, be used to increase the number of a cube's segments but the quality of the cube as such is not affected.



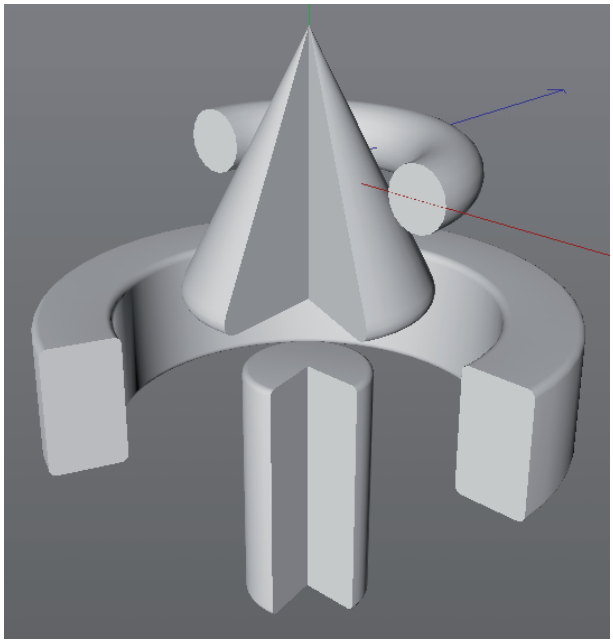
This can be seen in a display mode that shows the object's wireframe (e.g., **Gouraud Shading (Lines)** with **Wireframe**): additional lines will appear but the object's shape stays the same.



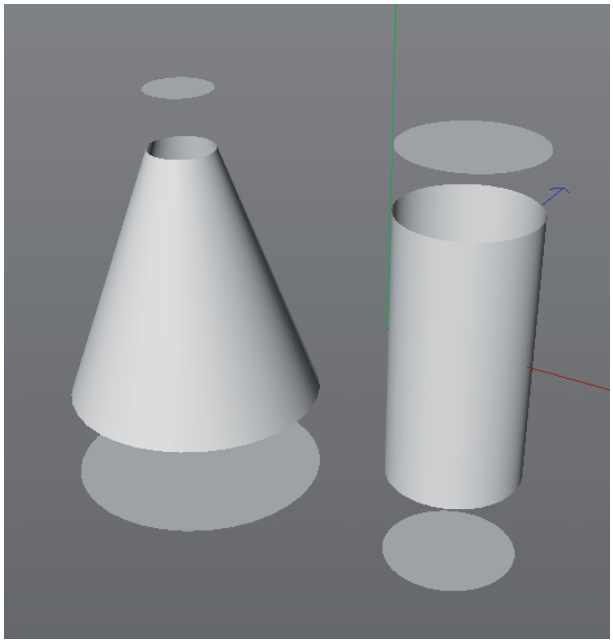
The **Fillet Subdivision** setting, however, can be used to define the rounded parts of an object. Therefore, increasing this value can help round the affected parts of the object.

5.1.2 Primitives' Options

- Demonstrate how to use the handles and the effect of various segment settings, also on other Primitives e.g., **Plane**, **Landscape** or **Sphere**.
- Demonstrate the **Slice** tab's options for the **Torus**, **Tube**, **Oil Tank** and **Disc** objects, which can be used to quickly slice an object.



- Objects such as the **Cylinder** and **Cone** objects offer additional **Caps** settings, which can also be rounded. Explain the term 'caps' as a two-dimensional surface that closes otherwise open regions of the object.



5.1.3 Examples

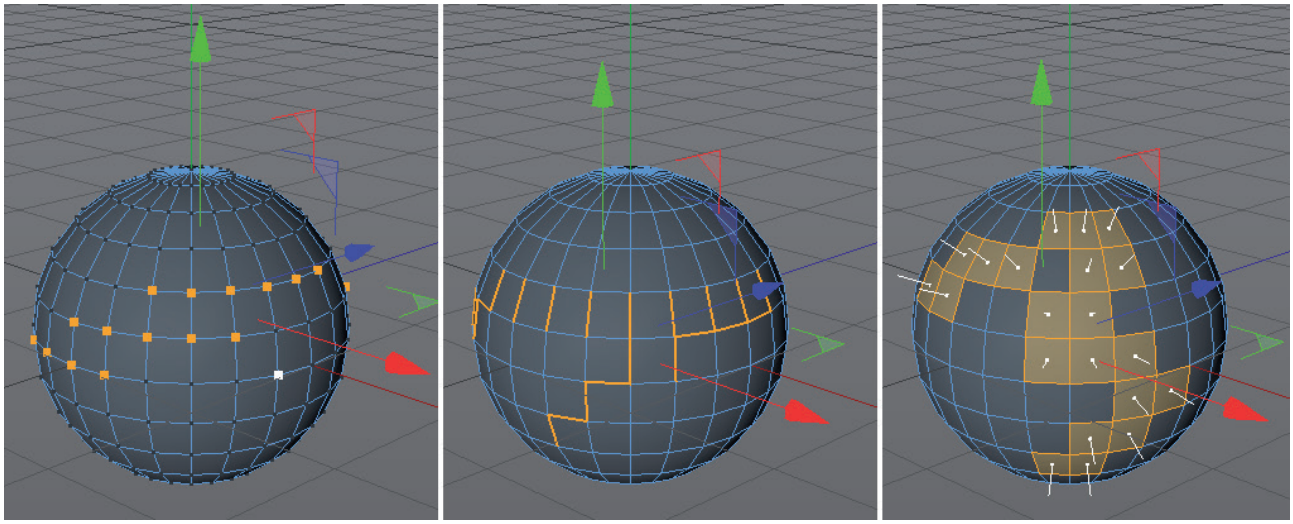
Practice how to create and modify various Primitives using the handles and basic move and rotate tools. A simple building should be created on a landscape. The correct types of Primitives should be used and positioned accordingly.

Additional examples can be the modeling of a game controller or a cartoon character. The scene **12_Primitive Examples** contains final objects that can be shown as examples.

5.1.4 Converting Primitives

Converting primitives is done using the Make Editable tool. Making an Primitive object editable means that a parametric object will be turned into an object consisting of points, edges and polygons. All previous object settings will be lost – often, only the **Basic** and **Coord.** tabs' options will remain in the ***Attribute Manager***. Therefore, converting an object must be well-planned.

The advantage of converting an object is that it can be edited in different ways. For example, sections of the surface can be removed, deformed or reshaped. Three modes are available for editing the object: **Point**, **Edge** and **Polygon**. A Primitive can be made editable by clicking on the icon at the top of the left icon palette, selecting **Mesh**/**Conversion**/**Make Editable** from the main menu, or by simply selecting the object and pressing the **C** key on your keyboard.



A converted object can be edited in any of the three aforementioned modes. For example, the **Move** tool can be used to move selected points with the mouse. Larger regions of an object's surface can be influenced simultaneously by selecting multiple elements.

5.2 Making and Working with Selections

We already know how to select objects, but it is also possible to select an object's points, edges and polygons once an object has been converted.

The following simple selection methods are available:

Selecting by clicking with the mouse, e.g., using the **Move** too. **Shift** + click to select additional individual elements. **Shift** or **Ctrl** + click on already selected elements to deselect them.

Disadvantage: An element can be inadvertently moved when clicked upon.

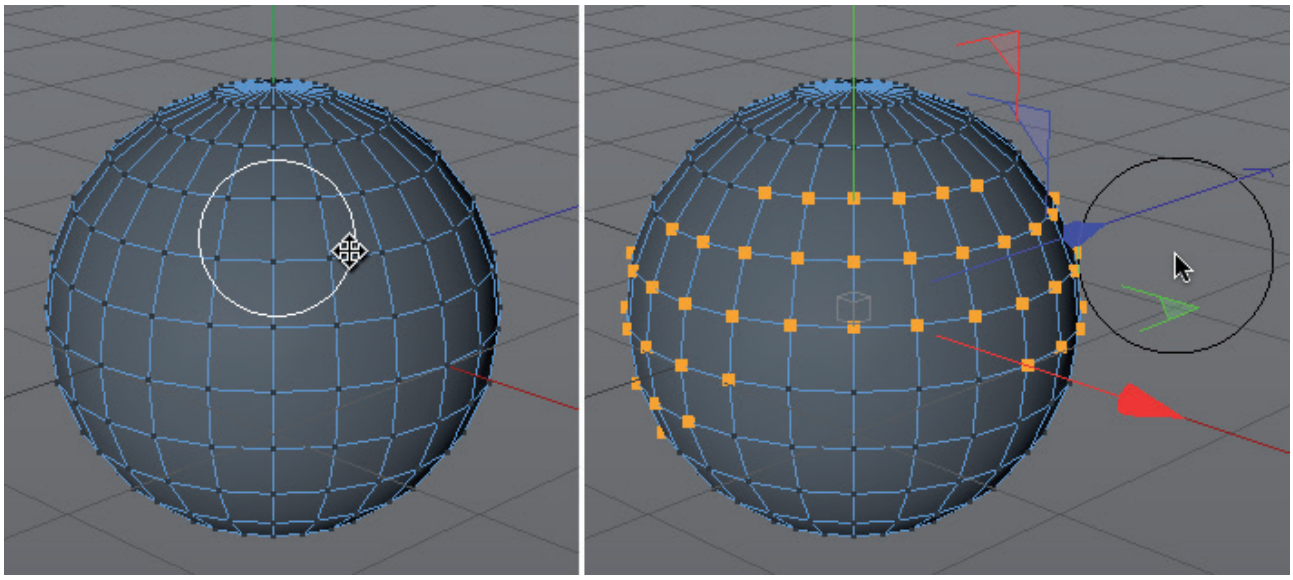
Note: If a shaded display type is active in the Viewport, only elements on the front side (what is visible) can be selected. Elements on the backside of objects can only be selected in Wireframe mode without having to navigate within the Viewport.

If the Move tool is selected and the right mouse button is pressed, a selection can be dragged around multiple elements. Again, elements on the backside of an object can only be selected if a wireframe display mode is active. Using **Shift** and **Cmd/Ctrl** keys works here as well.

5.2.1 Live Selection

This selection tool works like the right mouse button in conjunction with the **Move**, **Scale** and **Rotate** tools. However, it has several additional options that can be accessed via hotkeys or in the ***Attribute Manager***.

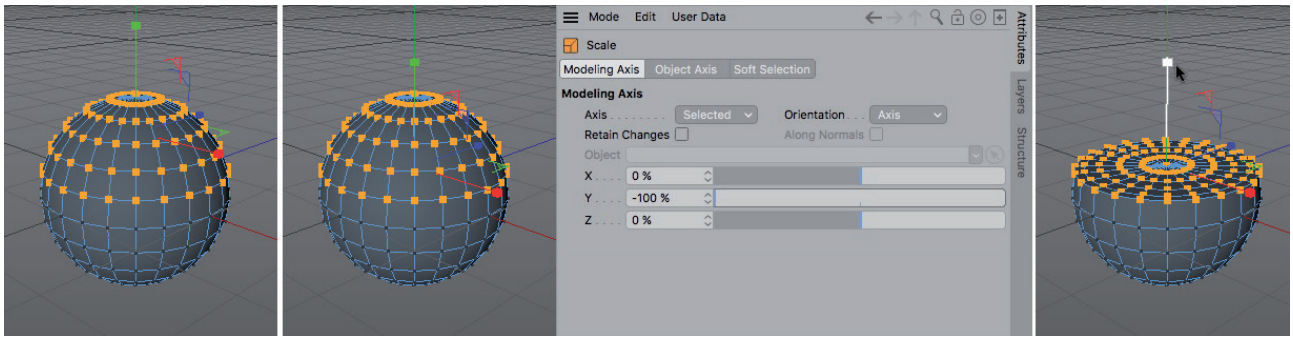
The **Radius** value defines the selection radius around the cursor. This can also be defined interactively with the mouse in the Viewport by pressing the middle mouse button or mouse wheel and moving the mouse to the left or right.



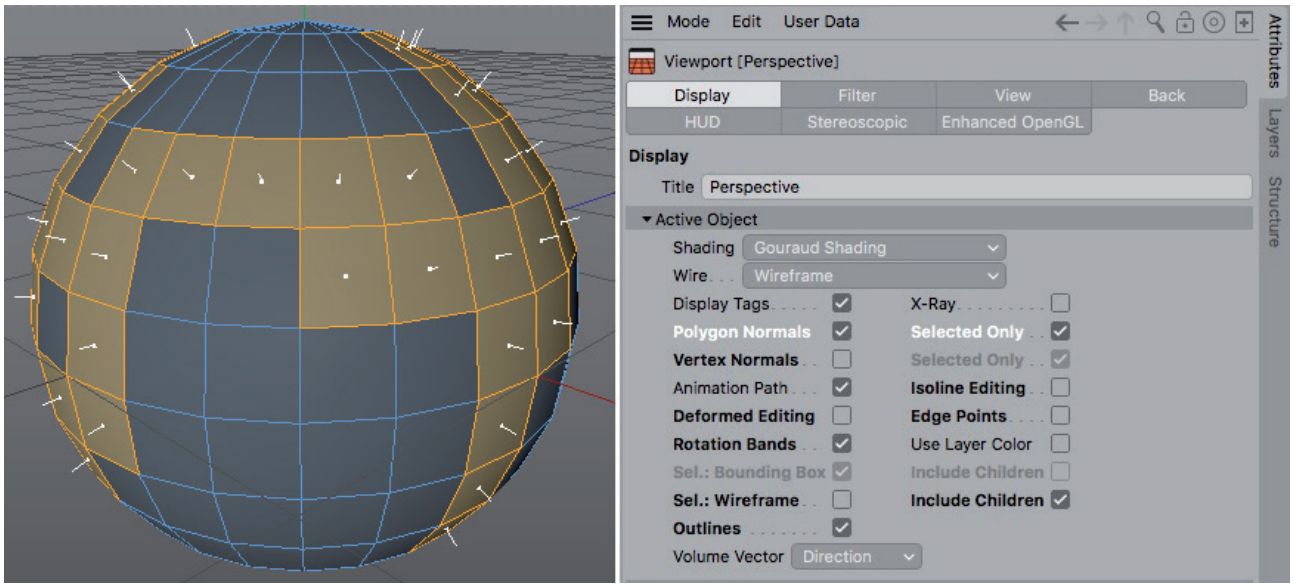
A very important option is the **Only Select Visible Elements** option. This option defines if only elements on the front side of the object will be selected (disabled) or if objects on the backside of the object (enabled) will be included in the selection. This option works independently of the type of display that is active in the Viewport. Hence, elements on an object's backside can also be selected even in **Quick Shading** mode if this option is enabled. The **Tolerant Edge/Polygon Selection** option is only relevant for **Use Edge** and **Use Polygon** modes and defines when an edge or surface will be selected: upon simple selection by a selection tool (enabled) or when an element has been enclosed completely by a selection tool's radius (disabled).

5.2.1.1 Modeling Axis

When selecting points, edges or polygons you will see that an axis is shown that is automatically centered to the selection. This modeling axis can be positioned independent of the object axis. Since the modeling axis acts as a point of reference for scaling and rotation, repositioning the axis can be useful.



The **Live Selection** tool offers the **X**, **Y** and **Z** sliders in its Modeling Axis tab in the *Attribute Manager*. The modeling axis' orientation can also be defined using the **Orientation** options. You can select between **Axis**, **Camera** and **World** coordinate systems or perpendicular to the selection, which is the **Normal** option. Normals are vectors lie on all polygons by default and as a rule they lie perpendicular to the polygon's surface. Normals can also be made visible in the Viewport. To do so, select **Display / Configure …** in the Viewport's menu and enable the **Normals** and **Selected Only** options in the **Display** menu in the *Attribute Manager*.



The normal will now be displayed as short white lines on each selected polygon. Normals are used for numerous modeling tools but are primarily by Cinema 4D to help calculate surface shading using light. A surface's shading can be calculated by taking into consideration the incidence of light relative to the Normals. We also discussed the **Phong** tag and its role in creating shading. This tag also uses Normals in its calculation. Other orientation settings for the modeling axis use the axis system of Parent objects. The modeling axis can only be rotated using the **Rotate** tool if **Orientation** is set to **Axis**.

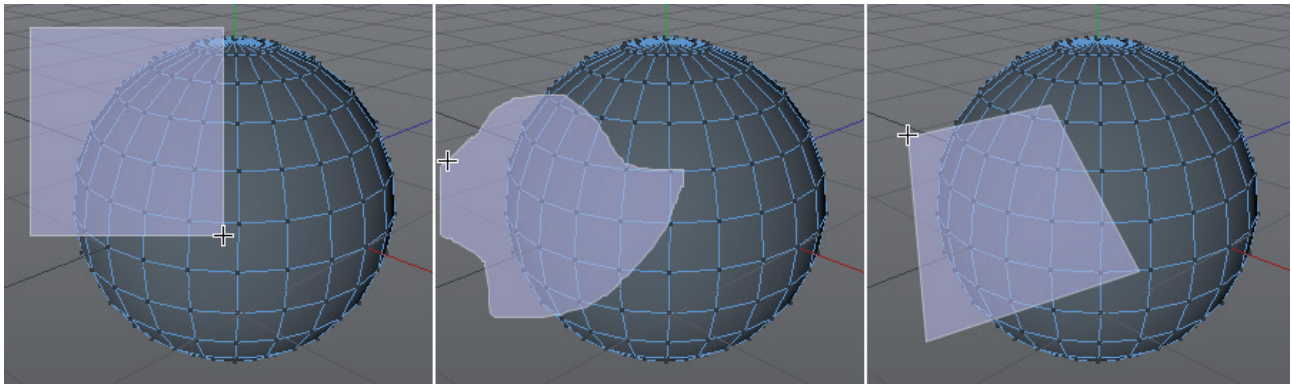
5.2.1.1.1 Enable Axis Mode

The modeling axis can be repositioned quite easily when in **Enable Axis** mode e.g., in conjunction with the **Move** tool. The modeling axis can be positioned on an object's point by simply clicking on that point. The **Enable Axis** mode can not only be used in conjunction with the **Use Point**, **Edge** or **Polygon** modes but also with the **Use Model** or **Use Object** modes. In these instances, the position of the object axes can be modified. However, this does not work with parametric Primitives. Their axis must lie at the center of the object. If a Primitive's axis is moved, the object will behave as if the entire object had been moved. The **Enable Axis** mode can be enabled or disabled using the hotkey **L**.

The fact that an object's axis or a selection's modeling axis is used as a reference for scaling and rotation is a great advantage for creating a wide variety of shapes. For example, a door modeled from a cube that opens at its hinges. If the cube's axis is positioned on the side of the object and the **Enable Axis** mode is subsequently disabled, the door can be rotated (opened/closed) correctly. The same applies to points, edges and polygons.

5.2.2 Other Standard Selection Modes

As can be done in graphics programs, a selection can be made by dragging a selection box or by freely drawing a selection area around elements. The same methods can be used in Cinema 4D using the **Rectangle**, **Lasso** and **Polygon Selection** tools.



These tools also have options in the ***Attribute Manager***, e.g., that can be used to define whether or not only visible elements should be selected.

5.2.3 Additional Selection Modes

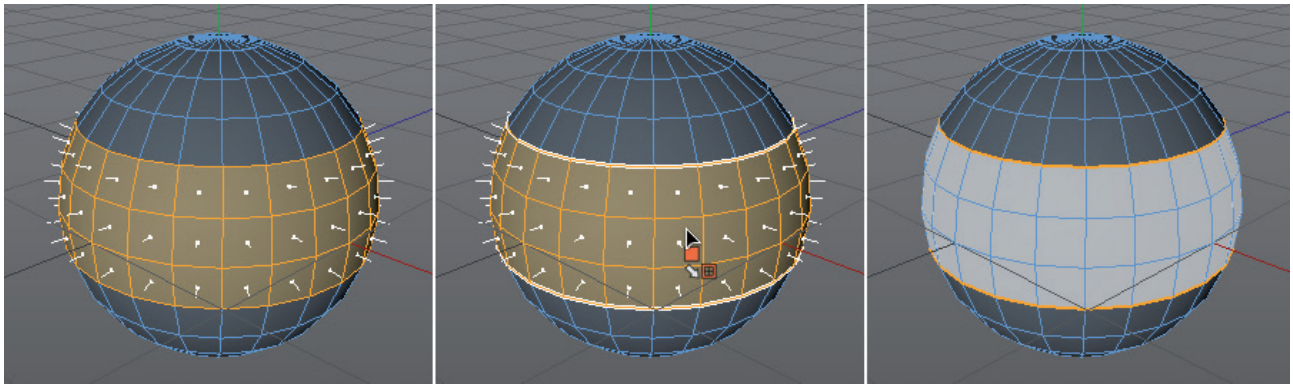
In many cases it's easier to select elements that should not be selected later. This is where the **Select** menu's settings come in handy. The Invert command will invert the selection. This menu also contains standard commands such as **Select All** or **Deselect All**.

Use the **Grow Selection** and **Shrink Selection** commands to expand or reduce the selected region, respectively, at its edges. The **Select Connected** command will automatically search for all elements that are directly connected at a common edge or a polygon of the already selected element.

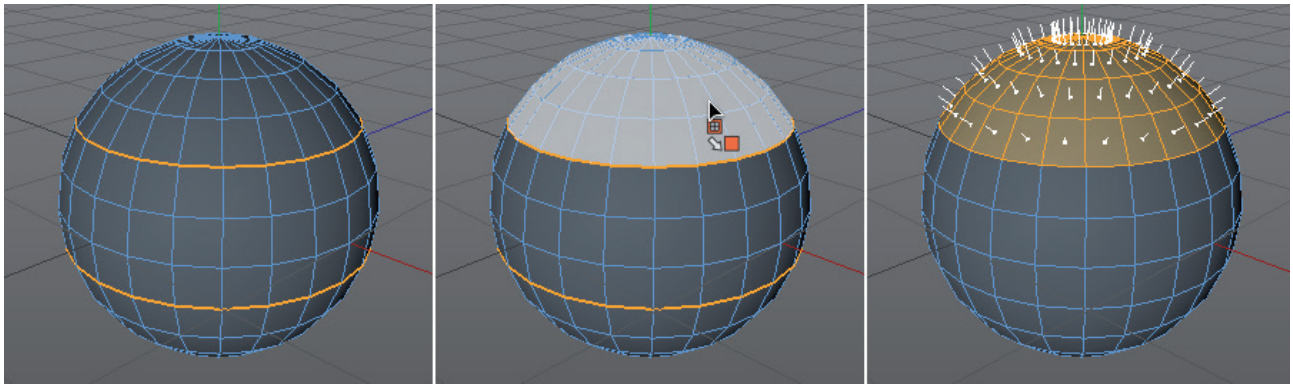
The **Loop Selection** and **Ring Selection** tools can be used to make a selection relevant to the position of the cursor on the object. **Loop Selection** will search for structures that continue in the direction of the edge over which the cursor lies. **Ring Selection** searches for structures that run parallel to the edge over which the cursor lies. Almost all **Select** tools can be used in **Use Point**, **Edge** and **Polygon** modes.

Loop Selection can also be made on an edge by double-clicking on it if the **Move**, **Scale** or **Rotate** tool is selected. Double-clicking on a polygon will automatically execute the **Select Connected** command.

If polygons are selected, the **Outline Selection** command can be used to create an edge selection using the polygons' contour. To do so, click once on the selected polygons.

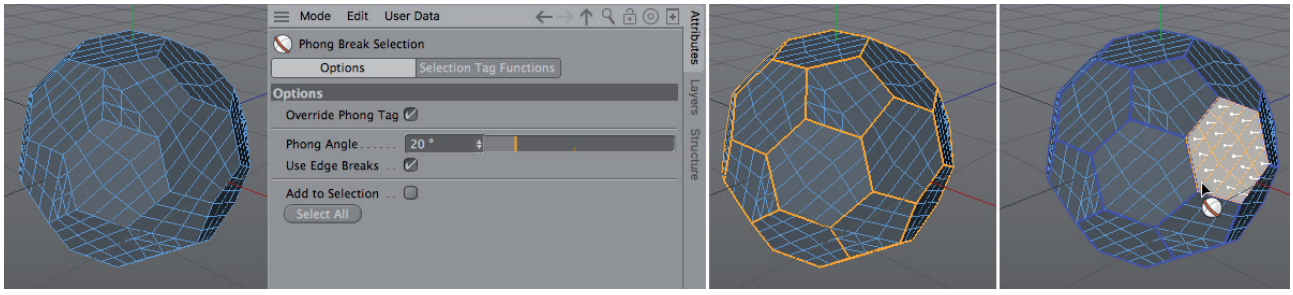


Fill Selection works similarly. If a region is restricted by an edge or polygon selection the **Fill Selection** command can be used to select the inner part or outer region of this selection. Here also the position of the cursor when the left mouse button is pressed determines which region will be selected.



Path Selection makes it possible to draw over a series of points or edges using the mouse to select them. If the left mouse button is pressed, a point-to-point or edge-to-edge selection can be drawn.

With **Phong Break Selection**, points or edges can be selected where certain surface angles are exceeded.



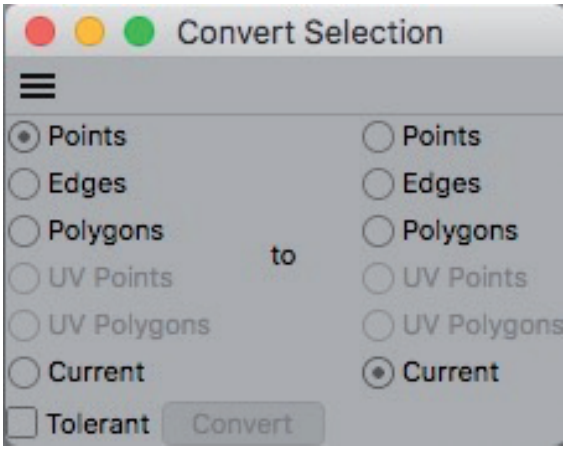
The same principle is used by the **Phong** tag, which is used to define the surface shading where an angle can be used to define which edges should be hard and which should be shaded.

5.2.4 Converting and Managing Selections

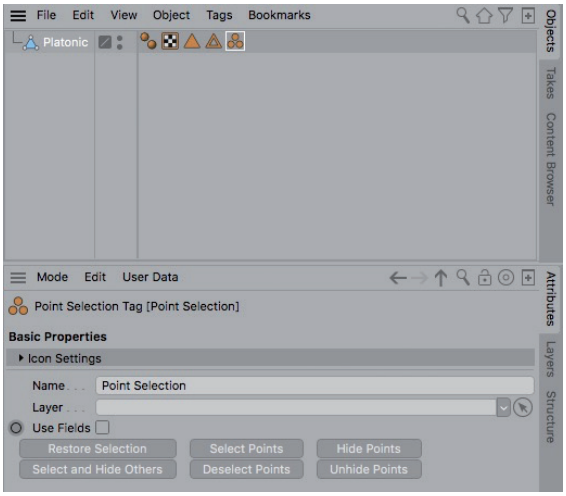
Cinema 4D manages all selections separately, which means that point, edge and polygon selections can be made independently of one another and the selection will not be lost when modes are switched.

Selections can also be transferred, e.g., a polygon selection can be turned into a point selection. In doing so, all points will be selected that are part of a selected polygon. Then simply switch to a different mode, e.g., **Use Polygon**, **Edge** or **Points** mode, while pressing **Cmd**/**Ctrl**.

This function is also available in the **Select** menu. The **Convert Selection** command lets you define which type of selection should be transferred to which mode.



As we will discuss later, selections are helpful for other things such as assigning materials or working with deformations. Selections can be saved and accesses at a later time. This function is called **Set Selection** and is also located in the **Select** menu. This command work in **Use Point**, **Edge** or **Polygon** modes and will generate a new **Selection** tag in the ***Object Manager*** next to the corresponding object.



These tags can be renamed in the ***Attribute Manager***. Double-clicking on one of these tags will show the corresponding selection. Furthermore, various options are available for these tags in the ***Attribute Manager***, which lets the corresponding selection be accessed, deleted or hidden at any time. This is especially useful when working with polygon selections. Hidden polygon selections have the same effect as if the polygons were deleted, which lets you look inside the object. However, when the object is rendered, the hidden polygons will also be rendered. Of course all hidden elements can be made visible again at any time.

The same function is also available directly in the **Select** menu, without having to save a **Selection** tag. The **Hide Selection** or **Hide Unselected** commands can be used to hide elements; the **Invert Visibility** and **Unhide All** commands can be used to unhide hidden elements.

Note that **Selection** tags can also be overwritten. If you want to set multiple selections successively, you must deselect the **Selection** tag that was created in the ***Object Manager*** before continuing. To do so, simply **Cmd**/**Ctrl** + click in an empty area next to the tag. A new **Selection** tag will be created only if no other **Selection** tag of the same type (point, edge or polygon) is selected for the object.

Briefly mention that Selection tags can in part also be created automatically by Spline Generators such as the Extrude object, for example. Selections can then be accessed even though the object has not been made editable.

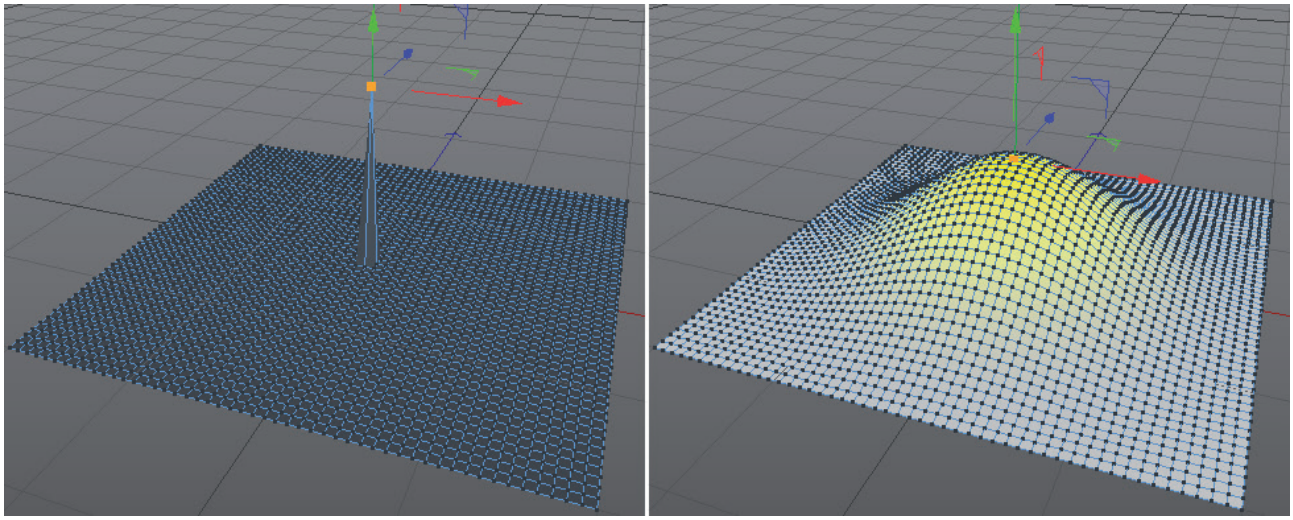
▶ *See: Exercises for making selections*

5.2.5 Soft Selections

Normally, selections do not have an intermediate state. Either an element is selected or not. The **Soft Selection** option softens the border between these states, which makes it easier to create organic shapes.

Create a **Plane** object and increase its number of segments to 50 x 50 segments before you convert it to a polygon object. Select a point at the center of the **Plane** and use the **Move** tool to drag is upwards along the **Y**-axis. An isolated tip will be created. Undo this step and select the **Move** tool again. The tool's settings will appear in the ***Attribute Manager***, including a **Soft Selection** tab. As soon as this function is enabled, the object's appearance in the Viewport will change (make sure the center point is still selected) to a gray-to-orange gradient from the selected point outward.

The **Radius** value, together with the **Falloff** function, defines the size of the soft selection. The **Mode** option defines the center point of the selection. If **Group** is selected, the mathematical center of the selection will also lie at the center of the soft selection.



The **Radius** value will define the area of influence starting from this point within the soft selection.

If **Center** is selected, the center of all selections will be used as the center of the soft selection. The differences between **Group** and **Center** will first become apparent when multiple elements at different locations on a surface are selected.

All will be the most commonly used mode because the soft selection will start at the edge of each selected element group, e.g., at the edge of a selected group of polygons on the surface. This ensures that all selected elements will always be influenced 100% and only neighboring elements will be affected by the soft selection. The intensity of the soft selection within the defined **Radius** can be adjusted by changing the **Falloff** value accordingly.

If the Rubber option is enabled, an additional lag within the soft selection can be added to elements that are affected by the movement. Moving the mouse back-and-forth very quickly can even create wave-like shapes on the **Plane**.

Enabling the **Surface** option will cause the soft selection to expand only along points, edges or polygons that are connected directly with the selected elements. If this option is disabled, all parts of the surface will be affected that lie within the radius around the actual selection.

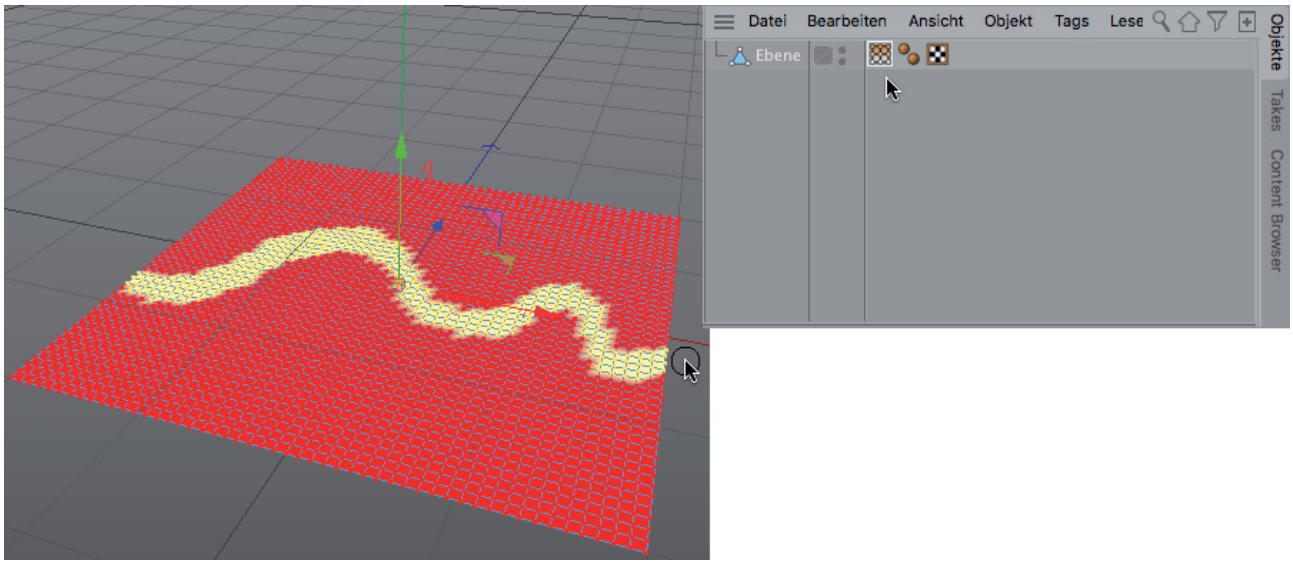
If the **Restrict** option is enabled, the soft selection will behave like a normal selection. Only the elements that are actually selected can be moved.

5.2.6 Vertex Maps

In the following, the terms ‘vertex’ and ‘points’ are used interchangeably. Both refer to the points on a given surface.

We have already discussed one possibility for making “blurry” selections to create organic transitions between selected and surrounding elements. The **Vertex Maps** function works in much the same way by assigning each point on a surface a percentage value between 0% and 100%. These Vertex Maps an then be used together with materials or deformers.

Vertex Maps can, for example, be painted on using the **Live Selection** tool.



After selecting the **Live Selection** tool you can select **Vertex Painting** from the **Mode** menu in the ***Attribute Manager***’s **Options** tab. In this mode you can define the vertex weighting (strength). The Mode options within the **Vertex Painting** menu lets you define various types of painting methods. If **Set** is selected, the percent value defined for **Strength** will be painted onto the points. The **Add** and **Subtract** modes are also available.

Clicking on a selected object with the Live Selection tool will turn the object red. Depending on the **Strength** value defined, points weighted at 100% will turn yellow. This information will be saved in a new tag next to the object in the ***Object Manager***. Clicking on this Vertex Map tag will display the vertex weighting and it can be edited.

Alternative to this mode you can select points using any selection tool and call up **Set Vertex Weight** via the **Select** menu. Here you will also find a corresponding Strength value that can be used to define point weighting. The Mode menu contains the same options for **Set**, **All** and **Subtract**.

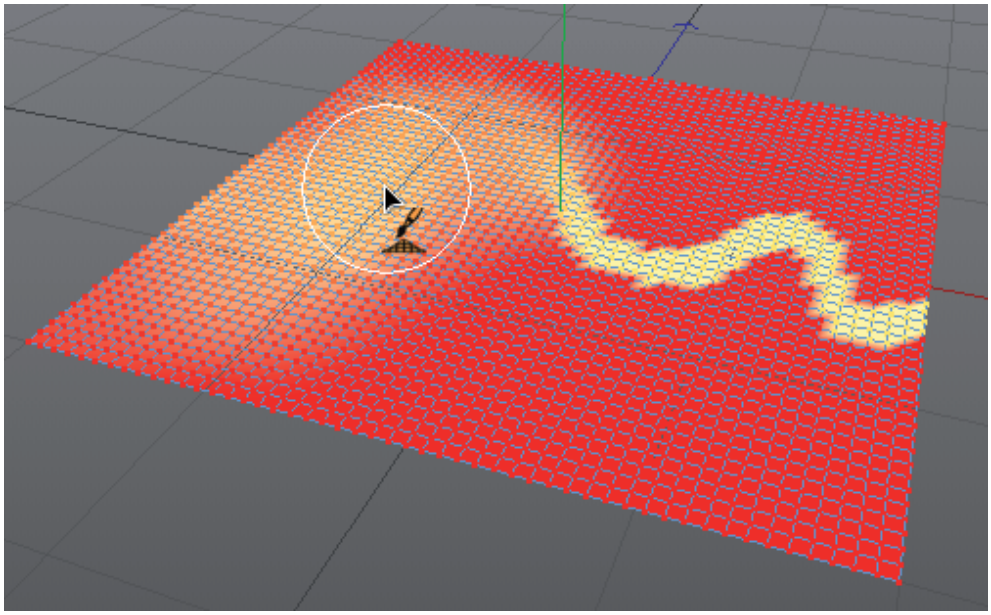
5.2.6.1 The Brush Tool

The aforementioned methods are less suited for creating vertex maps with blurry edges and transitions because they lack functions for blurring the vertex weighting or for painting with gradually decreasing intensity at the edge of the brush’s radius. These functions are offered by the **Brush** tool, which is located in the **Mesh/Transform Tools** menu. The **Brush** tool has numerous ***Attribute Manager*** settings e.g., for defining the radius or the falloff curve.

If **Mode** is set to **Paint** and the **Strength** value is used to define the desired point weighting directly beneath the cursor, weighting that gradually diminishes towards the **Radius’** edge can be painted.

Note: The **Brush** tool can be restricted to current selections, which is not suited for drawing Vertex Maps. Before you use the Brush, make sure that no elements are selected. This can be done by selecting the **Deselect All** command in the **Select** menu.

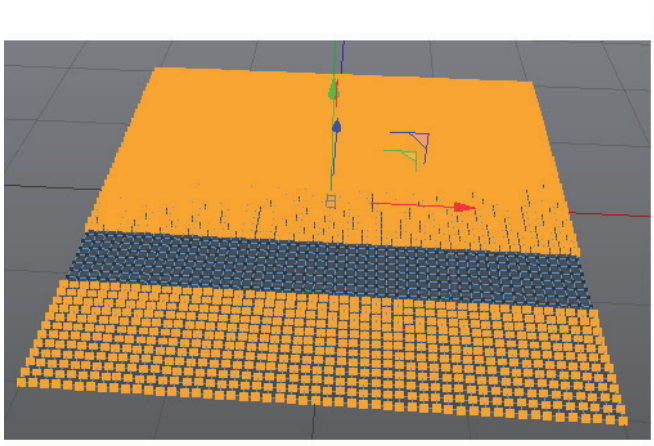
The **Blur** mode can be used to blur hard transitions between weightings. When this mode is used, **Strength** does not affect the weighting that is painted on but instead affects the intensity of the blur effect.

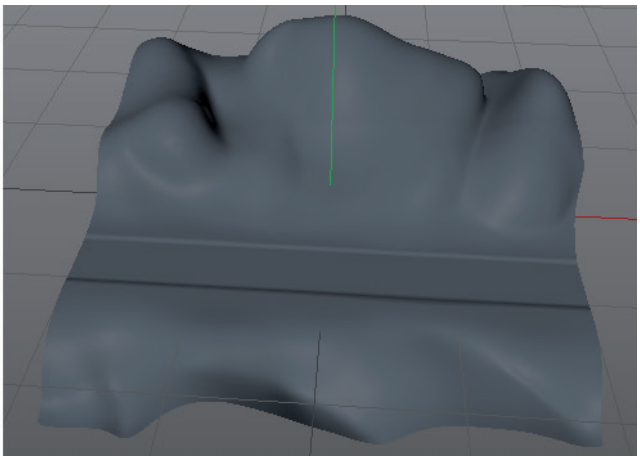


The **Brush** tool has additional modes that have nothing to do with Vertex Maps or point weighting. For example, the **Smear** mode works like a magnet or finger swipe and can be used like a soft selection for creating organic deformations on a surface.

Other modes such as **Normal** move the surface in the direction of the surface Normals, which can have an inflating effect on the shape. A helpful mode in conjunction with this is the **Smooth** mode, which can be used to smooth rounded geometry.

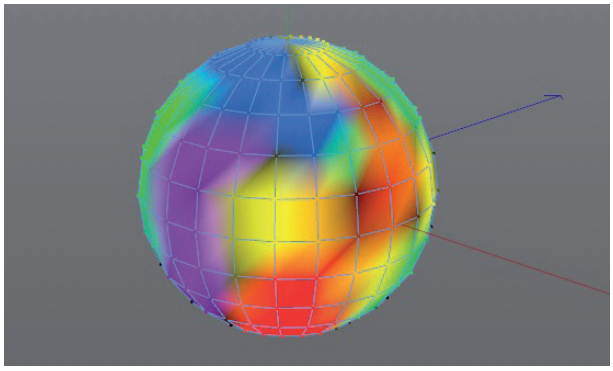
Exercise: Use the **Brush** tool to paint soft selections onto a **Plane** object to create a landscape. By using normal selections, regions for roads or rivers can be explicitly protected from the Brush tool's influence.





5.2.6.2 Vertex Colors

A secondary type of vertex map makes it possible to also save color information per vertex or polygon on a surface. To do so, assign a **Vertex Color** tag to the polygon object. This tag can be set to **Points Only** or **Polygon Points** in the **Vertex Colors** setting. The only difference between these modes is that a clear border for the applied colors along the polygon edges can be created in **Polygon Points** mode. Otherwise the colors will always transition softly, as is the case when Vertex Map weighting is applied. In both modes, the **Paint** tool can be activated by double-clicking on the **Vertex Color** tag. This tool can also be accessed via the Cinema 4D **Character** menu where size, opacity, falloff curve and of course the desired color can be defined. Furthermore, the **Paint Mode** setting lets you choose from various painting modes, including applying colors to alpha masks – or a combination. The alpha and color information can, for example, later be evaluated via a **Vertex Map** shader and used directly to design the surface for rendering. The Image shows an example of such a surface with various colors applied to it.



If the **Polygon Points** option is selected, a selection of polygons can also be colored in addition to the previously described painting of surface vertices. To do so, use the **Paint** tool's **Apply Selected** function. This will create hard edges between the colors.

If alpha masks were also applied, these can be displayed instead of the colors by enabling the **Vertex Color** tag's **Display Alpha** option. Please note that this only works if the **Paint** tool is not simultaneously active. Otherwise only the properties that are selected in the **Vertex Color** menu will be displayed.

Enabling the **Draw Points Always** option will display the surface's vertex colors even if the **Vertex Color** tag itself is not selected. The colors within the polygon cannot however be displayed

5.2.7 Selection Filter

Similar to Point, Edge and Polygon selections, selected objects or even tags can be selected automatically. This can be done using the **Select / Selection Filter … / Select** menu. This dialog window's **Objects** tab contains all object types present in the Project. Enabling the check box for a given object type will cause all objects of this type to be selected. The options in the **Tags** tab work in a similar fashion. Here, the selection of tag types can be restricted to the currently selected objects (**Restrict to Active Objects**).

The options directly below the **Select / Selection Filter …** command have the opposite effect. Only those object types selected can be selected by clicking on them. Selecting objects in the ***Object Manager*** is always possible, independent of these settings.

If you have to select the same objects again and again, the selection process can be simplified using the **Create Selection Object** command in the **Select** menu. Double clicking on this object will automatically restore the selection of all objects in the list. It works like a **Selection** tag for points, edges or polygons – but for objects.

Additional objects can be dragged into the Selection Object's **List** field in the ***Attribute Manager***. Superfluous objects can be removed by right-clicking on them and selecting the **Remove** command.

This method of saving objects is particularly helpful when working with animations. Keyframes can automatically be created for all objects in the Selection Object's list.

SUMMARY: SELECTIONS

- Basic shapes can be created using parametric Primitives and modified interactively in the Viewport using handles.
- More advanced settings and numeric values, e.g., for an object's precise dimensions, can be accessed in the *Attribute Manager.*
- Using the **Use Points**, **Edge** and **Polygon** modes is only possible for vertex or polygonal objects. Parametric Primitives can be converted to polygonal objects by pressing the **C** key. This will also cause all parametric settings to be lost.
- In order to modify a polygon object's elements, e.g., to move or delete points or polygons, they must first be selected. Numerous selection tools are available in the **Select** menu for this purpose.
- When using the standard selection tools **Live**, **Rectangle**, **Polygon** or **Lasso Selection**, always check their *Attribute Manager* settings. Particularly important is whether only visible elements should be selected or those on an object's reverse side as well.
- Additional elements can be selected by pressing the **Shift** key during selection.
- Individual elements can be deselected by pressing **Cmd**/**Ctrl** during selection.
- Double-clicking on a polygon with the **Move**, **Scale** or **Rotate** tool will execute the **Select Connected** command.
- Double-clicking on an edge with the **Move**, **Scale** or **Rotate** tool will automatically make a **Loop Selection.**
- Selections can be set in a **Selection** tag using the **Select**/**Set Selection** command. Double-clicking this tag will restore the set selection.
- Polygons can be hidden, e.g., to make it easier to modify a given object.
- Transitions to unselected elements can be added to selections by enabling Soft Selection. This option is located in the **Live Selection** tool's or the **Select** menu.
- A soft transformation or smoothing of the surface can also be done using the **Mesh** menu's **Brush** tool.
- Points can be assigned percentage values between 0% and 100% using the **Brush** or **Live Selection** tool, or the **Set Vertex Weight** command. A **Vertex Map** tag will be created.
- Objects or tags can, depending on their type, be selected via **Select**/**Selection Filter …**/**Selector.**
- A selection list under **Select**/Selection Filter … defines which object types can be selected in the Viewport.
- Multiple object selection can be saved using **Select**/**Selection Filter …**/**Create Selection Object**. Double-clicking on the **Selection** object will restore the selection.

5.3 Spline Object

Spline objects have no surfaces but serve as helper objects for modeling and animating. A spline can be used for modeling a vase or cable as well as for defining a flight path, e.g., for a camera through a building.

Splines can be compared to paths in Photoshop or Illustrator, for example, but can also be deformed in 3D space.

Splines are made up of points that are connected by a curve. This type of connection is called **interpolation**. A linear interpolation produces a simple, straight connection of spline points. A Bezier interpolation can also use tangents and therefore be used to create curved sections between points. A Spline's interpolation **Type** can be a key factor for defining a spline's shape, even if the points are located at the exact same position.

To practice working with tangents, a spline should first be created. Get accustomed to doing so in the Front view so the spline lies on the XY plane. This makes it easier to edit. A side or the Top view can be used when creating splines that, for example, will be used as a path or for such objects as cables or tubes. The Perspective view should be avoided for creating splines because it offers very little control over the placement of spline points.

5.3.1 Drawing a Spline

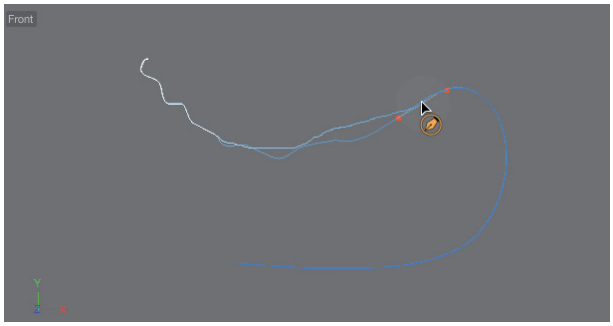
There are three ways of drawing splines:

1. By selecting **Spline / Sketch**
2. By selecting **Spline / Pen**
3. By selecting **Spline / Spline Arc Tool**

These options are also available as icons in the Cinema 4D interface.

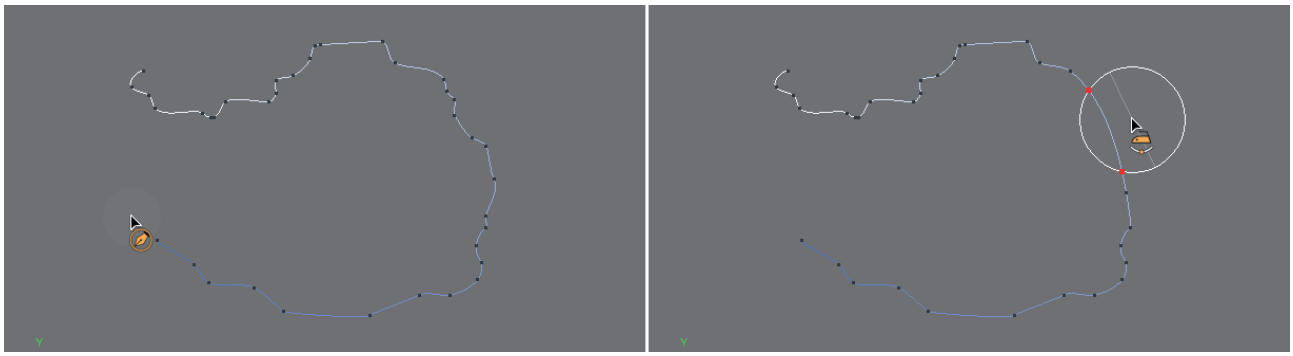
The **Sketch** tool is applied by clicking and dragging with the mouse. This can, for example, be used in conjunction with a graphics tablet when tracing a shape. The other tools create splines point-by-point and are therefore better suited for creating very precise splines.

The **Sketch** tool can be used to correct spline sections by painting over them to replace them.



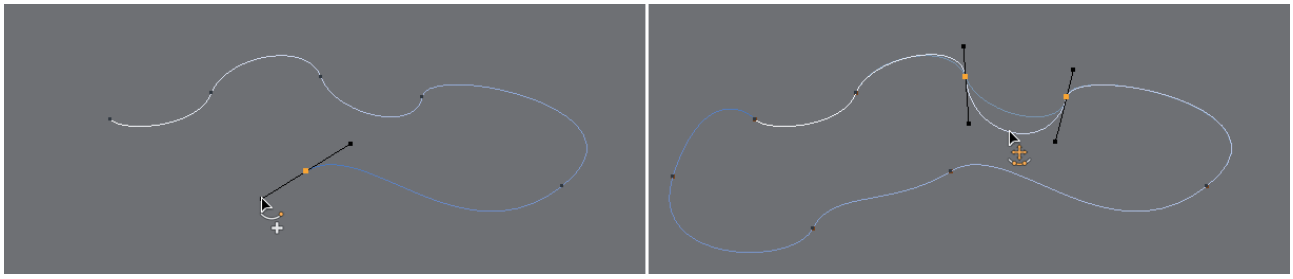
A **Radius** value and a **Blend** setting can be used to define the tool's range.

If the **Shift** key is pressed during application, the **Spline Smooth** tool will be added automatically, which lets a spline be smoothed, twirled or crumpled.



The **Spline Pen** tool can be used to build up splines point-by-point. Existing splines can be continued from the last selected point. This can be interrupted at any time by pressing the Esc key and resumed by clicking on a spline end point. When creating a new spline point and the **Spline Pen** tool is set to **Beziér**, the LMB can be pressed and held and dragged to create a tangent for this point.

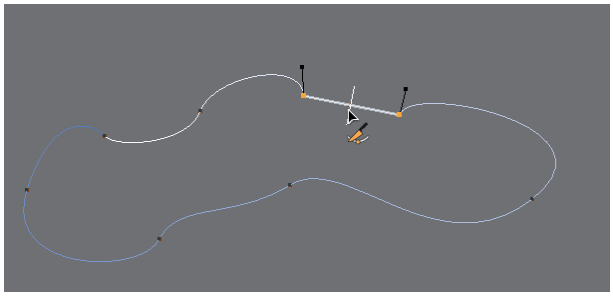
The spline segments between existing points can be grabbed and moved.



Settings in the ***Attribute Manager*** can be used to define the length or angle of the respective tangents.

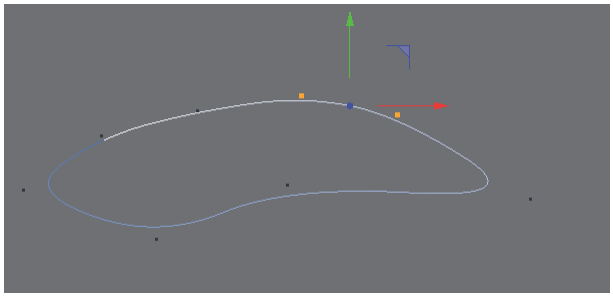
Spline points can be repositioned using the mouse during creation. To edit the tangents of existing points, the respective spline point must first be selected.

Double-clicking on a spline segment between two points or on a point will make these linear.



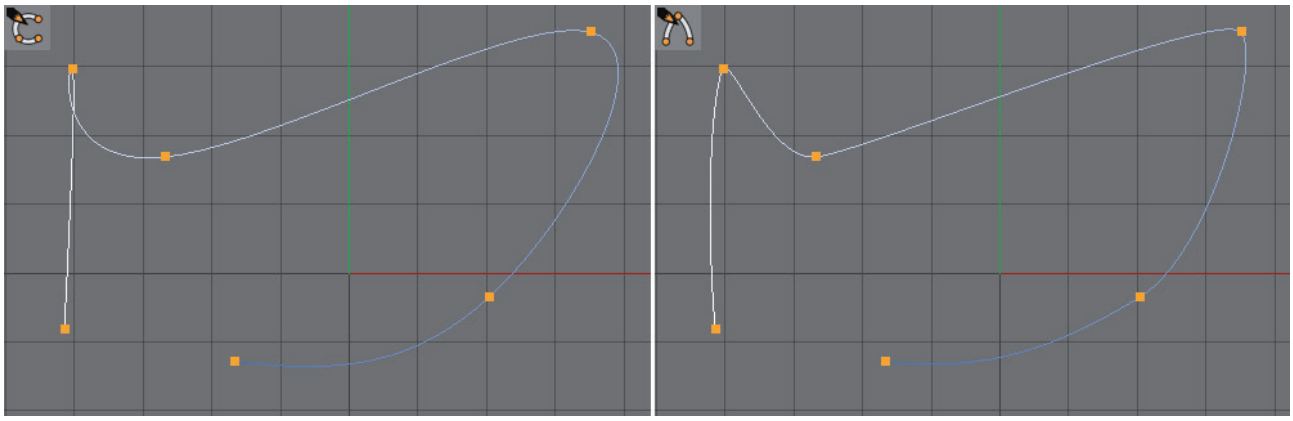
A subsequent double-click will restore a soft interpolation for that segment or point. **Cmd/Ctrl**+click on the spline will set a new point without affecting the spline's shape.

If the other end of the spline is clicked upon during creation, the spline will be closed. Alternatively, the **Close Spline** option can be used to close a selected spline. This option is located in the spline object's ***Attribute Manager*** settings. The spline **Type** can also be changed at any time and subsequently modified. You should therefore note that a spline's shape can be changed by switching its type. This is especially true for the **B-Spline** whose spline does not necessarily run through the points set for it.

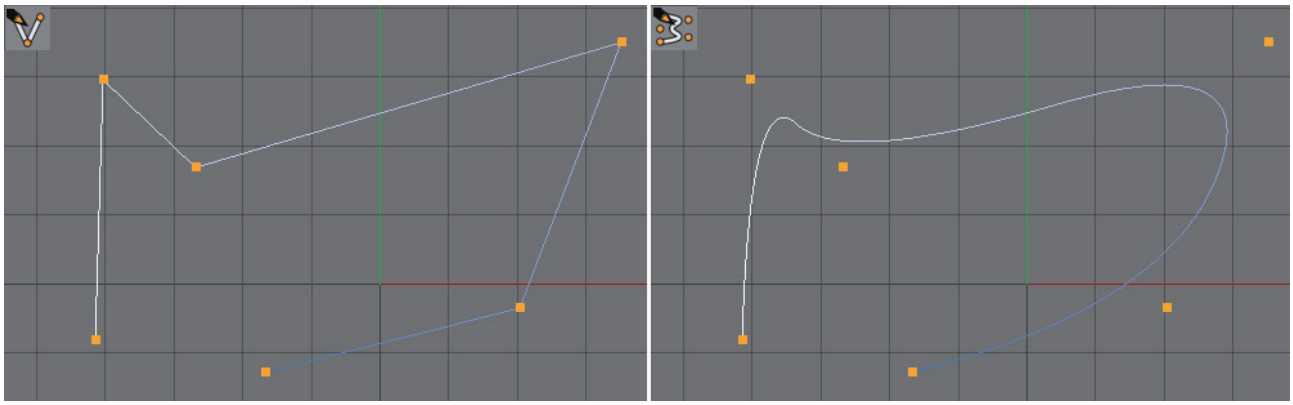


Generally speaking, if the Esc key was pressed to interrupt the creation of a spline using the **Spline Pen**, common settings can also be accessed via a context menu that will be displayed by pressing and holding the LMB on a spline segment. This function is specially designed for graphic tablet or track pad users who do not have the option of using the RMB.

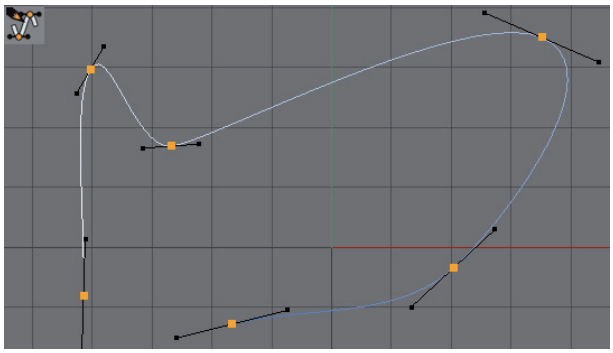
The interpolation types **Cubic** and **Akima** create splines whose curves have different strengths between points.



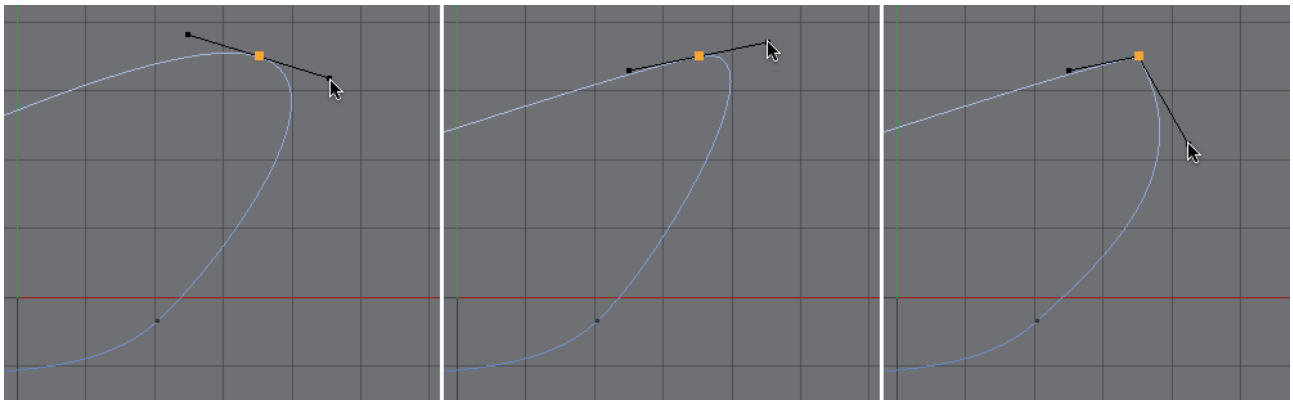
The **Linear** interpolation creates straight lines between points. The B-Spline interpolation interprets straight connections between neighboring points as tangents. Therefore, the spline that is created is very curved and does not necessarily run through a given point.



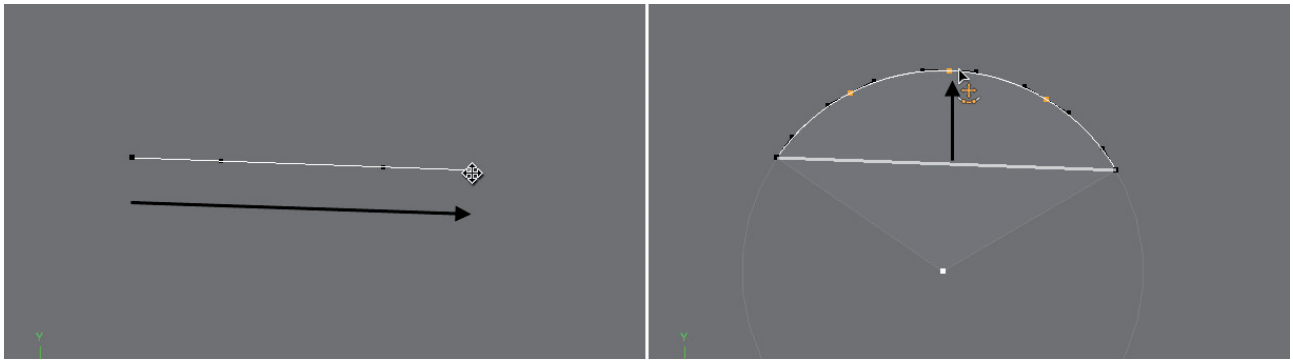
The **Bézier** spline is the only interpolation method that also adds tangents to the spline points, which makes it the most precise method of controlling the shape of curves between points.



If the **Shift** key is pressed during scaling or rotation, the selected tangent can be moved independently of the other tangent. This is called **breaking the tangent**. This way irregularly shaped curves and sharp bends can be created along the spline.



The **Spline Arc Tool** can be used to create precise arc segments. To do so, click and drag a line. After the mouse button is released, start and end points of an arc will be created. The **Esc** key can be pressed to interrupt the process and you can click and drag the segment between the points to create an arc.



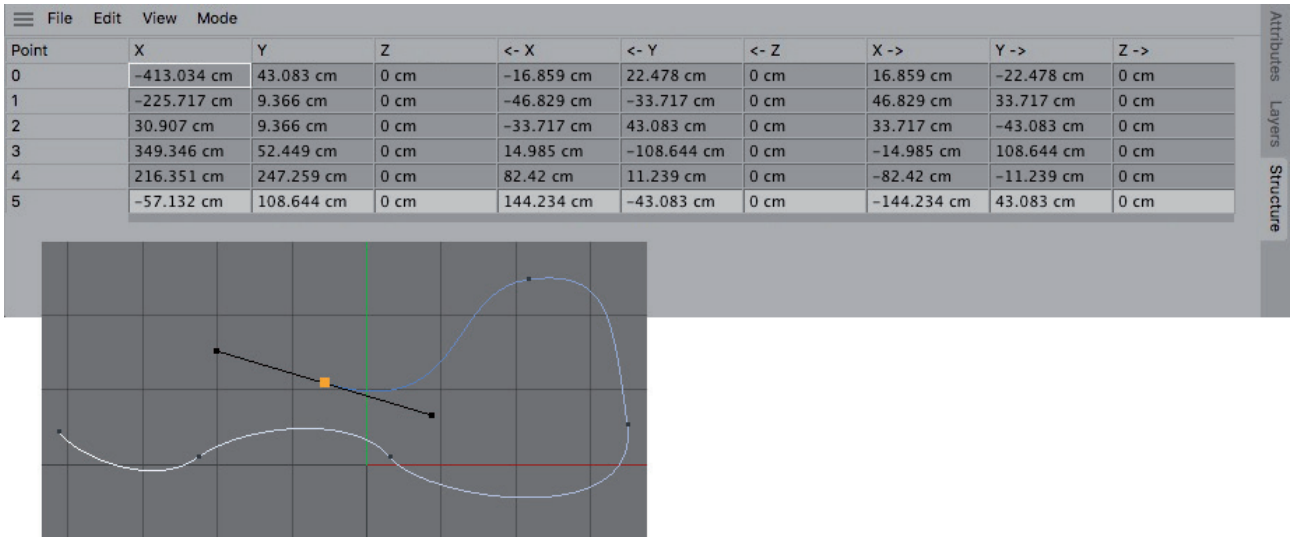
The center of the circle of curvature, its end points, the arc itself or the connecting lines from the center of the curvature to the end points can be clicked upon and moved. After the arc has been created, the shape can be continued by clicking on one of its end points with the **Spline Pen** or **Spline Arc Tool**.

5.3.2 A Spline's Inner Structure

Splines are, like converted polygon objects, made up of points and can therefore be edited in **Use Point** mode directly in the Viewport, e.g., using the **Move**, **Scale** and **Rotate** tools. The order in which a spline's points are arranged is very important. Cinema 4D uses a simple color gradient along the spline to clearly show the direction in which the spline flows. If a spline is selected and **Use Point** mode is activated, the white end of the spline marks its starting point and the dark blue end its end point.

This point order must be noted when splines are used for modeling, e.g., in conjunction with the **Loft** or **Sweep** object. The order in which points are arranged is also important for animation because it defines, for example, the end at which a camera path begins.

An overview of a spline's points, their order and position as well as an overview of all tangents can be found in the ***Structure Manager***.



Point	X	Y	Z	<- X	<- Y	<- Z	X ->	Y ->	Z ->
0	-413.034 cm	43.083 cm	0 cm	-16.859 cm	22.478 cm	0 cm	16.859 cm	-22.478 cm	0 cm
1	-225.717 cm	9.366 cm	0 cm	-46.829 cm	-33.717 cm	0 cm	46.829 cm	33.717 cm	0 cm
2	30.907 cm	9.366 cm	0 cm	-33.717 cm	43.083 cm	0 cm	33.717 cm	-43.083 cm	0 cm
3	349.346 cm	52.449 cm	0 cm	14.985 cm	-108.644 cm	0 cm	-14.985 cm	108.644 cm	0 cm
4	216.351 cm	247.259 cm	0 cm	82.42 cm	11.239 cm	0 cm	-82.42 cm	-11.239 cm	0 cm
5	-57.132 cm	108.644 cm	0 cm	144.234 cm	-43.083 cm	0 cm	-144.234 cm	43.083 cm	0 cm

5.3.2.1 The Structure Manager

The ***Structure Manager*** is located in the standard Cinema 4D layout next to the ***Attribute Manager***. Click on the **Structure** tab to show its contents. The ***Structure Manager*** offers several modes, e.g., for displaying polygons, point weighting, UVW coordinates or an object's points. If a spline is selected, the point numbers will be displayed in the left column in increasing order. The sequence will generally begin with 0.

The neighboring columns **X**, **Y** and **Z** show the point coordinates in the spline's local coordinate system. Generally speaking, the coordinates are listed relative to the spline's coordinate system.

If a **Bézier** spline is selected, additional columns will follow in which the length and direction of the left and right tangent arm on each point are shown. These coordinates are also saved locally, relative to the point coordinates. Values of 0 for **<-X**, **<-Y** and **<-Z** mean that the left tangent arm for this point has no length. If the spline was drawn in the Front view, a tangent can very easily be made perfectly horizontal by entering 0 for **<-Y** and **Y->**. If **<-X** and **X->** are both set to 0, a vertical tangent will be created.

All point position and tangent values can be modified manually by double-clicking on the corresponding value field in the ***Structure Manager***.

The point order can be re-arranged by dragging and dropping the point numbers in the far left column. However, this can be done more easily by using the context menu in the Viewport.

5.3.2.2 Special Spline Functions

If a spline is selected in **Use Point** mode, a context menu can be opened by simply right-clicking in any Viewport. This menu contains various commands for editing splines and their tangents.

If a spline point is selected, the **Set First Point** command can be used to move the point to the top of the list in the ***Structure Manager***. This point will now serve as the spline's start point and the remaining points will be arranged accordingly. If the spline is not closed, the gap between the first and last points will be repositioned, which can change the spline's shape.

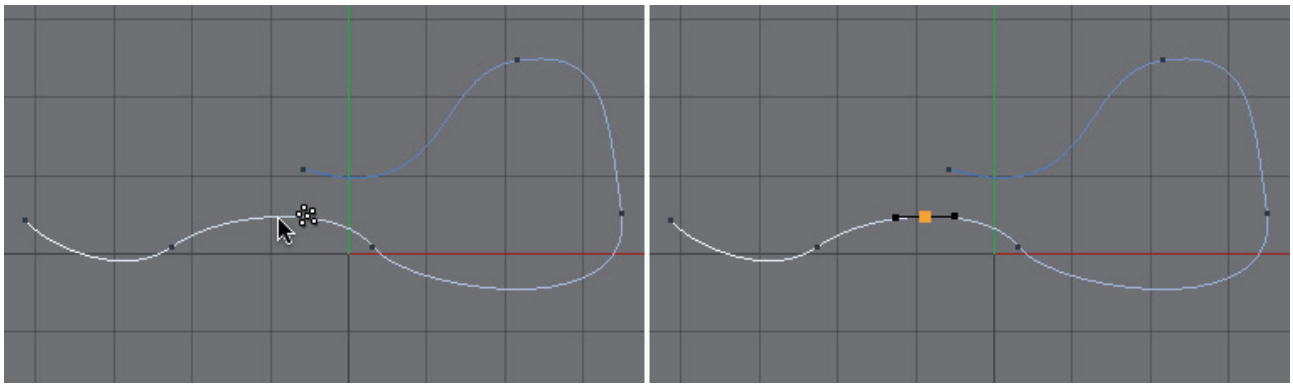
Selecting the **Reverse Sequence** command will invert the sequence of points and therewith the direction in which the spline flows. The spline's shape will remain unchanged.

Move Down Sequence and **Move Up Sequence**, respectively, moves the order of the points down or up one position in the ***Structure Manager***. If the spline is open, the gap between the first and last points will be moved.

The following commands are only relevant when working with **Bézier** splines with tangents:

Hard Interpolation:	The tangents of the selected points will be set to 0
Soft Interpolation:	The tangents of the selected points will be configured to create harmoniously-shaped curves. Broken tangents will automatically be restored to a consistent line and scaled symmetrically.
Equal Tangent Length:	The left and right tangent arms of the selected points will be scaled equally.
Equal Tangent Direction:	Broken tangents of selected points will be restored to a uniform direction.

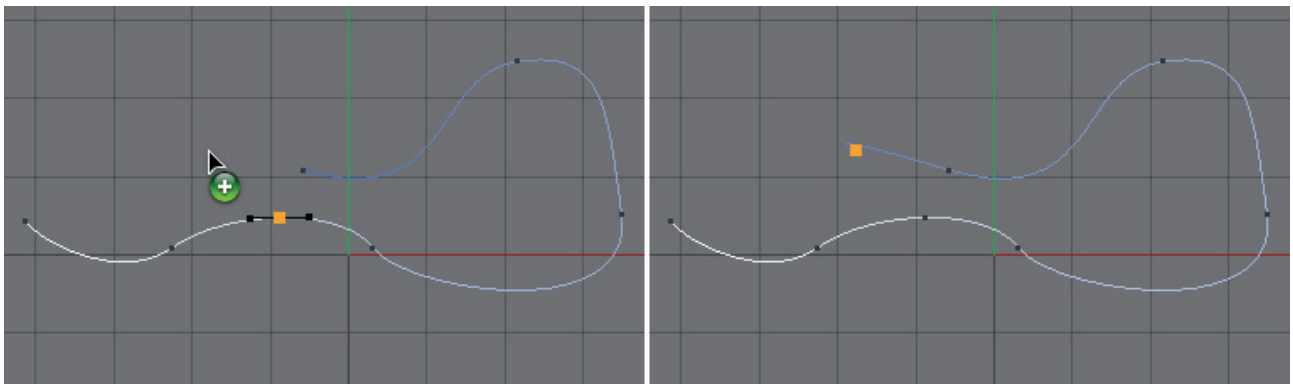
If a point was omitted when a spline was created it can be easily added by **Ctrl**/**Cmd** + clicking on the spline with the **Move** tool.



A point will be created at this location on the spline. Alternatively you can use the **Create Point** command from the context menu and simply click on the spline without using any hotkeys.

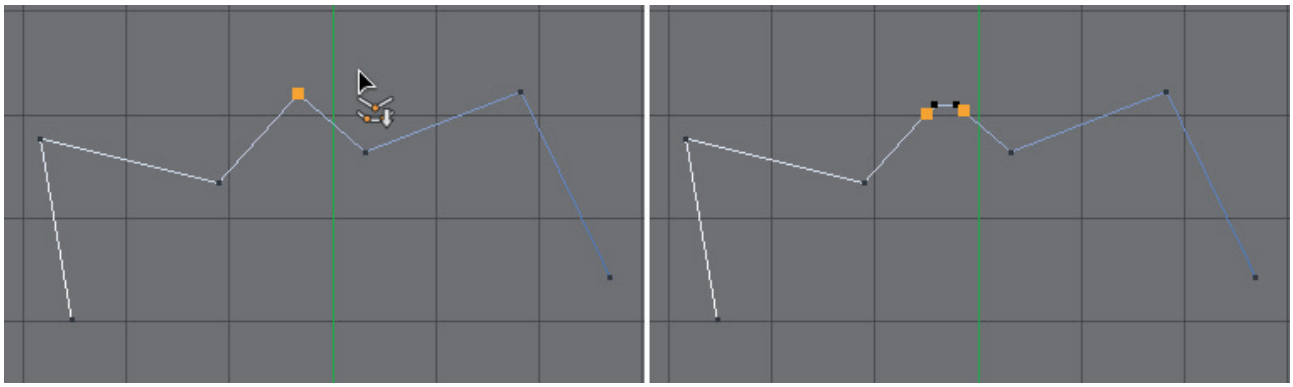
If the spline is **Linear** or **Bézier**, the original spline shape will be maintained. For other interpolation types, the curve's shape will change when a point is added.

If the cursor is located too far away from the curve when it is **Cmd**/**Ctrl** + clicked upon, a green plus symbol will be displayed. Clicking will create a new point that will automatically be connected to the spline's end point.



If the new point should be connected to the beginning of the spline, the direction of the spline must be inverted before clicking. Creating a new point outside of the existing spline curve can also be done using the **Create Point** command in combination with pressing the **Cmd**/**Ctrl** key.

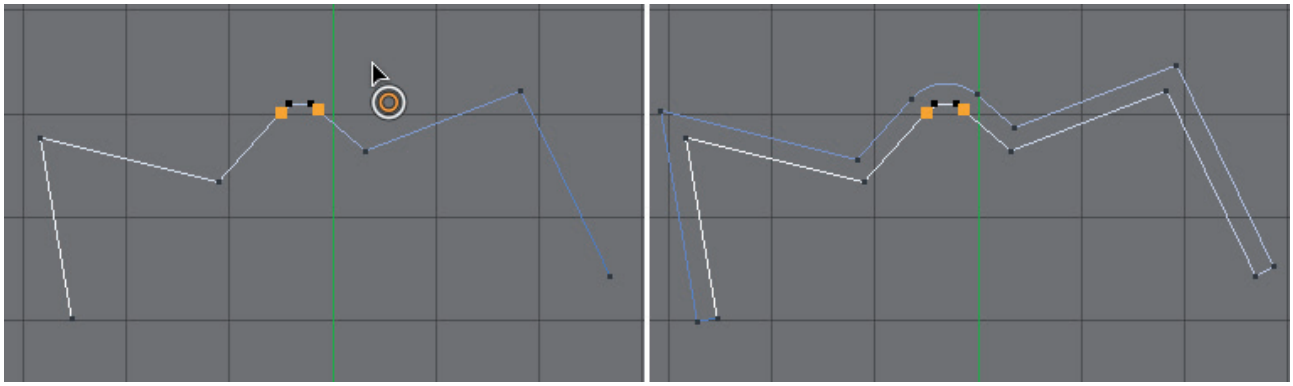
When a soft interpolation is applied to a point with a hard interpolation, the shape of the spline to the neighboring point will change. If the spline's curvature at a given point should be restricted, select the point, right-click in the Viewport and select the **Chamfer** command from the context menu. Click and drag next to the selected point. The point will be split into two new points whose tangents create a curve.



The curve's **Radius** can be modified in the ***Attribute Manager***. The **Flat** option can be enabled to create a straight chamfer for the previous point.

Important! When using the tool interactively, note that the Chamfer command is executed each time the mouse is clicked. Briefly releasing the mouse button and again clicking and dragging will result in multiple splitting of the spline point.

Using the **Create Outline** command is identical to using the **Chamfer** tool, only that no points must be selected before it is applied. The **Create Outline** command always affects the entire spline and creates an additional curve that lies for the most part parallel to the original curve.

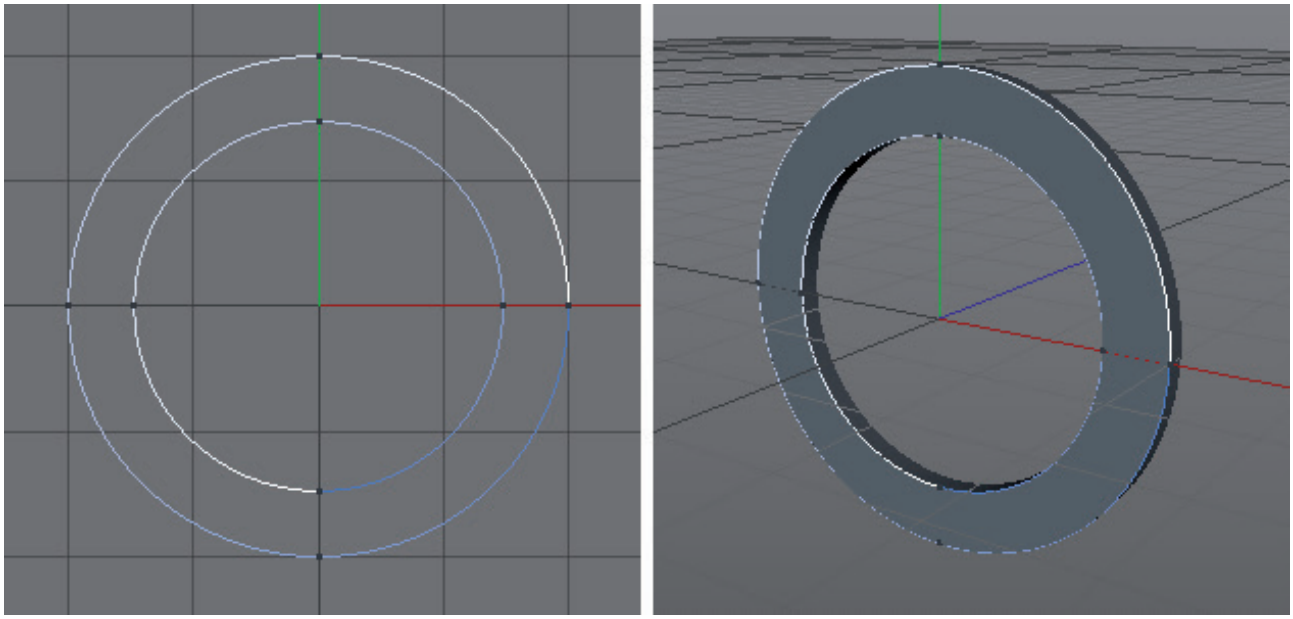


A new spline object will not be created. Splines can be made up of several separate **Segments**.

5.3.2.3 Spline Segments

Segments are like separate curves with in a spline object. They can be recognized by their own white-to-blue color gradient in the Viewport. Closed splines automatically also contain closed segments. Mixing closed and open curves within a spline object is not possible.

Closed segments that lie completely within another closed segment curve will automatically be seen as an opening or gap by Cinema 4D during modeling. A typical example are letters such as O that are made up of a circular outer line and a circle at the center. The inner circle will automatically be treated as an opening.



Segments can also be created by connecting different spline objects. For example, draw several curves as separate spline objects (enabled **Create New Spline** option) and select them in the *Object Manager*. In the main **Mesh** menu the **Connect Objects** and **Connect Objects + Delete** will now be available on the **Conversions** sub-menu.

The **Connect Objects** command will create a new object that contains all selected splines as segments. The original splines will remain unchanged.

The **Connect Objects + Delete** option has the exact same function but will delete the original splines from the scene.

A spline's segments can be subsequently connected or new segments can be created using a spline curve's points. The commands in the context menu that appears when you right-click in the Viewport can be used for this. Alternatively you can use the **Mesh / Spline** menu. If you select one spline's end point and another's start point, both segments can be combined by using the **Join Segment** command. Points selected along a spline can be separated from the spline using the **Break Segment** command. If you want to create a new spline object using the selected points, use the **Disconnect …** option. The original spline shape will not be changed.

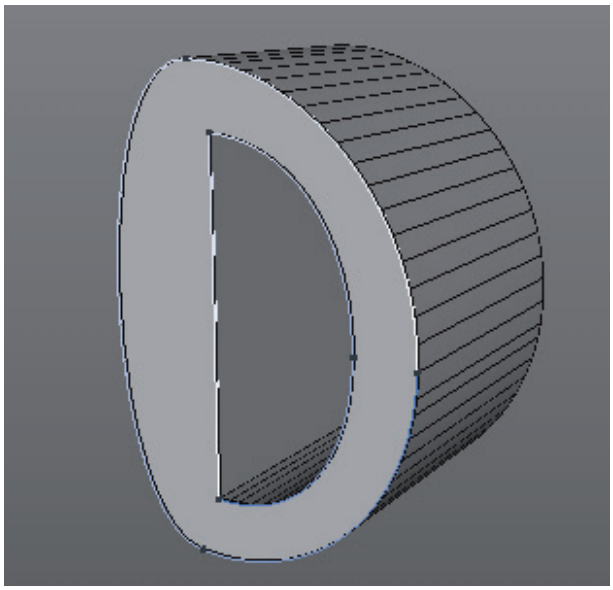
5.3.2.4 Intermediate Points

Only at first glance does it look like a spline's curve runs precisely through its points. At second glance you will notice that even the curves seem to be made up of small straight sections. This is because splines already contain all information regarding how they help to create polygons and thus visible shapes. Hence, the spline's straight sections represent the edges of polygons that can be created with the splines.

As with the rounding of the cube, the precision of the rounded edge increases as the number of segments increases, which also increases the number of polygons. This indirectly reduces the edge's size and the polygons are no longer noticeable.

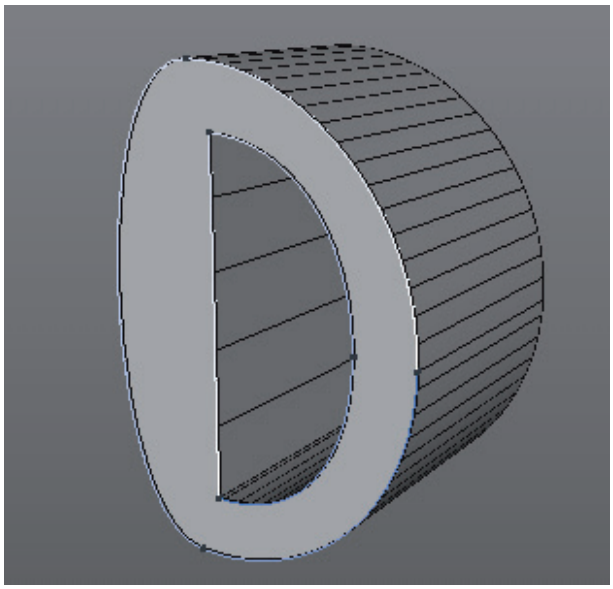
These types of segments are called **intermediate points** and they can be modified in the ***Attribute Manager***. Various algorithms are available for placing intermediate points that help determine a good ratio between a precise shape and an acceptable number of segments.

The default mode is **Adaptive**, which only uses intermediate points wherever the spline is curved. This setting is also affected by the **Angle** value.



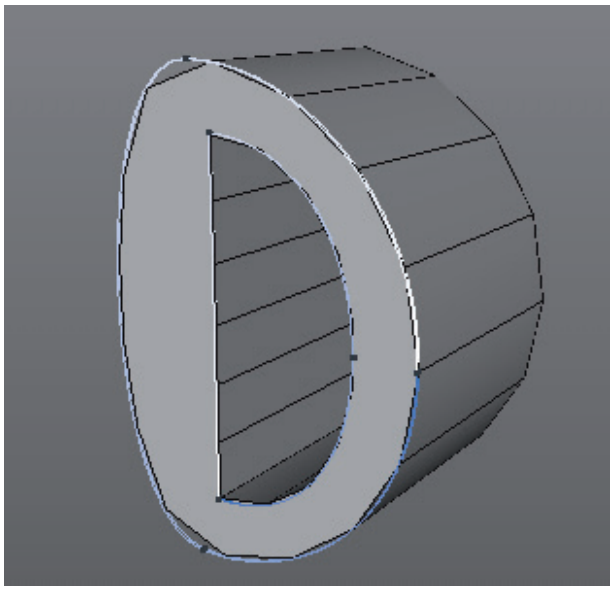
Whenever the spline's curvature exceeds this angle, a new segment or intermediate point will be added. This mode is very useful because it restricts the number of segments for sections that require higher subdivision.

If the spline's shape only changes slightly, these details can easily get lost. In such instances it can help to lower the Angle value. However, this will in turn increase the number of segments in regions that already have a high enough subdivision. It would then be better to simply switch to the **Subdivided** mode.



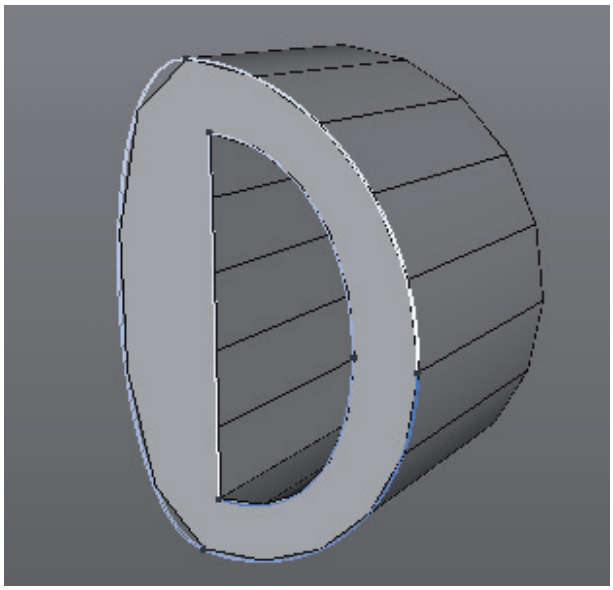
The **Maximum Length** setting will be available, which subdivides curves independent of their shape but relative to their length. As soon as a section is longer than the defined **Maximum Length** value an intermediate point will automatically be added, even on linear sections.

The **Uniform** mode can also be interesting in many cases.



A **Number** setting will be made available, which can be used to indirectly define the total number of segments for a given spline. The segments will all have a uniform length and will be dispersed uniformly along the entire length of the spline. The actual number of segments is based on the number of spline points -1. This value is used if **Number** is set to 0. If **Number** is set to 1, the number of segments will be doubled; if **Number** is set to 2 the number of segments will be tripled and so on.

The **Natural** mode uses the same mechanism for determining the number of segments.



However, sections of different lengths will be created. Each section between neighboring points will be assigned the same number of segments.

5.3.3 Spline Primitives

Fortunately we don't have to create all spline shapes manually – we can use the predefined spline Primitives. These can be found in the **Create**/**Spline** menu or in the top icon palette. You will find the most commonly used shapes such as **Circle** or **Rectangle** as well as more complex shapes like **Text** or **Cog Wheel**.

Similar to the parametric Primitives, the ***Attribute Manager***'s settings can also be used to adjust the size and shape of spline Primitives. The only difference is that no handles will be made available in the Viewport.

Spline Primitives can also be converted to a normal spline object by pressing the **C** key or selecting **Mesh**/**Conversion**/**Make Editable** from the main menu. This object can then be edited using the previously described spline tools.

SUMMARY: SPLINE OBJECTS

- Spline objects are helper objects for modeling and animation.
- Splines can be sketched directly in the Viewport. However, a more precise creation of splines can be achieved using the point-by-point method using the **Spline Pen** or the **Spline Arc Tool**. The interpolation type determines how the spline will run through the points.
- With only a few exceptions, splines should always be created in the Front view.
- Only the **Bézier** spline lets spline curves be adjusted using tangents.
- A tangent can be broken by clicking on a tangent arm while pressing **Shift**.
- Additional points can be added to a spline or at the end of a spline by **Cmd**/**Ctrl** + clicking with the **Move** tool.
- Splines can be closed or opened at any time by enabling the **Close Spline** option in the ***Attribute Manager***.
- The **Spline Smooth** tool can be used to deform the spline or to reduce detail.
- The order of a spline's points is listed in the ***Structure Manager*** and also displayed using a white-to-blue gradient in the Viewport.
- Items in the ***Structure Manager***'s list can be rearranged by dragging & dropping the point numbers.
- Position and tangent values in the ***Structure Manager*** can be edited directly by double-clicking on them.
- Special commands for editing splines can be found in the **Spline** menu or by right-clicking in the Viewport in **Use Point** mode.
- Splines can consist of independent segments connected to make up a single spline object.
- Closed segments that lie within a larger segment shape will be interpreted as openings during modeling.
- Segments can be created by connecting individual splines, e.g., using the **Mesh**/**Conversions**/**Connect Objects + Delete** command.
- A curve's precision is defined by intermediate points, which are not visible. Their arrangement and number can be defined in the ***Attribute Manager***. This creates the foundation for modeling using splines and in many cases determines the size and number of polygons that will be created.
- Many common shapes, and in particular curves, are available as presets and can be configured using the ***Attribute Manager's*** settings.
- Parametric splines can be converted to spline objects, which makes it easier to modify them using special tools.

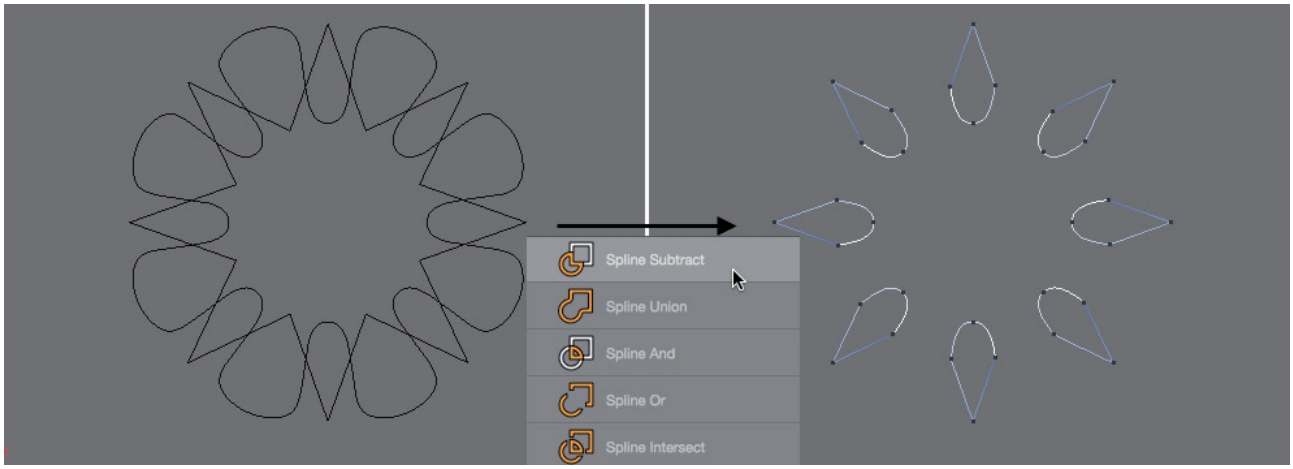
5.3.4 Modeling with Splines

Splines are not designed to model all shapes because only four types of objects can be used to generate geometry in conjunction with splines. However, these offer very helpful procedures with which rotational symmetrical objects, cables, tubes and pipes, logos and 3D text can be created. The working principle is always the same. The spline object must be made a subordinate object of the respective **Generator** object in the *Object Manager*. Several of these **Generator** objects require or allow multiple spline objects to be used simultaneously. In these instances, the order of the spline objects in the hierarchy is important.

5.3.4.1 Combining Splines

Multiple splines can be combined with one another as a preliminary stage for modeling polygons using splines, which makes it possible to create even more complex shapes or paths. To combine splines, you must first select at least 2 overlapping splines that lie on the same plane. Various functions are available in the **Spline/Boole Commands** menu where you will find the functions **Spline Subtract, Spline Union, Spline And, Spline Or** and **Spline Intersect**. For some of these options, the order in which the splines are arranged in the *Object Manager* can be important. For example, different shapes can result when splines are subtracted, depending which spline is subtracted from which. Two colors are used to display selected objects in the *Object Manager*: the object selected last will have a slightly lighter color. These colors can be defined with the **Preferences** menu's **Scheme Colors/Interface Colors/Object Manager – Active Selection** option if you want to make the difference even more apparent.

After selecting one of the aforementioned functions, the original splines will be replaced by a new Spline object, which is the result of the respective operation. The **Spline Or** and **Spline Intersect** operations in particular can generate a high number of segments. Select **Spline/Segments/Explode Segments** to create individual new segments. The spline functions for combining multiple splines work with both custom splines and parametric Spline objects alike (see SplineBool.tif).



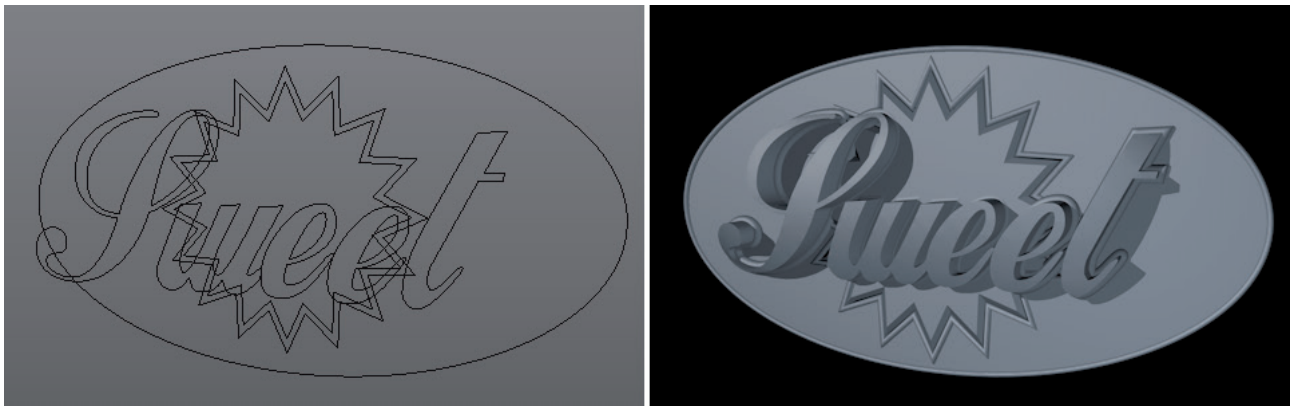
A comparable function is the **Spline Mask** object in the **Create/Modeling** menu. The original splines will be maintained and can be replaced, animated or edited at any time. We will discuss this later in conjunction with the Modeling objects.

5.3.4.2 The Extrude Object

The Extrude object is located in the **Create**/**Generators** menu.

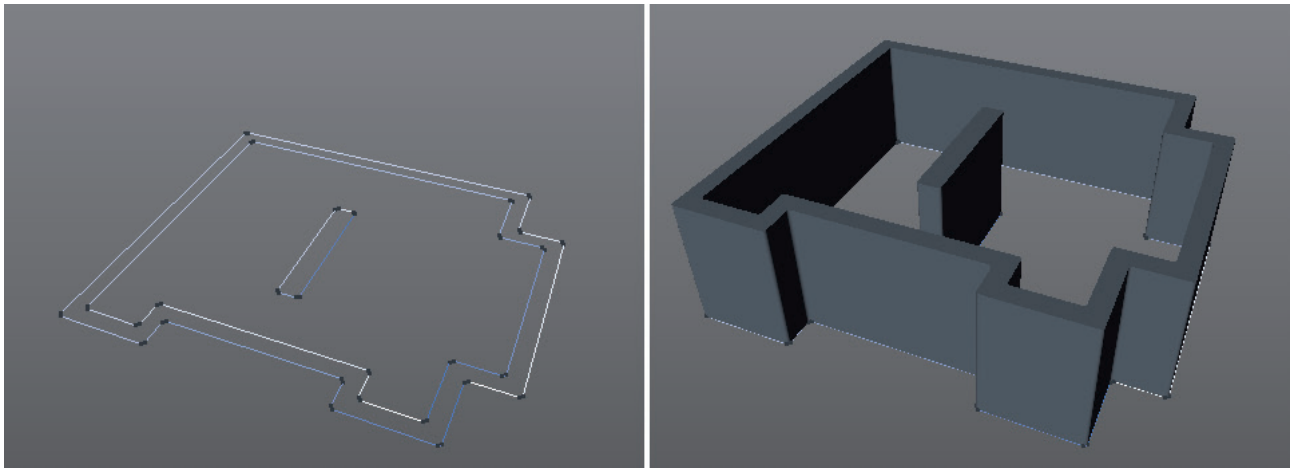
The spline will be moved relative to the **Extrude** object's coordinate system. A surface will be created between the original spline and the moved (extruded) spline. The **X**, **Y** and **Z** distance that the spline is moved can be defined in the ***Attribute Manager*** using the **Movement** values.

This effect can be used to create 3D text or logos, or, for example, for tracing a floor plan and extruding it upwards. This method can be used to create entire stories.



The number and arrangement of polygons along the spline curve is defined using the spline object's **Intermediate Points** setting. Additional subdivisions along the extrusion can be defined using the **Extrude** object's **Subdivision** setting.

The **Extrude** object can simultaneously extrude multiple subordinate spline objects.



The order in which the splines are arranged in the hierarchy is not important. To activate this mode, enable the **Hierarchical** option in the **Extrude** object's settings. Otherwise only the top-most spline in the hierarchy will be extruded.

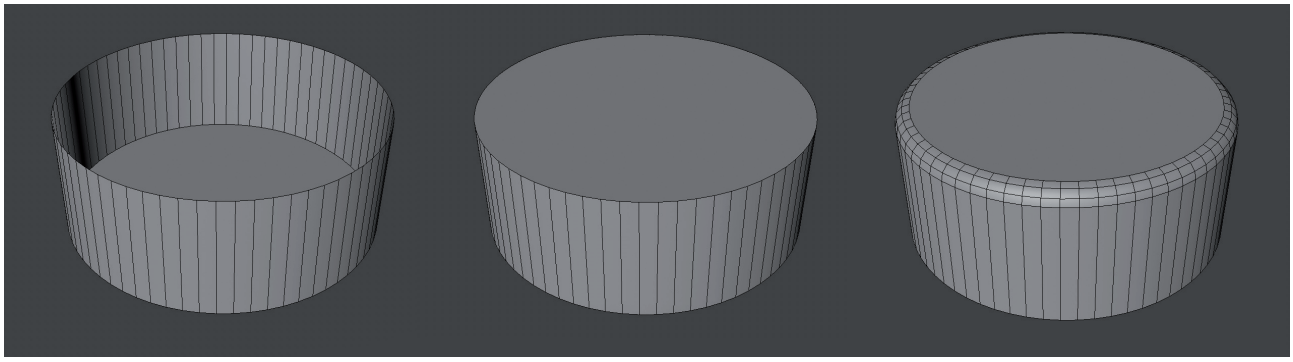
Enabling the **Hierarchical** option will also affect the coordinate system for the extrusions. The **Extrude** object's coordinate system will no longer be used. Instead, the coordinate system of each subordinate spline will be used. The extrusion will also take effect even if the splines are rotated. For best results, the extruded splines should be two-dimensional. Otherwise problems can occur when the Caps surfaces are calculated.

5.3.4.2.1 Caps Surfaces

Caps surfaces can only be created for closed splines. They close the area within the spline and create a solid surface. The ***Attribute Manager's*** **Caps** tab's settings let you define whether or not Caps surfaces should even be created and if so, how they should be created for the selected **Extrude** object.

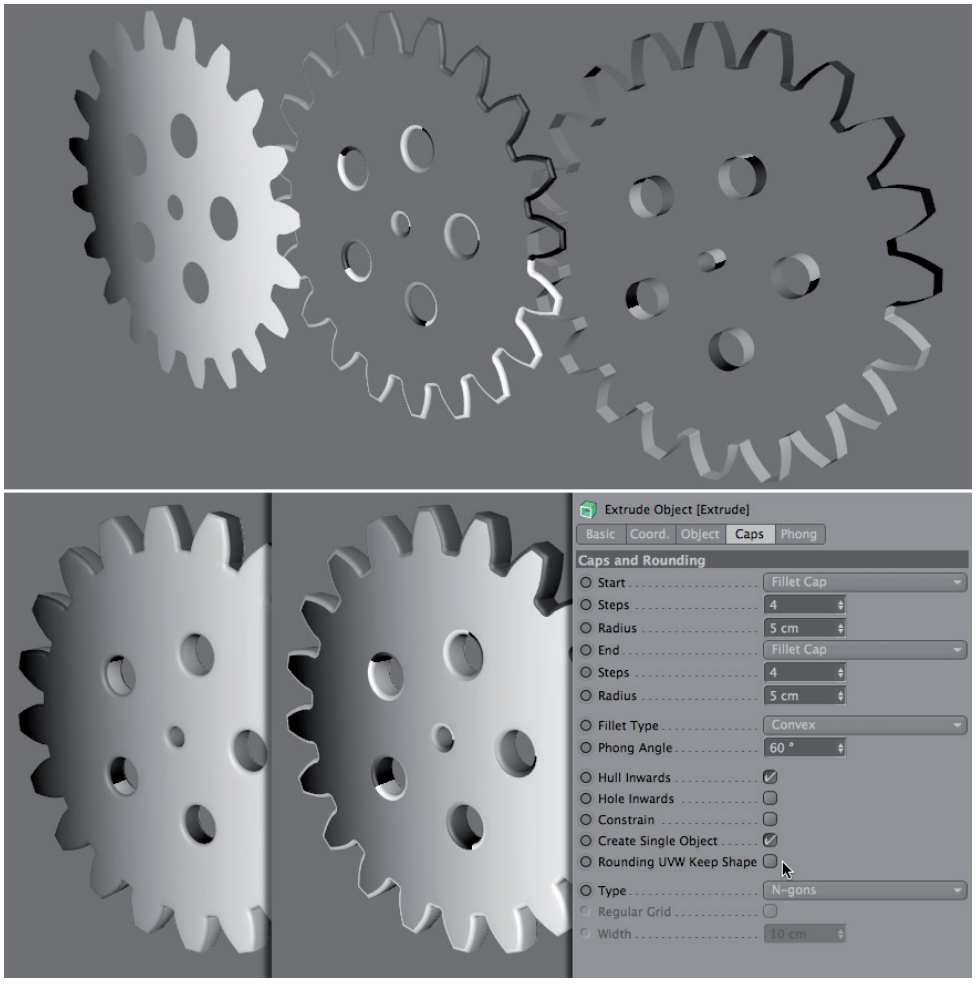
This menu contains individual settings for **Start** and **End** parameters, if **Separate Bevel Controls** is enabled, which refer to the front and backside of the shape, respectively. The **Start** is always the side on which the original spline lies; **End** is the region at which the **extrusion** of the spline ends.

Thy type of fillet can be defined using the **Shape** setting. The width of the fillet is defined using the **Size** setting. If the **Self-Intersection** option is **enabled**, only those fillet radii will be used that fit within the existing spline shape. Otherwise unwanted overlapping and intersecting of surfaces can occur.



Creating polygons for the Caps surfaces can be controlled using the Caps **Type** setting. The default setting is **N-gons**, which is generally a good choice in order to keep the number of surfaces created to a minimum. However, if the object should be deformed, a **Uniform Subdivision** can make more sense. Its grid size cn be defined using a separate **scale** setting.

Subsequent assignment of deformations or materials is made easier by the automatically generated selections, which can be selected in the **Selections** tab. Pre-defined **edge** and **polygon selections** are available. Take a brief look at the options for closing caps surfaces.



5.3.4.2.1.1 N-gons

N-gons are special types of polygons. Whereas normal polygons have three or four corner points, N-gons can theoretically have any number of corner points. N-gons automatically determine how to best connect to form three-sided or four-sided polygons. Even if corner points are deleted or added, the polygons within the N-gon that are by default not visible to us will be re-arranged automatically.

Because in the end N-gons also only have three or four corners and we can't control their arrangement, N-gons only work dependably on two-dimensional planes. This is one of the reasons why the splines used should be created two-dimensionally.

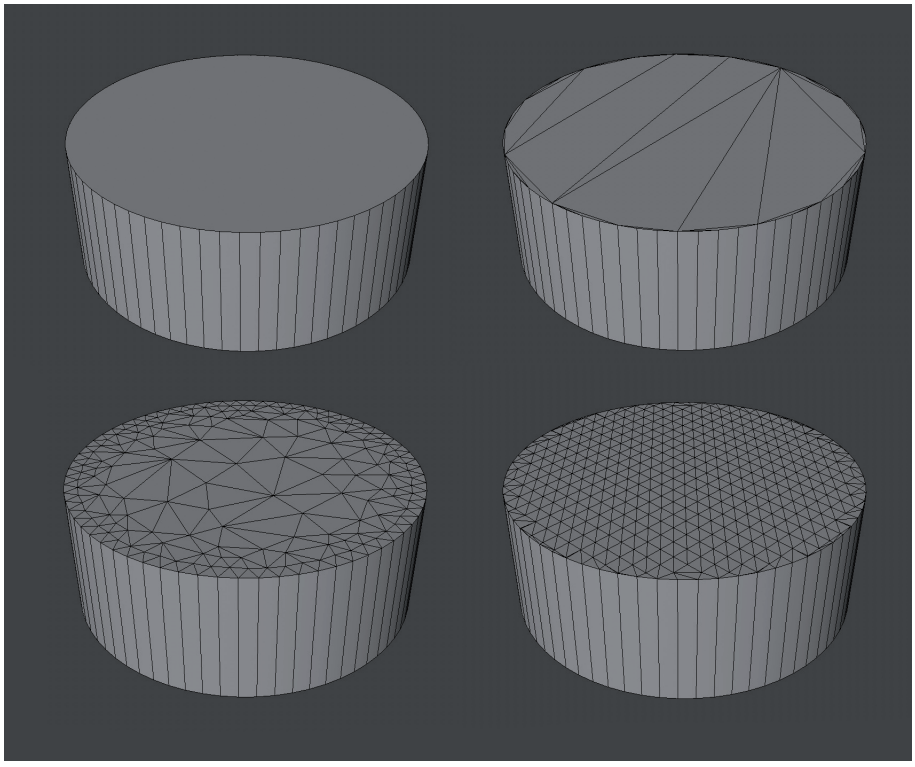
N-gons are useful for subsequent modeling because they can help close openings surrounded by surfaces with a single click or to subsequently reduce the number of an object's polygons.

5.3.4.2.1.2 Caps Surfaces made up of Triangles or Quads

If **Type** is set to **Triangles** or **Quads**, only triangles or quads will be used to close the cap surfaces. It can also happen that triangles are used for the **Quads** option if the number of corner points at an edge makes this necessary.

However, the basic issue for these modes is that the Caps surface must be as even, i.e., as two-dimensional as possible.

If the object and its Caps surfaces will be deformed, there is a special mode available to improve the flexibility of the Caps surfaces. The **Regular Grid** option for Caps must be enabled.



Additional points will be created within the Caps surfaces in relation to the **Size** value defined, which are connected via a uniform grid. Edge regions can nevertheless have chaotic connections of triangles and quads. These regions can only be affected by adjusting the corresponding spline's **Intermediate Points** value.

If the object should not be deformed, using the Regular Grid option is generally not necessary and will only serve to increase the total number of surfaces and memory required.

5.3.4.2.1.3 Delaunay Subdivision

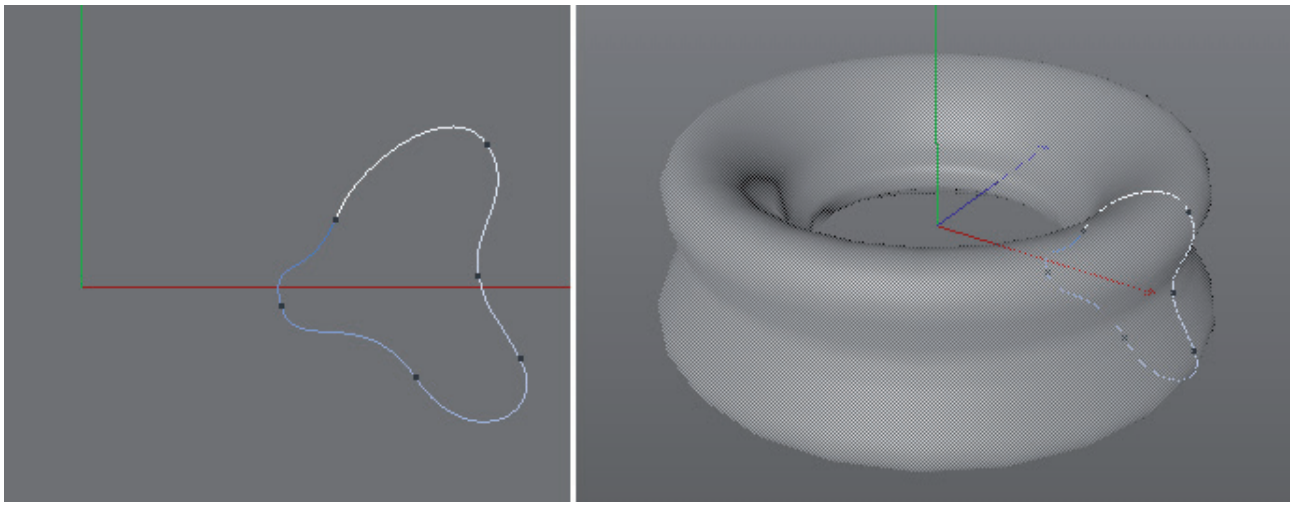
Triangles and quads are also used here to subdivide the entire caps surface into smaller surfaces, which can make deformations or subsequent modeling steps easier to master. A defined surface pattern is not created, rather an irregular arrangement of polygons that can look like a spider's web or shattered glass. The **Density slider** can be used

to individual adjust the number of polygons. However, this only affects the regions at the center of the caps surfaces. The surface density at the edge regions of the caps surface is only defined by the intermediate points of the respective spline.

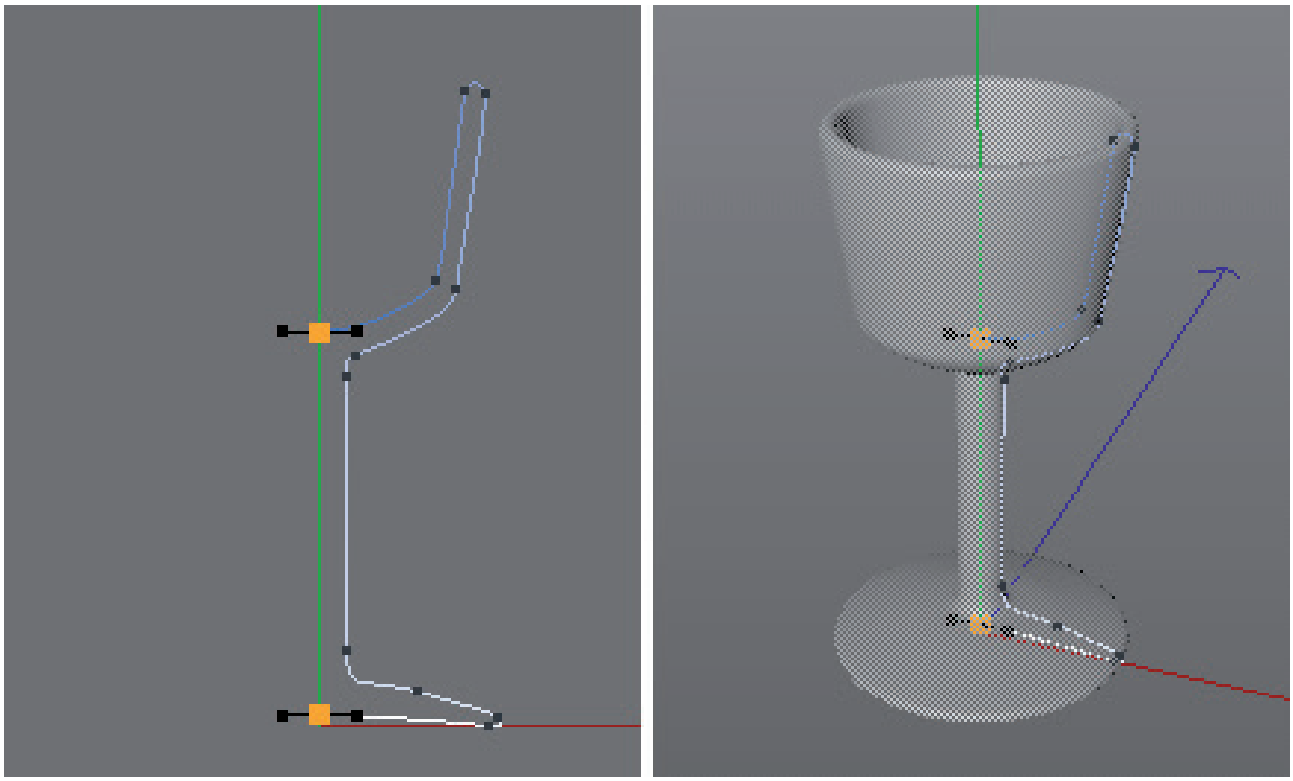
The settings already described for Caps surfaces on closed splines will be repeated in the descriptions below for modeling objects for splines. Therefore, these descriptions will not be repeated.

5.3.4.3 Lathe Object

This object's icon already reveals the purpose for which it's designed: creating rotationally symmetric objects. Imagine it as a type of potter's wheel that lies on the **Lathe** object's XY plane. The wheel's rotational axis is the same as that of the **Lathe** object, the Y-axis. If the spline is drawn in the Front view it already has the correct orientation for use with the Lathe object.



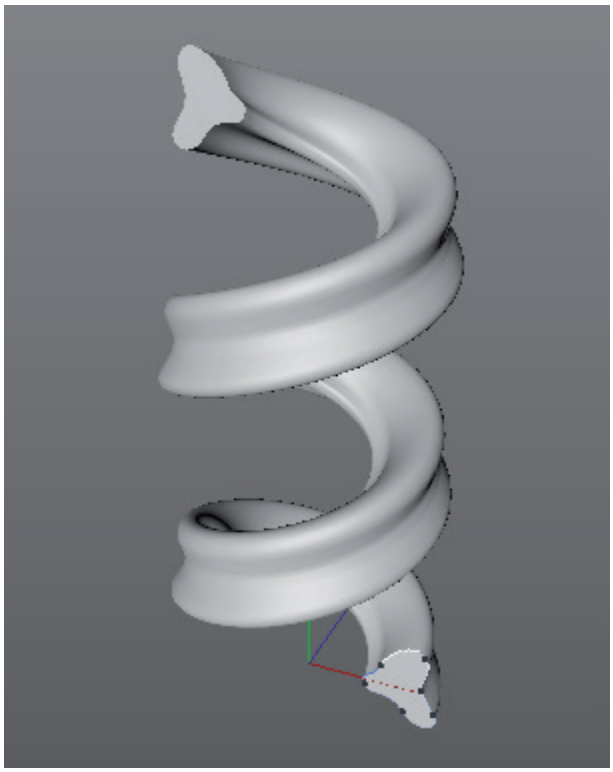
Solid objects can be created even if a closed spline is not used. The spline's first and last points must both lie exactly on the same Y axis as the Lathe object. This method is, for example, used to create drinking glasses, vases or bottles. Only have of the object's profile must be drawn using the spline – the other half will be created automatically when the **Lathe** object rotates it around the Y axis.



If the spline lies entirely outside of the Lathe object's Y axis, ring or tire shapes can be created that are open at the center.

The quality of the shape is determined by the number of and arrangement of **Intermediate Points** along the Y axis on the subordinate spline object. The **Lathe** object's **Subdivision** setting is used to define the number of segments of the rotated object. The higher this value is, the smoother the surface will appear. The rotation does not necessarily have to be a complete 360°. The **Angle** value can be adjusted to any value less than 360°.

Using an **Angle** value greater than 360° may seem senseless but this opens up new possibilities for creating unique shapes in conjunction with the **Movement** value. Changing this value will move the spline in the Y direction along the **Lathe** object. A helix or corkscrew effect will result.

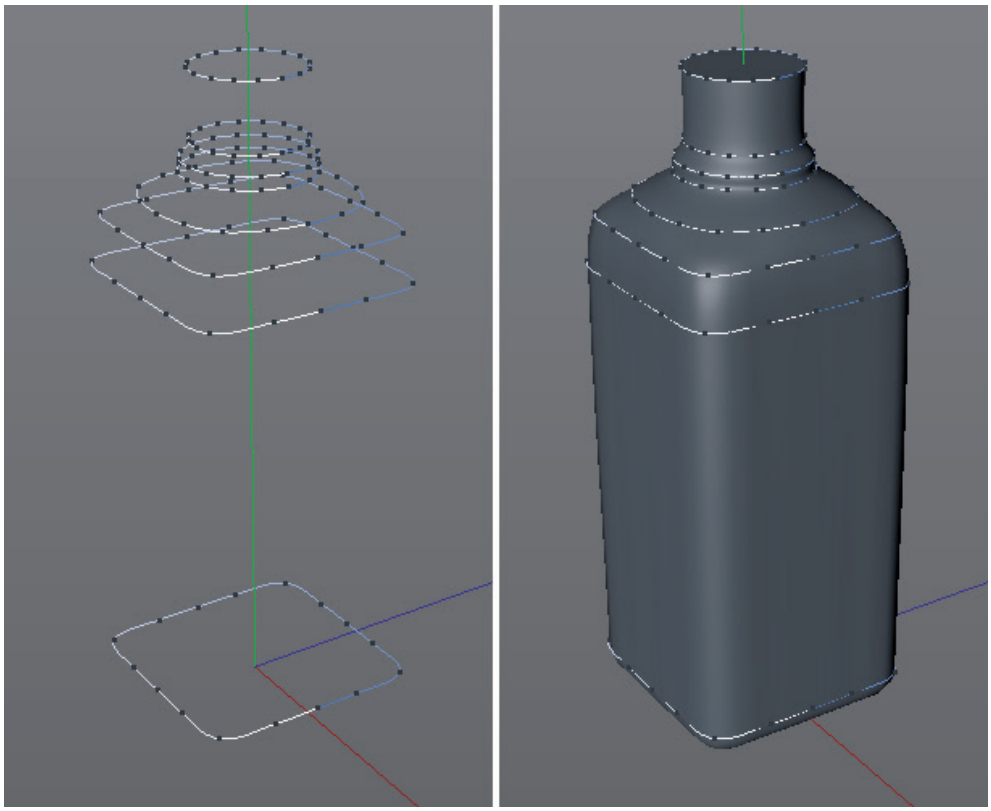


In addition, the spline can be made larger or smaller during rotation by adjusting the **Scaling** value, which, for example, makes it possible to create a snail shell shape.

The **Lathe** object can generally only be used in conjunction with a spline. If several splines are made subordinate objects of the **Lathe** object, only the top-most spline in the hierarchy would be used.

5.3.4.4 The Loft Object

This object is a little different from the rest because it can only be used in conjunction with two or more splines. This object is able to span a surface between subordinate splines in the order in which they are listed in the hierarchy. This is very useful, for example, if you know what an object's cross-section looks like. These can be traced using splines and connected with a surface using the **Loft** object.



Any number of splines can be used and they should all lie on the same plane, if possible, even if they're placed at different locations in 3D space.

The **Loft** object does not take into consideration the number intermediate points along the splines it connects. It uses its own parameters to determine the number of circumferential segments and the number of subdivisions between the splines.

Mesh Subdivision U defines the subdivision along the spline profile and **Mesh Subdivision V** defines the subdivision of the segments between splines. Other options can be used to control the creation of geometry.

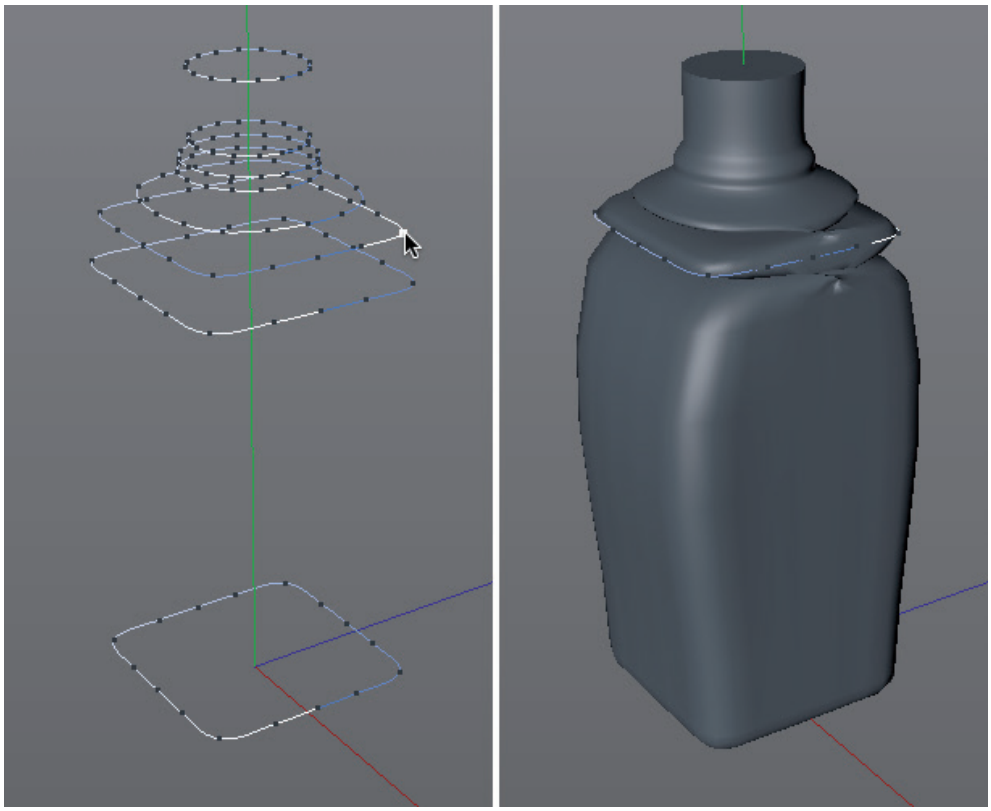
Linear Interpolation only takes into consideration the straight connection to the next spline. The result is angled transitions between splines and a correspondingly mechanical-looking shape. If this option is not enabled, the position of the splines in relation to one another will be taken into consideration and a soft, flowing shape will be generated through all splines.

Enabling the **Loop** option will connect the first and last splines and create a closed shape. The distance between these splines should be minimal because no splines exist between them to affect the shape of the surface.

If **Subdivision per Segment** is enabled, the value defined for **Mesh Subdivision V** will be applied to two neighboring splines. If this option is disabled, the entire shape will be affected by the **Mesh Subdivision V** value only. However, the segment sizes will be uniform and not dependent on the distance between splines.

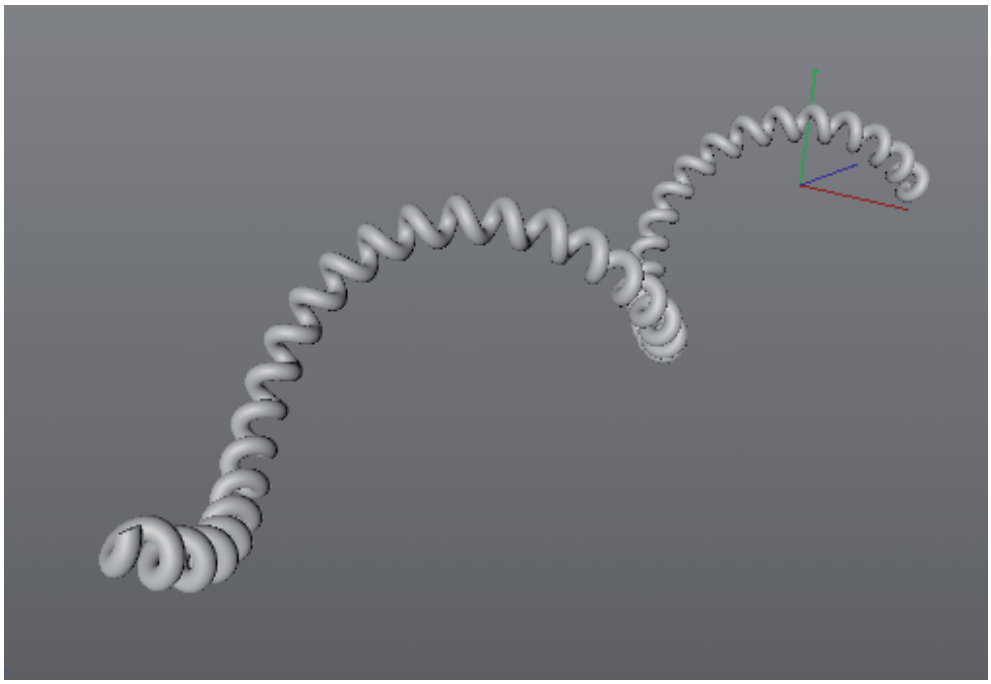
The **Organic Form** option only affects the distance between segments in the U direction, i.e., in the direction of the curves. If this option is disabled, the segments will run exactly through the spline's points. If enabled, the available number of segments will be arranged uniformly along the spline. This can cause details to be lost on the spline but the transition to the next spline will be softer and more organic.

Note that all splines must have the same orientation, i.e., their points must, for example, all run either clockwise or counterclockwise. Furthermore, each spline point must lie on a line if at all possible to avoid twisted shapes from being calculated.



5.3.4.5 The Sweep Object

This object also requires more than one spline – a profile spline and a path spline – and is primarily used to create tube or cable-like structures. One spline, the **Profile**, defines the cross-section of a tube or cable. The other spline, the **Path**, defines the course of the tube or cable. The profile spline must be created in the Front view because the **Sweep** object expects it to be created on the XY plane! The path spline can be positioned anywhere in 3D space.



As subordinate objects to the **Sweep** object, these splines must be placed in the correct order with the profile spline at the top of the hierarchy and the path spline below.

Optionally, a third spline can be used, which should be placed at the bottom of the hierarchy. This spline can be used to control the size and rotation of the profile along the path spline. The **Sweep** object also offers options that can be used in some instances instead of adding a third line.

Let's take a look at the **Sweep** object's most important settings:

The **End Scale** setting works like the **Lathe** object's scaling and defines the profile's size at the and of the path. The **End Rotation** setting works similarly and defines the rotation of the profile around its Z axis over the course of the path. The level of detail for the profile's rotation is directly dependent on the number and density of intermediate points along the path spline.

If a rotation needs to be applied, the **Uniform** option in conjunction with a high **Number** value for intermediate points is recommended for the splines.

Start Growth and **End Growth** define the section of path spline that should be used by the **Sweep** object in percent. If **Start Growth** is set to 50% and **End Growth** to 100%, only the last half of the path spline will be covered by the profile. These settings can be very useful for animations, e.g., for letting a blade of grass grow or for squeezing out toothpaste.

A brief look a the most important options for animation:

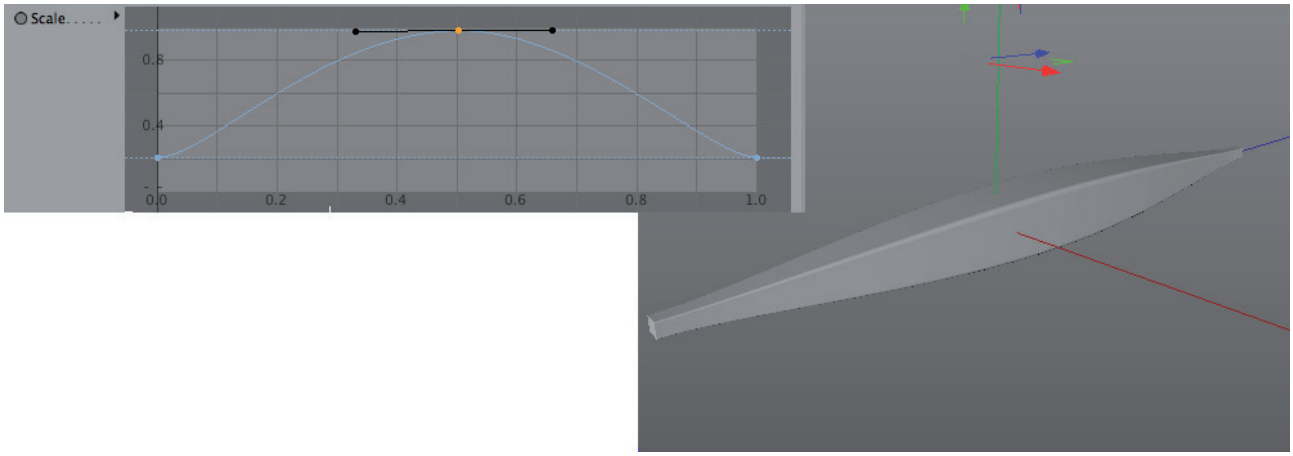
Enabling the **Parallel Movement** option will maintain the profile's original orientation while it's moved across the path. **Banking** affects the profile's automatic rotation around the axis along the path. The profile leans into the curves. A profile's banking can be controlled even more precisely using a **Rail** spline or a **Rotation** curve.

The **Constant Cross Section** option automatically scales the cross-section if hard bends or kinks occur along the path. The **Sweep** object's cross-section will remain more uniform. The following options are only relevant if a third spline is used as a virtual rail – also called a **Rail** spline. Such a third spline is often a copy of the path spline and runs more or less parallel to the path. The path spline's size, rotation or both can be modified by adjusting the rail spline's distance from the path and its orientation.

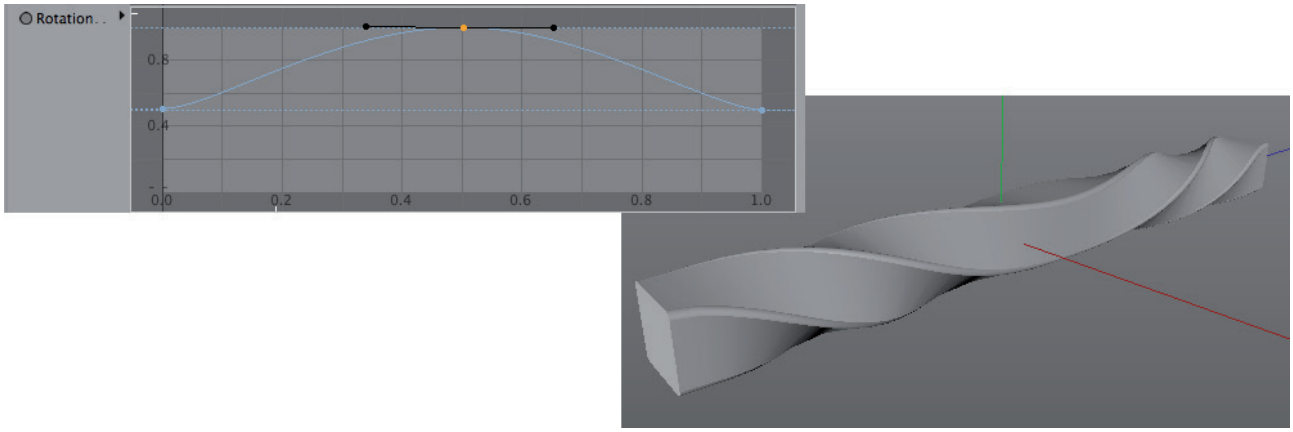
If only **Use Rail Direction** is enabled, the profile's angle will orient itself between the path and rail. If only Use Rail Scale is enabled, the profile's size will be based on the gap between path and rail. Both options can also be combined.

If the **2-Rail** option is also enabled, the profile will always be centered between path and rail. Otherwise it will always stay on the path, which results in a larger shape.

The profile can be scaled or rotated along the path using the two function curves in the **Object** tab's **Details** menu in the *Attribute Manager*. These function curves represent the path's shape from left to right. The **Scale** curve's height represents the profile's size. If the curve lies entirely at the top edge of the graph, the profile's size will be 100% from start to end.



The **Rotation** curve's height automatically oscillates between both angles defined in the **From** and **To** settings at the bottom of the window. If the curve runs horizontally at a height of 0.5, its angle will lie exactly between the **From** and **To** values.



Next to each curve's name you will see a small black triangle. Clicking on the triangle will make additional options available.

The function curves themselves can be modified by clicking and dragging on their blue lines. Points can be added by **Cmd**/**Ctrl** + clicking on the curve. To remove a curve, click on it and press the **Delete** or **Backspace** key on your keyboard.

▶ *See: Exercises for spline modeling objects*

SUMMARY: GENERATORS

- Multiple splines can be combined, subtracted or combined to create a union of splines. The order in which splines are selected in the ***Object Manager*** can affect the result; the object selected last will be displayed in a different color in the ***Object Manager***.
- The **Extrude**, **Lathe**, **Loft** and **Sweep** objects can use splines to create geometry. The spline objects must be made subordinate objects of these objects.
- The **Extrude** object moves the subordinate spline and can create an object with a consistent cross-section.
- The coordinate system used for extrusion is the **Extrude** object's coordinate system.
- If the Hierarchical option is enabled, multiple splines can be extruded simultaneously. Each subordinate spline's coordinate system will be used individually for extrusion.
- The Lathe object rotates the subordinate spline around the Y axis. This can be used to create rotationally symmetrical shapes.
- If additional movement or scaling is added to the rotation, spiral or snail shell shapes can be created.
- The quality of a cross-section is defined by the spline's intermediate points. The segments for the spline's rotation are defined by the Lathe object.
- The Loft object covers subordinate splines with a polygon cover. The splines must be placed in same hierarchical order as the direction in which the polygon cover should run.
- The direction in which the splines run must be identical to avoid twisting. Also, the end points of all splines should lie on or very near the same plane.
- The number of subdivisions for the Loft object is only controlled using the Mesh Subdivision setting. The spline's intermediate points will not be used for calculation.
- The **Sweep** object makes it easier to create objects such as cables, pipes or hoses.
- A profile and a path are needed to create such objects. The profile spline must lie at the top of the hierarchy below the **Sweep** object.
- A third spline can be used to control the size and angle of the profile along the path.
- The angle and scale of the profile can also be defined using the function curves in the **Sweep** object's Object tab menu.
- All **Generator** objects for splines can also generate Caps surfaces. To do so, the splines must be closed.
- Caps surfaces can optionally be given rounded edges, whose shape can be defined. The type and arrangement of polygons for the Caps surfaces can also be defined.
- N-gons help reduce the total number of polygons and simplify the geometry.
- The uniformly arranged triangles or quads are better suited for Caps surfaces if the object will be subsequently deformed.

5.4 Polygon Modeling

In addition to modeling using Primitives or splines, polygon modeling is Cinema 4D's most powerful modeling method for creating just about any shape. Converted Primitives or spline models can be used as a basis. You can also work with a completely empty polygon object and add polygons step-by-step. This section will introduce various polygon modeling techniques as well as the most important polygon modeling tools.

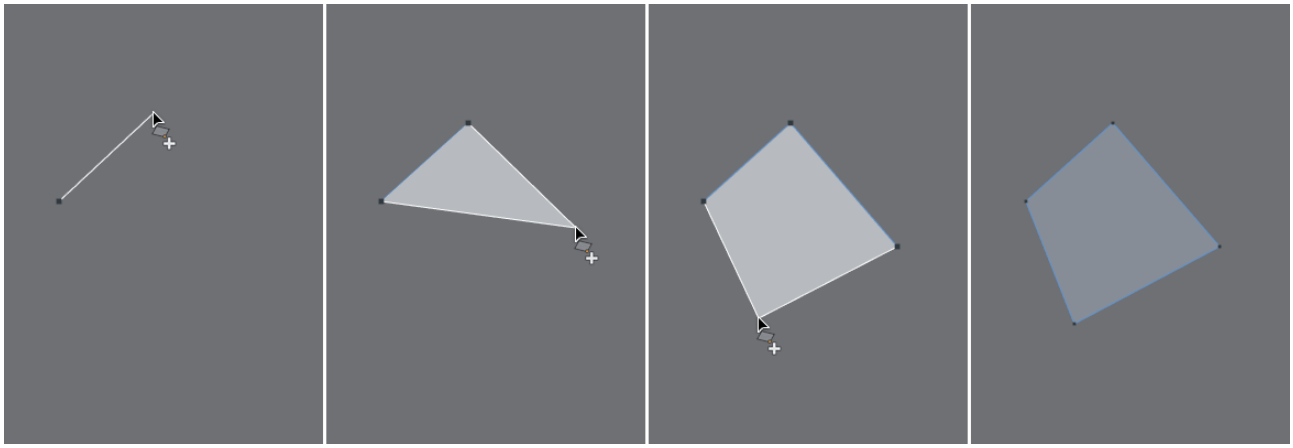
5.4.1 Creating Polygons

Generally speaking, all commands and functions for polygon modeling can be found in the main **Mesh** menu. Note that the **Use Edge** or **Use Polygon** modes must be selected before a command is executed or a function selected. Commands that do not apply will be grayed out. You will also notice that not all commands are available for all modes. Several commands behave completely differently depending on the mode you're in. On the other hand, tools work independently from the operating mode that is currently active. This also applies to the **Polygon Pen**.

To create polygons, a **Polygon** object (**Create**/**Primitives**/**Empty Polygon**) must be present on which these surfaces and points can be created. If new polygons are created using the **Polygon Pen** (**Mesh**/Create **Tools**), an **Empty Polygon** object will automatically be created (if another **Polygon** object was not already selected) to which the new surfaces will be added.

The Polygon Pen, which can be found in the **Mesh** menu, offers a menu from which you can define if you want to work with points, edges or polygons. Depending on the selection made, the tool will behave differently when creating polygons.

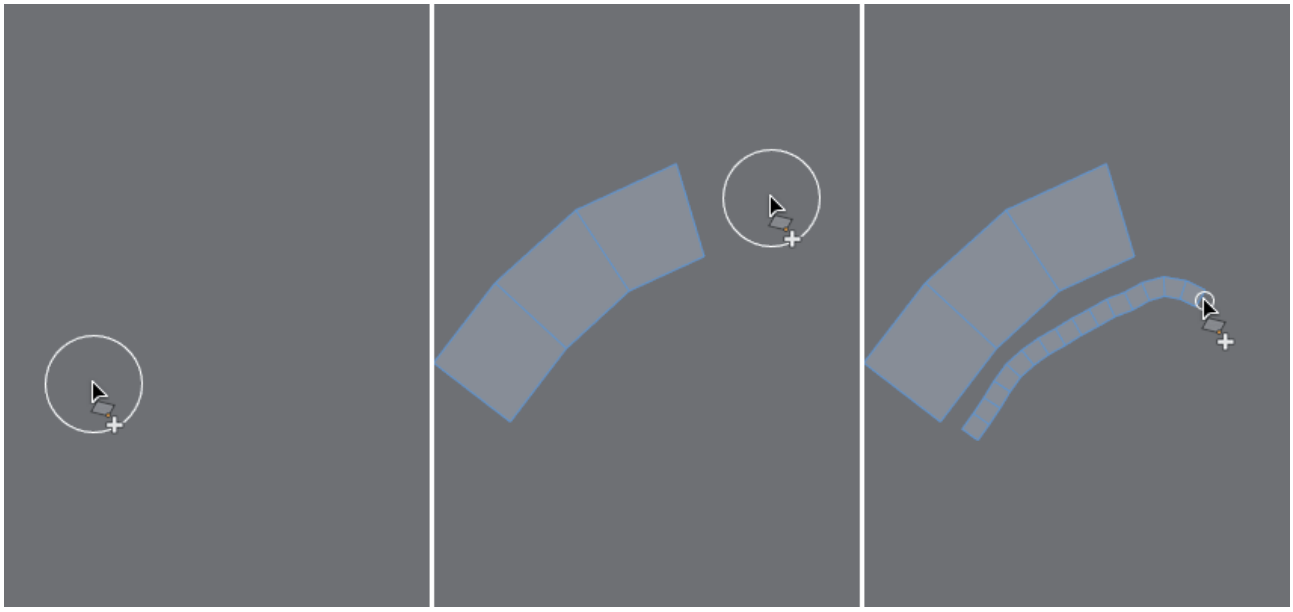
In Use Point mode, new points can be created directly by clicking in the Viewport.



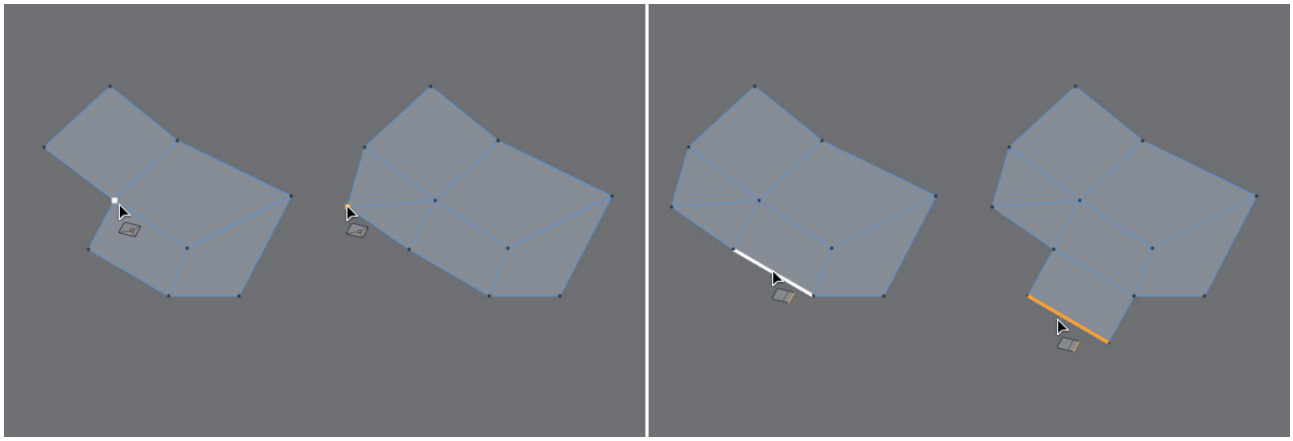
Double-clicking on the last point will create a surface between all points. The **Polygon Pen's** ***Attribute Manager*** settings can be used to define which type of polygon should be created.

The **Create N-gons** mode lets you click any number of points in 3D space. Double-clicking on the last point created will create a surface. If **Quad Strip Mode** is enabled, the last polygon will automatically be created after the 4th point is set. Subsequently set points will automatically be connected to the previously created polygon, which means that only two additional points have to be set to create the next polygon. This makes it easy to quickly create polygon loops.

Switch to **Use** Polygon mode if you want to create a strip of polygons very quickly. The **Polygon Pen** can then be used like a brush to paint a series of polygons.

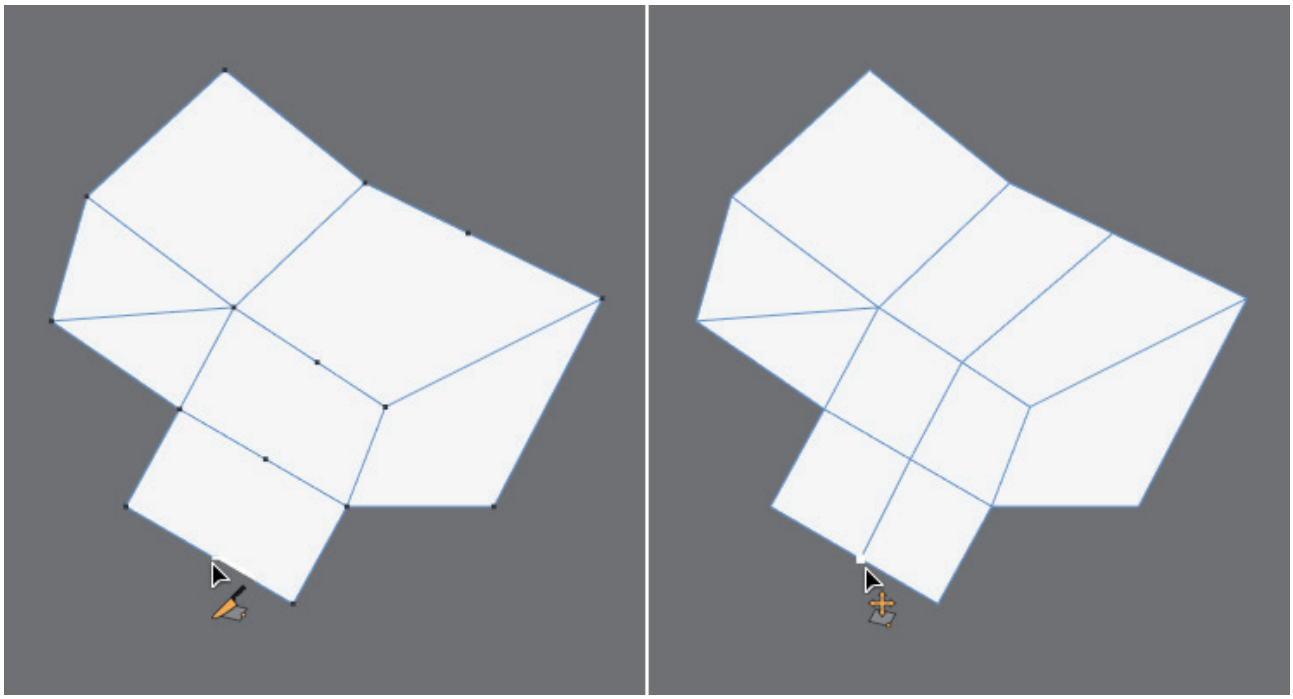


The size of the polygons is defined by the **Polygon Brush Radius** value. To start painting a new series of polygons starting at an existing edge, press the **Shift** key while painting. Otherwise, the **Ctrl/Cmd** key can be pressed to duplicate and move an existing element (point, edge or polygon).

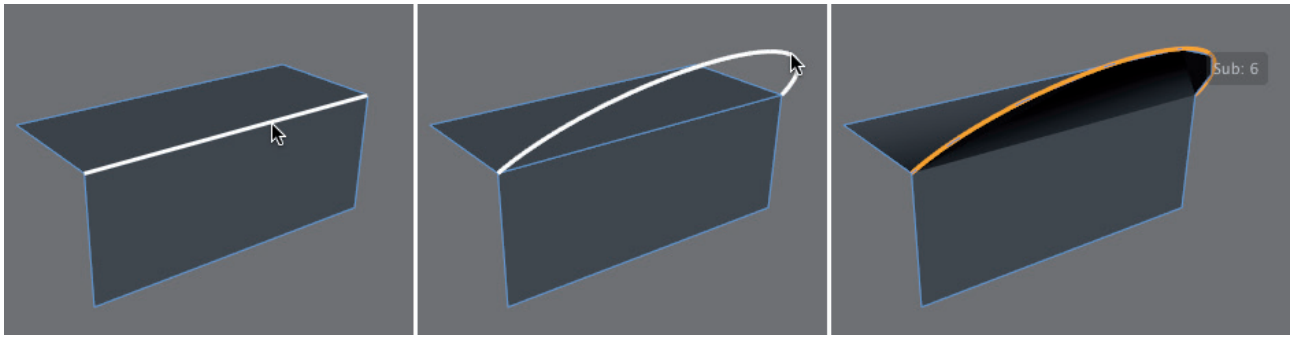


This process is called extruding and can also be done using a separate **Extrude** tool (**Mesh** menu).

Existing edges can be cut by **Shift** + clicking on an edge if the tool is in **Point** or **Edges** mode. If the **Create N-gons** option is enabled, only new points will be added to the edge, which can then be connected to a real edge by clicking on them in the correct order.



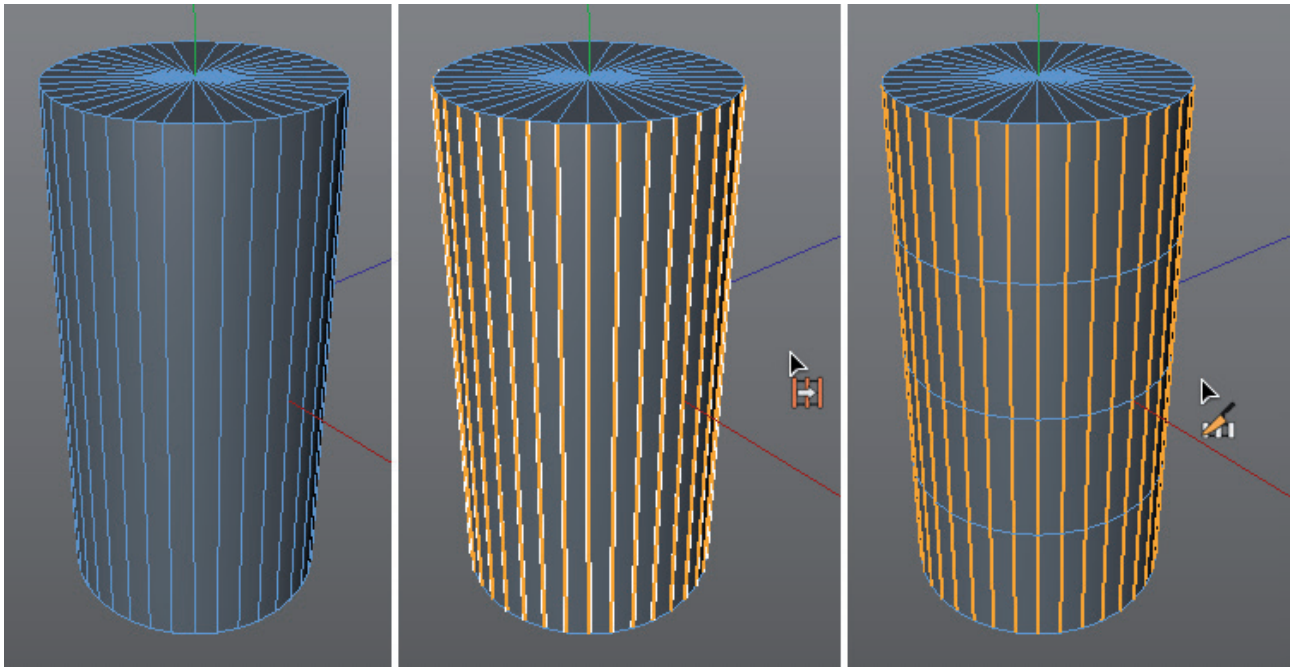
If the **Polygon Pen's** cursor is positioned over an edge while the **Shift** and **Ctr/Cmd** buttons are pressed, an arc will appear between the ends of the edge, which can be scaled by dragging the mouse. A surface will appear when the mouse is clicked and dragged to the left or right. Dragging the mouse lets you adjust the number of points along the arc. This subdivision is also displayed as an arc subdivision in the **Polygon Pen's Arc Subdivision** setting. The **Arc Direction Max Angle** setting helps define the orientation of the arc when an edge is used that lies between polygons. If the defined angle is less than that of the polygons the arc will be oriented perpendicular to the surfaces. Otherwise the arc will lie on one of the neighboring polygons.



If the arc's radius must be equal to half the edge length, enable the **Create Semi-Circle** option in the **Polygon Pen's** settings. If polygons already exist, their points, edges or the polygons themselves can simply be selected and moved using the **Polygon Pen**. A **Tweaking Mode** menu is available from which you can choose the mode in which you want to work. In addition, all newly created structures can automatically be projected onto surfaces that face the camera from the current angle of view. Enable the **Reproject Result** option to do so. The **Polygon Pen** combines numerous important polygon tools in a single tool. Each of these tools is also available separately if, for example, multiple elements have to be edited or complex cuts or extrudes have to be made. These tools can be found in the main **Mesh** menu and can be applied to selected points, edges or polygons. The most important tools are described below.

5.4.2 Cutting Edges

Many modeling tools are designed to further subdivide existing polygons. The additional points created after applying these tools can be used to further modify the surface. One of these tools is the **Edge Cut** tool, which is located in the **Mesh/Cut** menu and is only available in **Use Edge** mode. This tool needs an edge selection to work. As a rule, the **Ring Selection** tool can be used to make the corresponding selection because parallel or opposing edges are needed, which can be split (cut).



When using this tool, it's **Options** and **Tool** tabs should both be activated in the *Attribute Manager*.

To activate both tabs, simply click and drag over them. These settings can also be found with numerous other tools. The **Offset** value defines where the edges should be cut. The default value of 50% will cut the selected edges at the center. The **Subdivision** value defines the number of cuts. Here you can see that several modifications that work parallel to one another can be made in a single step.

If multiple cuts are made, the **Scale** setting can be used to define the spacing between them. Enabling the **Create N-gons** option will create new points on the cut edges. The surfaces in-between will be turned into N-gons, which means that they will not be given any new editable edges. As a rule it makes sense to disable this option.

This tool's function is first set into action when the **Apply** button is clicked in the **Tool** tab's menu. The Option tab's settings can subsequently be modified again. If **Realtime Update** is enabled, the results will be made visible in the Viewport.

Clicking on the New Transform button will complete this tool's function and automatically select it again for renewed use. The previous steps can then no longer be edited using the tool's settings. If this is necessary, use the **Edit** menu's **Undo** function. Note that the **Loop/Path Cut** tool also has a comparable function that offers an even more detailed preview function and a higher degree of control.

5.4.3 The Cut Tools

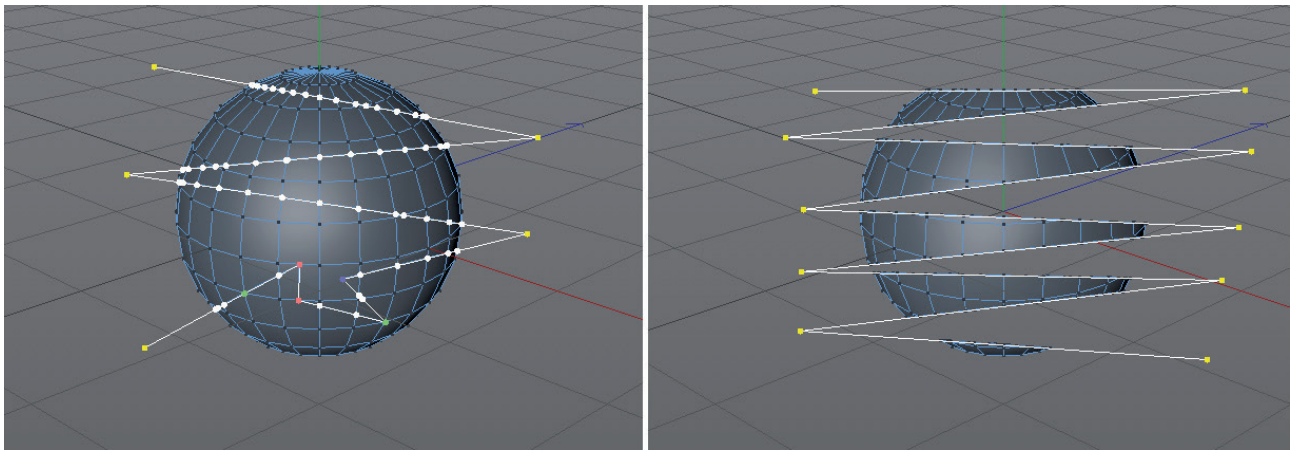
In addition to the cut tools there are three additional tools that can be used to subdivide an object: **Line Cut**, **Plane Cut**, and **Loop/Path Cut** tools. When using these tools it doesn't matter which operating mode you're in as long as points, edges or polygons can be edited. Each of these tools has interactive components, which, for example, make it possible to precisely position cuts before performing the final operation.

5.4.3.1 Line Cut

This tool requires at least two clicks of the mouse to define a cut line directly in the Viewport along which new points or edges should be created. If **Only Visible** is enabled, the cut will only be made on visible surfaces. Otherwise the cut will be projected onto the entire object from the current angle of view. The Line Cut tool can also be applied to Spline objects to add new spline points at the points of intersection.

If the Line Cut tool is applied to a polygon object, colored points will show where new points will be created on the object after the cut is applied. Yellow points represent the cut line's points that lie outside of the object. This is, for example, the case if the start or end points lie outside of the object. If the cut line cuts an existing object edge, a white point will be created. If the tool is clicked on an edge, a red point will be created; if the tool is clicked on a polygon a blue point will be created; if the tool is clicked on a point a green point will be created. This color scheme makes it easy to recognize which element was clicked upon.

If the **Single Line** option is disabled you can create a cut of any length, which can also run across an object multiple times.



If the **Single Line** option is enabled, a cut's existing points can even be moved. To do so, click and hold the LMB on the point in the Viewport. This also works with other points that are automatically set along a cut. **Ctrl+click** on a white cut line to add points, e.g., on a polygon. If the **Single Line** option is enabled, a cut line can, in conjunction with the activated **Infinite Cut** option, be extended at each end to extend the cut across the entire object.

The **Auto Snap** option ensures that the points snap to the corners or edges of the object to which the cut line is applied. Note that cut line points on polygons that are clicked upon will only generate new points on that polygon if the **Cut Polygon** option is enabled. Otherwise new points will only be created on existing object edges. The neighboring polygons will then automatically be turned into N-gons, which can themselves end up having more than four corner points.

If an object already contains N-gons at curved regions, this curvature can be maintained when the region is cut if the **Preserve N-gon Curvature** option is enabled. Such N-gon regions can be generated after a **Bevel** has been applied. If polygons are cut, the new edges that are created can be selected if the **Select Cuts** option is enabled. If edges or polygons were selected prior to activating the Line Cut tool, the application can be restricted to these if the **Restrict to Selection** option is enabled.

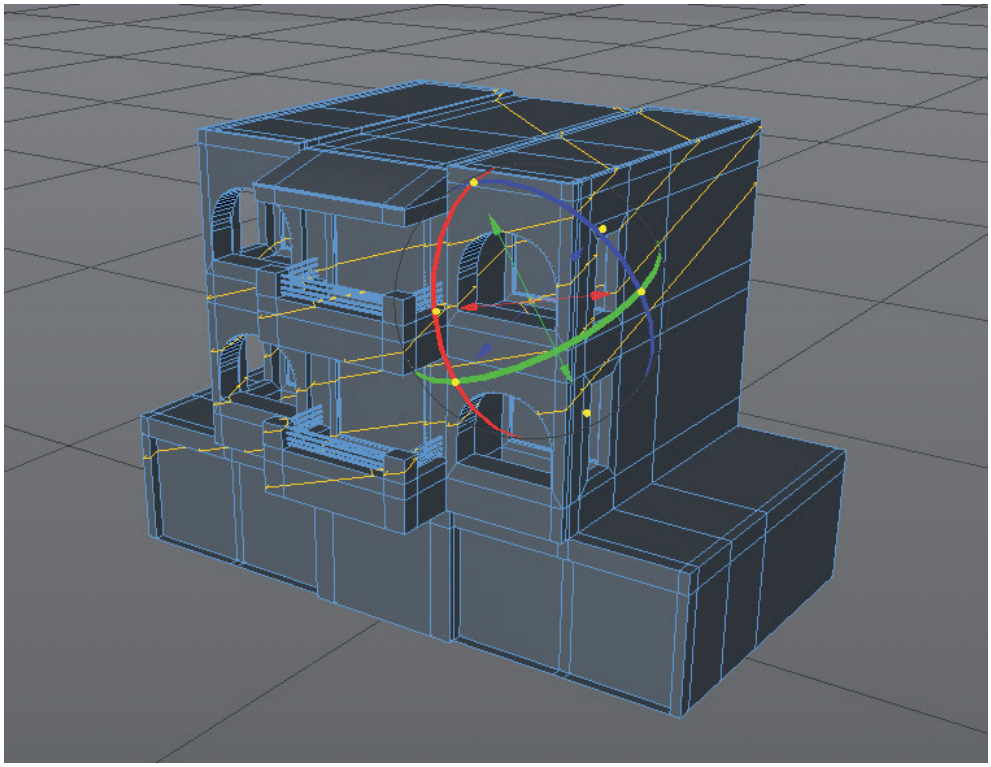
The **Angle** Constrain option can be enabled if the cut should run along a specific angle. The cut line can then only be created in multiples of the defined angle. A comparable effect can also be achieved by pressing the Shift key while creating cut lines. This will automatically apply the quantizing function.

Once the desired cut has been defined, press the Esc key to exit the interactive mode and apply the cut. The **Slice Mode** will define the type of cut that will be made. This menu is only available if the **Visible Only** option is disabled. If **Slice Mode** is set to **Cut**, points and new edges will be created along the cut line. The cut object will stay complete. The **Split** mode will, at first glance, have the same result but will in fact split the object along the cut line. If **Select Connected** is used, only the individual object halves can be selected and the polygons can be modified independently of the other object half. The modes **Remove Part A** and **Remove Part B** will delete one or the other half of the cut object, respectively. However, it's difficult to predict which half will be deleted. If the wrong half is deleted simple switch to the other mode.

5.4.3.2 Plane Cut

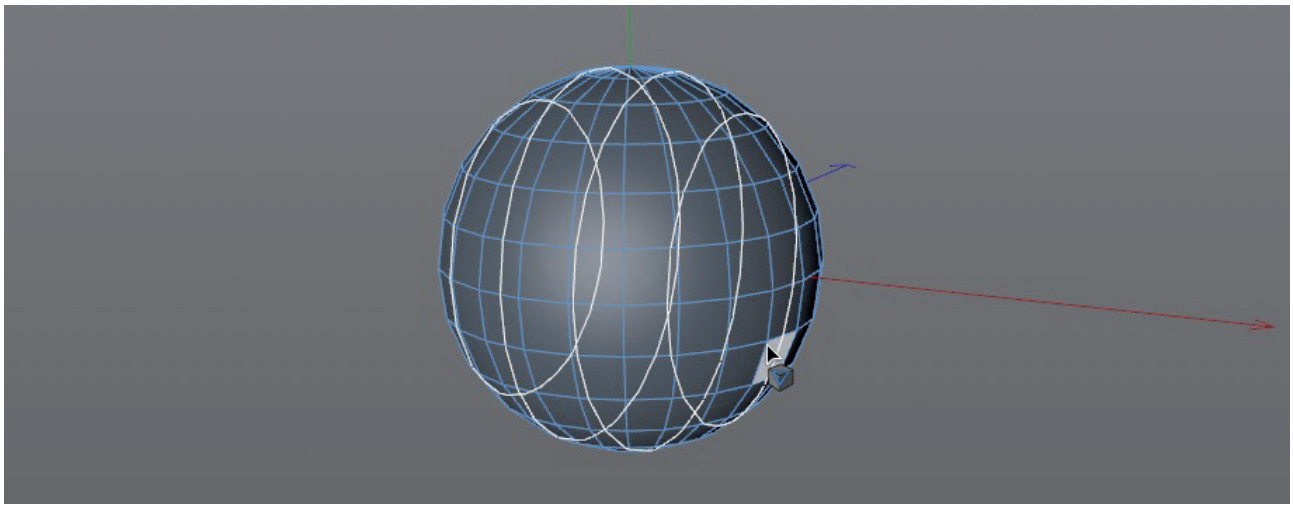
This tool can also be used to create a cut line between two points. However, this tool uses a gizmo axis system to precisely define the cut on a spatial plane. To do so, rotate the view slightly after setting the start and end points of the cut line. A special axis system will be displayed on the object that lets you move or rotate the cut plane for the object. This gizmo has handles that can be dragged to rotate the cut plane. These handles automatically snap to the object's corner points, which makes it easier to orient the plane precisely. If a **Plane Mode** other than **Free** is selected, the cut plane can be arranged relative to the Object, World or Camera axis system. The plane to which the cut should be applied parallel can be selected from the **Plane** menu. Alternatively, the **Plane Position** and **Plane Rotation** values can be used to define a precise numerical position for the cut plane.

If the value for **Number of Cuts** is increased, multiple cuts that run parallel to the original cut plane can be created.



The distance between these cuts can be defined using the **Spacing** value. The **Offset** value defines the translation relative to the original cut plane.

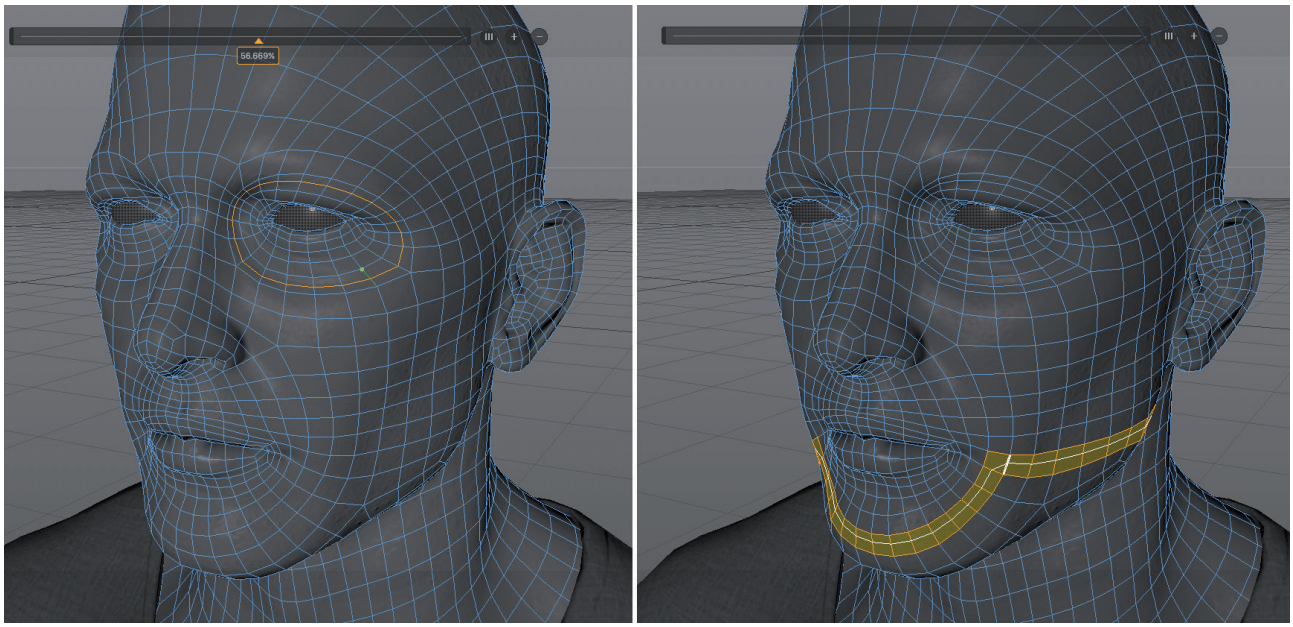
Enable the **Regular Slice** option if multiple cuts run across an entire object independently of a specific edge. This option is available in all modes except for the **Layer** mode and always references the defined **Number of Cuts** across the entire size of the object.



The remaining menus and options reflect those of the Line Cut tool and will not be explained again in detail here.

5.4.3.3 Loop/Path Cut

This tool offers two modes, which work similarly. In **Loop** mode a cut can be created from an edge over which the cursor lies. This makes it easy to create cuts that run along an existing band of polygons or a polygon ring, which, for example, is very useful for subdividing polygon loops when modeling 3D characters. This tool creates quads when applied, which makes it easier to smooth the surface using a Subdivision Surface object. If in **Path** mode, angled cuts can be created if an edge or polygon selection was previously set. If no selection has been set, the tool will behave the same as in **Loop** mode.



In both modes the **Number of Cuts** setting can be used to define the number of cuts that will be made when the tool is applied. The cuts will be distributed evenly along the cut edge. Otherwise, the proportional length relative to the edge length can be ascertained or defined by using the **Offset Mode** setting's **Proportional** option. If the **Edge Distance** option is selected, the position of the edge cuts is defined via the **Distance** value. This value can be subsequently modified. The Distance value always references the first edge that is clicked on, which is colored green in the Viewport. If edges are included within the cut that are shorter than this green edge, the cut will automatically be restricted to the selected segment.

In addition to the ***Attribute Manager*** settings there is also a slider displayed in the Viewport that shows the cut length of the selected edge. This slider can be adjusted to adjust the length of the cut interactively. You can double-click on the percentage value to manually enter a value. Additional cuts can be added to the edge by Ctrl + clicking on the slider. The "+" and "-" symbols next to the slider can also be used to add or remove cuts. Each time, a new cut will be set and distributed evenly along the edge. Clicking on the icon with three lines will evenly distribute existing cuts.

5.4.3.3.1 Additional Interactive Options

The **Reuse Cuts** option can be used to save selected cuts and, for example, apply them to the next edge that is clicked upon. The number and position of the cuts will be assumed. Cut shapes can also be transferred using this method. We will discuss this in detail later.

If the **Bidirectional Cut** option is enabled, the tool will search for polygons in both directions perpendicular to the selected edge. Otherwise the cut will only be applied in one direction. If **Symmetrical Cut** is enabled, a cut whose length is, for example, set to 10% of the edge length, will automatically be supplemented by a second cut that covers 90% of the original edge length. The cut will be applied mirrored at the center of the cut edge.

The cut position's Offset value will always be calculated from the end point of the selected line. This point will be colored green in the Viewport. If you want to have the offset calculated from a different edge point, enable the **Toggle Direction** option. If the **Stop Cut at N-gon** option is enabled, the cut will automatically end at the location at which an N-gon intersects with the cut edge. The cut edge can also be interrupted at pole points, e.g., of a sphere, by enabling the **Stop Cut at Pole** option.

If you want to restrict the placement of cuts along an edge you can enable the **Quantize Subdivision** option. The number of defined **Quantize Steps** will be used to evenly distribute the cuts along the edge. For example, a value of 5 will distribute the cuts in steps of 20%.

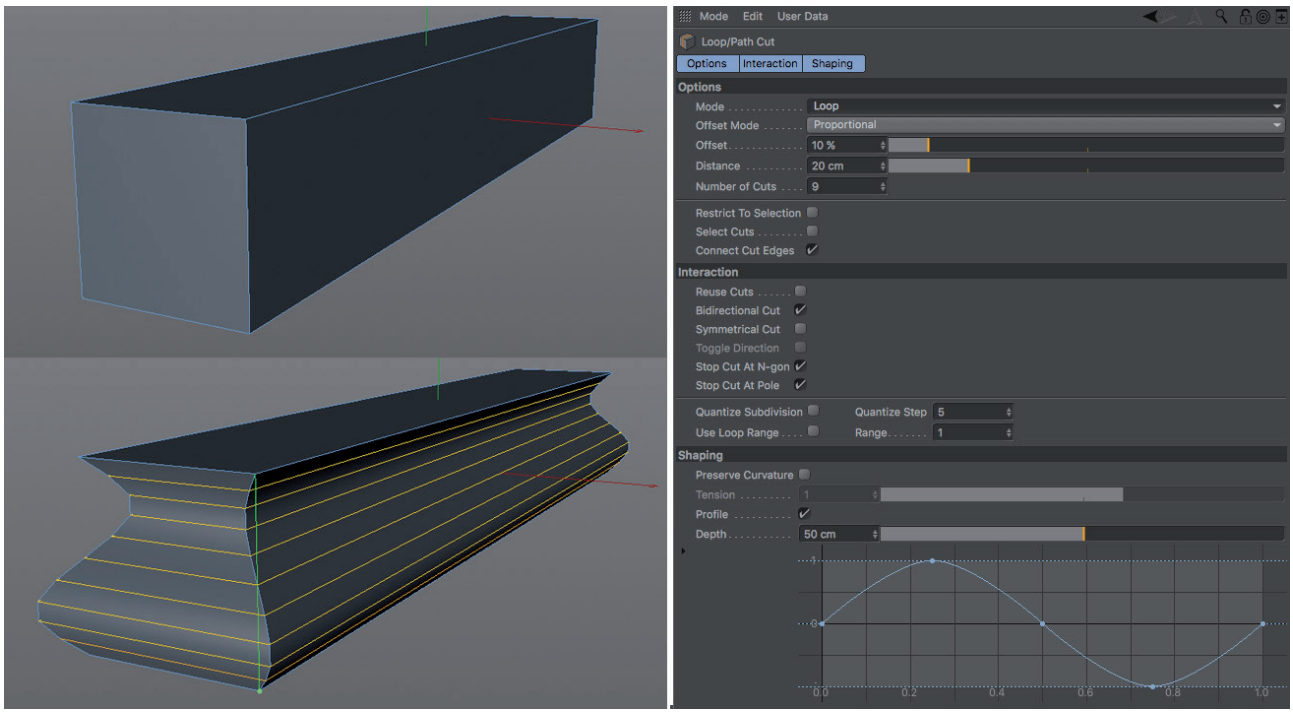
If you want to restrict the number of polygons that are cut, the **Use Loop Range** option can be enabled and a **Range** can be defined accordingly. The cut length will always be calculated starting from the edge that is selected. If the **Bidirectional Cut** option is enabled, the **Range** value will be applied in both directions, i.e., it will be doubled.

5.4.3.3.2 Shaping Settings

An object's shape will be maintained for all previously described cutting methods. Only new points, edges or polygons will be added. When cutting curved shapes, additional subdivisions are often applied to help smooth the object in these regions or to more precisely shape the object. The **Loop / Path Cut** tool lets you do this in basically a single step.

If the **Preserver Curvature** option is enabled, the cuts will automatically be positioned to follow the curvature of the neighboring polygons. A sphere will be smoothed in the cut region. The **Tension** setting can also be applied to move the curvature outward or inward in the cut region. A comparable function is offered by the **Bevel** tool.

If you want to be able to apply cuts more independently of the object's curvature, enable the **Profile** option.



A curve can then be used to position the cut. The more cuts that are controlled in this manner, the more precisely the curve can be displayed. The **Depth** value defines the scale of the spline curve's depth and controls the possible distance of the cuts from the surface. Ctrl + click on the curve to add points. Selected points can be deleted by pressing the Del key.

5.4.4 The Create Point Tool

This function should already be familiar to you from the spline section. If this tool is selected, points can be added to edges or polygons simply by clicking on them in the Viewport. If **Cmd / Ctrl** is also pressed, new points can even be created in space by clicking with the mouse. The **Polygon Pen** tool can then be used to create a surface between these points. If the **Create Tri- / Quadrangle** option is disabled, polygons created using the new points will automatically be converted to N-gons. Otherwise normal triangles and quads will be used to connect the points and create corresponding surfaces. This tool works identically in **Use Point**, **Use Edge** and **Use Polygon** modes. However, contrary to the **Polygon Pen** tool, a polygon object must be selected before selecting the tool.

5.4.5 The Subdivide Command

The **Subdivide** command in the **Mesh / Add** menu can be used to increase the subdivision of a given surface. The **cogwheel** next to the command name means that additional options can be displayed for this command. Simply selecting the command will execute it with the previously defined settings. Clicking on the **cogwheel** on the other hand will open a dialog window the corresponding settings.

The subdivision can be made in various degrees. A **Subdivision** value of 1 will subdivide each polygon horizontally and vertically in the center. A quad will be turned into 4 quads. A **Subdivision** value of 2 will double the subdivision and turn 4 quads into 16 quads. You can see that very small **Subdivision** values will produce a relatively high number of new polygons. Therefore, use small values and repeat the same subdivision if you need more polygons or until you reach the desired number of polygons.

If the newly created polygons should also be used to round the geometry, enable the **Smooth Subdivision** option before executing the command. This will smooth hard edges and the transitions from the original polygons. Note that the shape's original size will be reduced slightly with each subdivision.

The **Maximum Angle** value defines the angle to which neighboring polygons will still be smoothed. The default value of 180° will cause the entire object to be smoothed. Note that the **Subdivision** function can be restricted to selected polygons. If you want to subdivide an entire object, switch to **Use Model** mode or deselect all surfaces in **Use Polygon** mode before executing the Subdivide command.

▶ *See: Exercises for commonly used tools*

SUMMARY: POLYGON MODELING

- Polygons are made up of points connected via edges.
- To create polygons, a polygon object must be present on which points can be placed. This can be an empty **Polygon** object, a converted parametric Primitive or a converted spline **Generator** object.
- Almost all polygon modeling tools are located in the **Mesh** menu.
- Before using a given modeling tool, you must first switch to the correct operational mode. Not all functions and command work in all operational modes.
- The **Polygon Pen** tool can be used to create either an empty polygon object or to add new points and polygons to an existing polygon object by clicking on it in the Viewport. The **Shift** and **Cmd/Ctrl** keys, respectively, can be used to extrude, round or cut existing structures.
- The **Create Point** tool can be used to add individual points on edges or on an existing polygon surface.
- Already selected edges can be cut using the **Knife** tool. The **Ring Selection** tool is commonly used to select edges to be cut.
- The **Line Cut** tool can be used to cut or separate objects very precisely.
- The **Plane Cut** tool can be used to create a cut parallel to a specific axial plane.
- The **Loop/Path Cut** tool can be used to activate an automatic cut function that can be positioned absolutely or spaced according to defined values along an edge. A comparable function is offered by the **Edge Cut** tool. The **Loop/Path Cut** tool can, however, also be used to position the cuts to create additional curvature or to affect the shape of the cut object.
- The Subdivision tool can be used to uniformly subdivide selected polygons or entire objects. A smoothing algorithm can also be used to make surfaces look even more organic and round the subdivided regions even more.
- Several modeling commands offer additional settings that can be accessed by clicking on the cogwheel symbol next to their name in the menu.
- Almost all tools offer N-gon creation as an option. This special type of polygon can contain triangles or quads that are not visible to us. N-gons give the object a more tidy appearance. However, edges within an N-gon cannot be accessed, which restricts the degree to which the arrangement and number of triangles or quads can be modified within the N-gons.

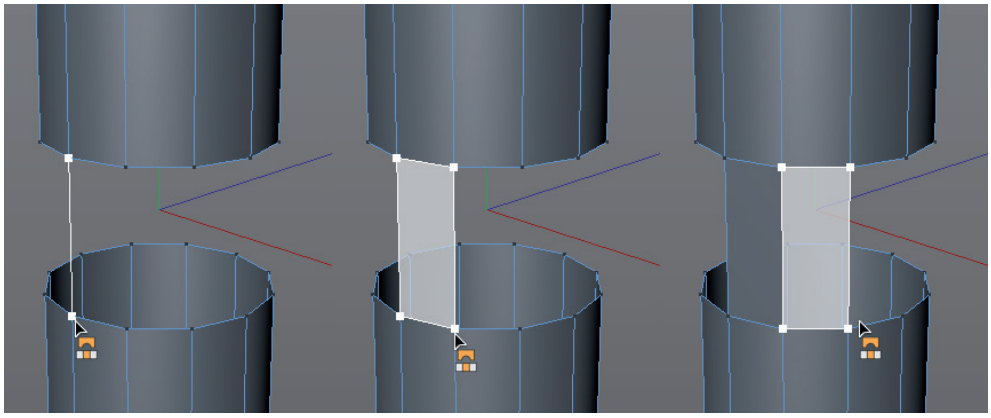
5.4.6 Additional Modeling Tools

The tools discussed to this point were designed for subdividing existing polygons and adding new points or polygons. The tools described in this section have additional functions and can be used to change the shape of an existing object. A major focus is placed on extrusion, i.e., the duplication and subsequent movement of surfaces or edges. Also, the **Bridge** tool makes it easier to connect existing elements.

5.4.6.1 The Bridge Tool

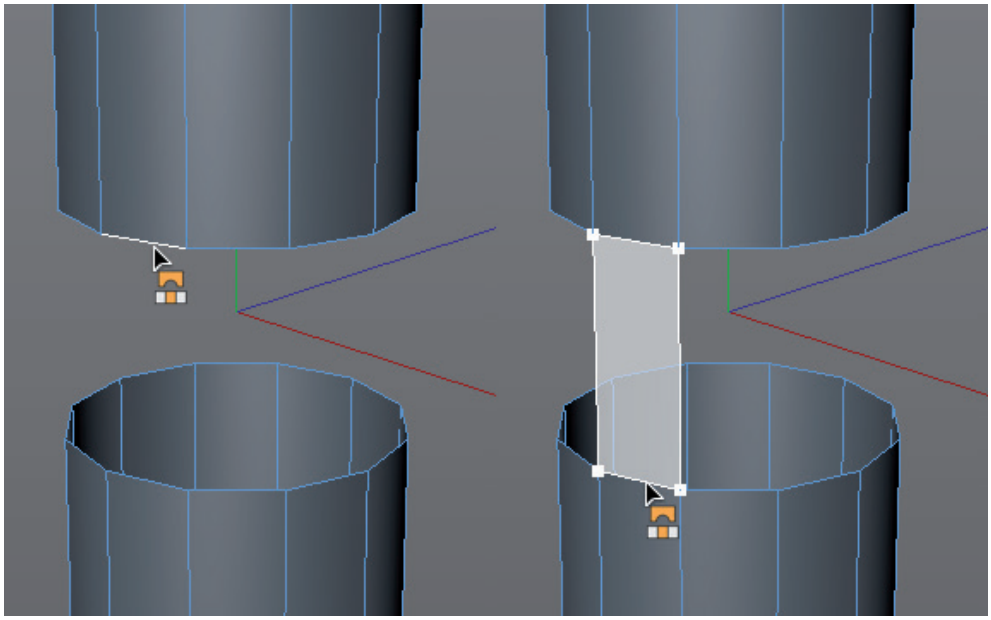
The **Bridge** tool can be used to connect points, edges or entire polygons. In doing so, new individual polygons or complex polygon tunnels can be created. The tool is implemented interactively by clicking in the Viewport.

If you want to create new polygons between existing points in **Use Point** mode, select the **Bridge** tool and click and drag to connect the first and second points. When the mouse button is released, a white line will remain, which represents the polygon's first edge. Use the same method to connect the two points of the polygon's opposing edge. When the mouse button in released, a polygon will be created.



This process can be repeated as often as necessary. Note that each subsequently created edge will be adjacent to the previously created edge and neighboring polygons will be created. Pre-selecting the points is not necessary.

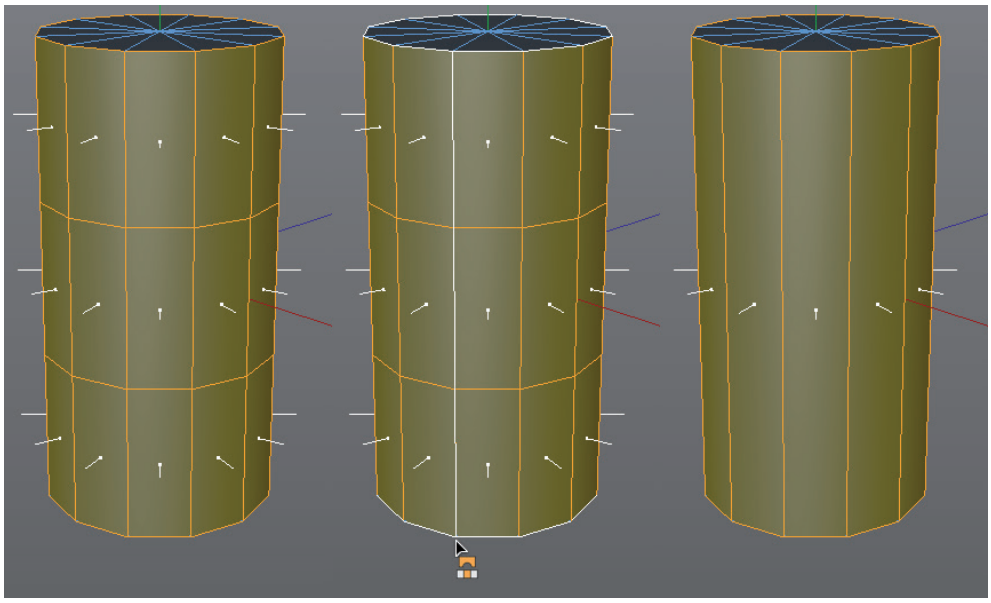
When in **Use Edge** mode, all you need to do is connect two edges to create a polygon – a surface will be created between them. Pre-selecting edges is not necessary.



When in **Use Polygon** mode, an additional option will be made available **Bridge** tool's in the ***Attribute Manager*** settings. You must first select the polygons you want to connect using the **Bridge** tool. Facing polygons are well suited for this. If a connection line is then dragged between the polygons' corner points, a connecting "tunnel" will be created between the original polygons. This also works if multiple opposing polygons are selected as long as they can be used to create a coherent surface.

Activate the **Delete Original Polygons** in the ***Attribute Manager*** if the original surfaces should subsequently be deleted.

The **Bridge** tool can also be used, for example, to connect multiple bands of parallel polygons into a single band.

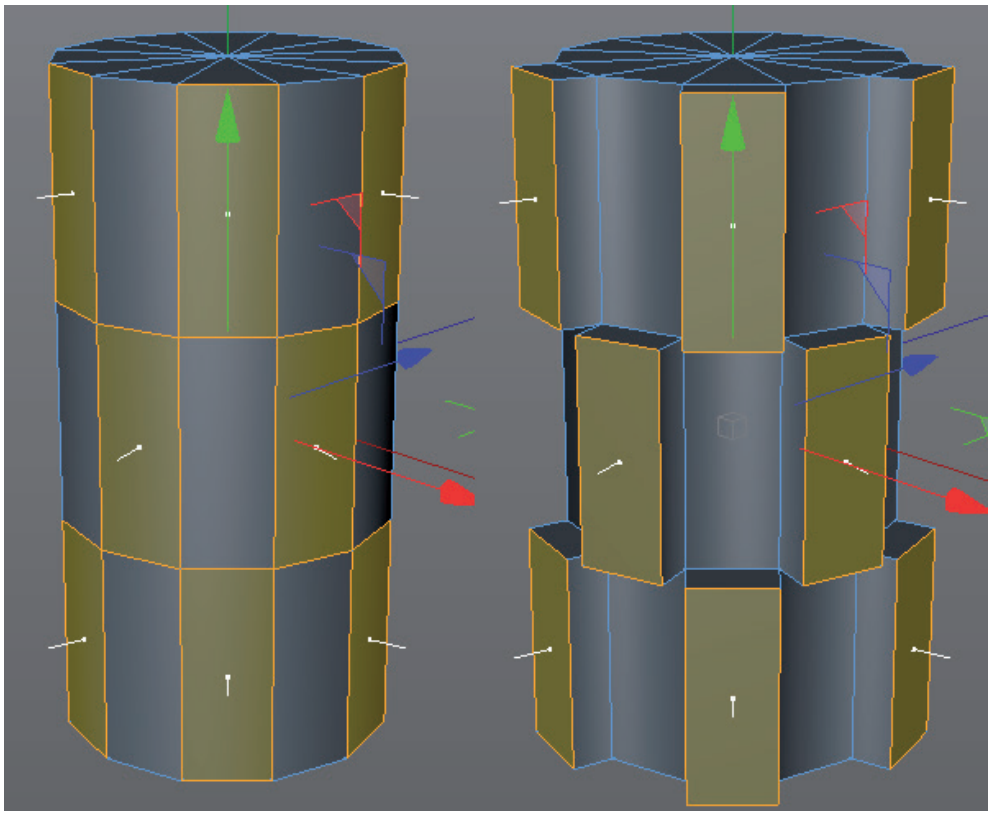


To do so, the **Delete Original Polygons** option must be enabled so no overlapping, redundant surfaces are created.

5.4.6.2 The Extrude Tool

The **Extrude** tool primarily used in **Use Point** and **Use Edge** modes but can also be used in **Use Point** mode. If multiple points, edges or polygons should be extruded simultaneously, they must be selected before activating the **Extrude** tool. To extrude individual elements, simply activate the **Extrude** tool and click and drag on a given element in the Viewport. A similar function can be used without activating the tool, e.g., by selecting the **Move** tool and **Cmd/Ctrl** + dragging the desired element in the Viewport. When a point, edge or polygon is moved, a copy will be created, which often results in the object's original shape being enlarged by that element.

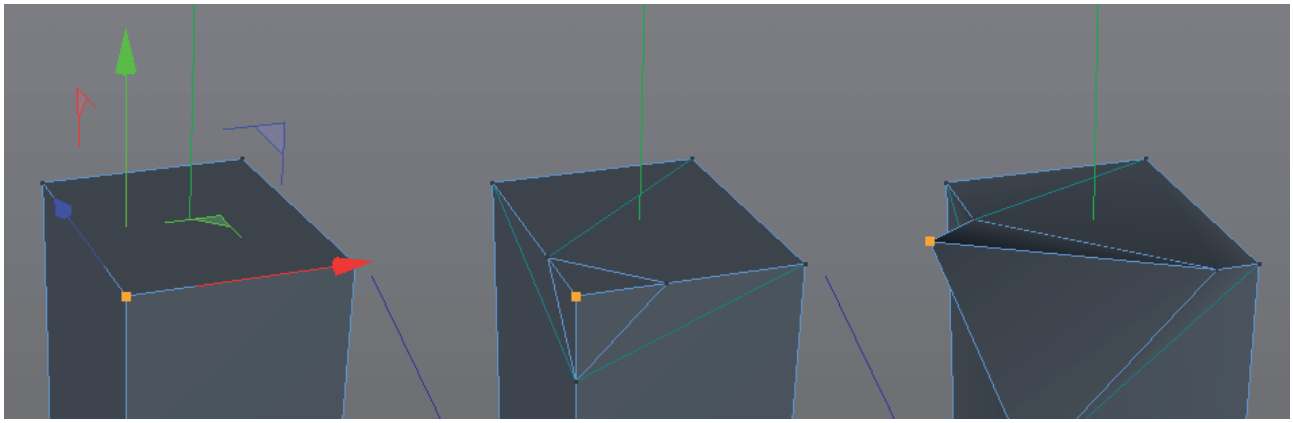
An extrusion can be done interactively with the mouse in the Viewport or numerically using the settings in the ***Attribute Manager***. These techniques can also be combined if, for example, edges or polygons should first be extruded by clicking and dragging in the Viewport.



The distance and direction can then be edited manually in the ***Attribute Manager***. This can be repeated until you switch to another tool, click on an empty region of the Viewport or by clicking on the **New Transform** button, which will reset the tool.

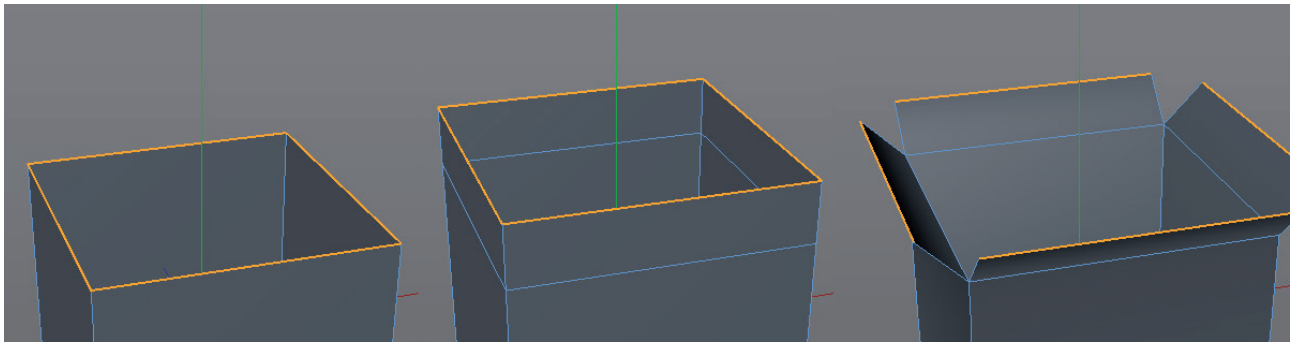
If the tool is implemented using only the *Attribute Manager* settings, the **Apply** button only needs to be clicked to confirm the first modification. All subsequent modifications must be initiated by clicking on the **New Transform** button. This procedure is identical for all modeling tools that can be implemented using *Attribute Manager* settings.

In **Use Point** mode, selected points can be moved in the direction of the surface Normals using the **Offset** value. The **Bevel** value can be used to place additional points at adjoining edges. Using this combination, chamfers can be easily created at corner points.



If the **Create N-gons** option is enabled, newly created polygons will be restricted to the region directly around the extruded point.

In **Use Edge** mode, open polygon edges can be expanded using the **Extrude** tool. The **Edge Value** option will be made available, which, in combination with the **Offset** value, can be used to precisely define the orientation of the new edges in relation to the original orientation of the edge.



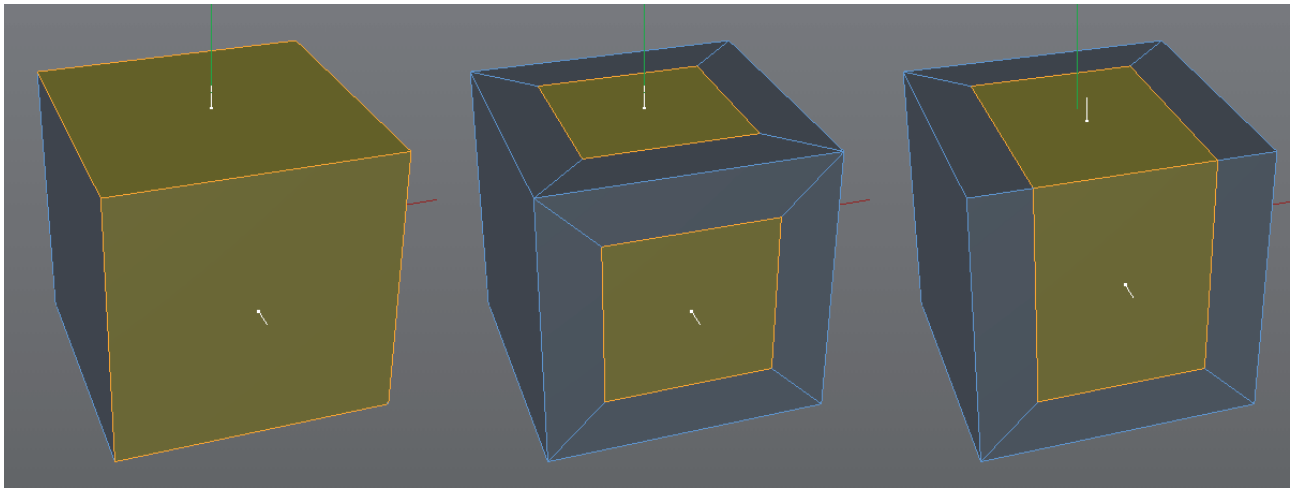
If multiple edges are selected, the **Preserve Groups** option, in conjunction with the **Maximum Angle** value, can be enabled to make sure that coherent structures are maintained even after being extruded. The extruded structure will be separated in this region only if the angle between adjoining edges or polygons is greater than the **Maximum Angle** value. If **Preserve Groups** is not enabled or if the **Maximum Angle** value is very low, each selected edge will be extruded separately. This technique can, for example, be used to extrude and rotate all four flaps on a box in a single step.

When in **Use Polygon** mode, the Extrude tool's behavior is like that of extruding edges, only that no angle can be defined for the offset. The **Create Caps** option ensures that the polygons originally selected are maintained after extrusion. This option is the opposite of the **Bridge** tool's **Delete Original Polygons** option. If **Create Caps** is enabled, only objects made up of polygon sides can easily be converted to solid objects. This can be irritating when using the **Subdivision Surface** modeling method. Therefore, make sure the **Create Caps** option is set correctly before implementing the **Extrude** tool.

Use the **Var.** values (Variation) to vary the points' bevel offset if you're less concerned with precision than with adding random-looking details. The best results can be achieved if multiple elements are selected and the **Preserve Groups** option is enabled.

5.4.6.3 The Extrude Inner Tool

This tool in a similar fashion as the Extrude tool but can only be used in **Use Polygon** mode. This tool scales the polygons instead of vertically moving the extruded polygons using an offset.



The **Offset** value makes the polygons larger or smaller. This can be used to increase the subdivision with in a specific region or to prepare a region's scale for subsequent extrusion. This is why the **Extrude Inner** and **Extrude** tools are often used together.

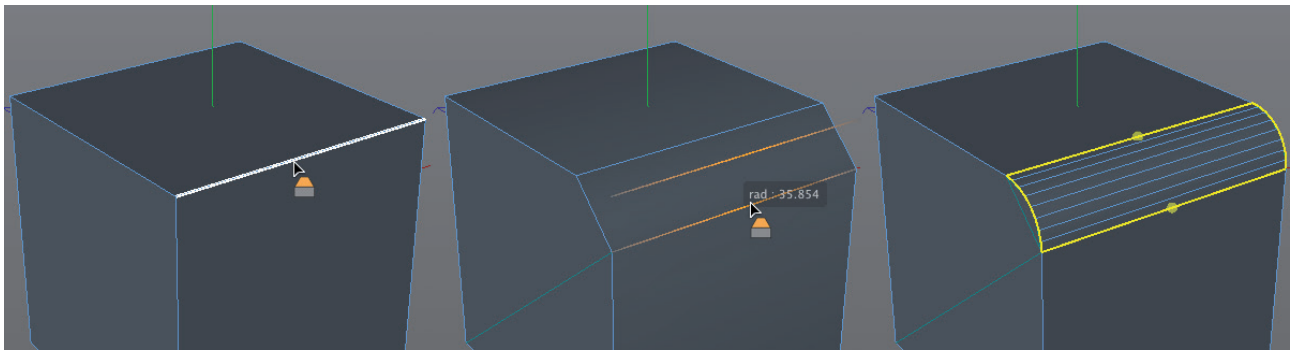
The Subdivision value can also be used to subdivide the added regions with additional polygons.

In all other respects, this tool's settings reflect those of the **Extrude** tool. You can use this tool interactively or via the *Attribute Manager* settings.

5.4.6.4 The Bevel Tool

Beveling is the rounding of a selected edge region and is a very important function since to real-world object has perfectly sharp edges. Even if a given object has a mechanical, hard look, rounding its edges very slightly is recommended in order to achieve a more natural look and a more realistic lighting effect on the object's surfaces.

The **Bevel** tool can be used in **Use Point**, **Use Edge** or **Use Polygon** mode and can be used to interactively modify individual, unselected elements. All you need to do is click on and drag the element in the Viewport.



To simultaneously modify multiple points, edges or polygons, these must be pre-selected. The **Bevel** tool offers numerous settings in the *Attribute Manager*, which can also be edited after a bevel has been created.

Use the **Bevel Mode** options to define if an edge should be rounded (**Chamfer**) or consist of parallel structures (**Solid**). The latter is mainly used in combination with **Subdivision Surface** modeling. This method is primarily used to model organic shapes and will be discussed later in this curriculum.

Offset Mode defines how the underlying **Offset** value will be applied. If set to **Fixed Distance**, the offset will be absolute, i.e., the current spacing and scaling of edges and surfaces will be disregarded. If two separate edges within a large and a small polygon are selected, both will be given the same amount of bevel, which can, of course, result in surfaces intersecting others if larger values are used. On the other hand, this is the only mode that will apply the **Offset** value as precisely as you want it.

If the **Proportional** option is selected, the Offset will orient itself according to the length of the neighboring edges and will therefore only be displayed as a percentage. This can result in an asymmetrical bevel if the size of neighboring surfaces differs greatly. A positive aspect is that it is easier to avoid new surface intersecting the existing structure. However, it is not possible to define a precise rounding radius, which you might want to use at other locations on the object.

The third option, **Radial**, behaves almost identical to the **Fixed Distance** option. Differences can be seen at corner points at which three edges meet. The rounding will be spherical.

Because a high number of polygons are required to create nice roundings, the number of bevel surfaces can be adjusted using the **Subdivision** value. The curvature of a given rounding can be scaled or even inverted using the **Depth** value. A rounding can, for example, be made concave instead of convex. Note that the **Depth** value has either an absolute or percentage effect depending on the **Shape** option defined below.

The **Limit** option restricts the size of the **Offset** to prevent rounded edges from bulging beyond neighboring surfaces. This option should generally be enabled to avoid unnecessary surface intersection.

The actual shape of the rounding is defined using the **Shape** setting. The default **Round** setting with a **Tension** of 100% represents a normal round bevel. Reducing the **Tension** value will reduce, flatten or even invert the rounding accordingly.

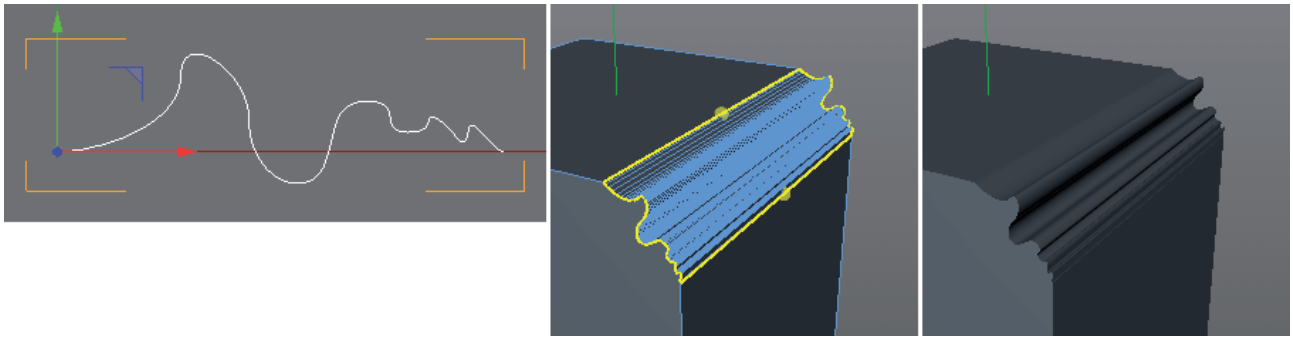
A custom rounding can be created by selecting the **User** option. A function curve will appear whose points and tangents can be edited to create the desired rounding. **Cmd**/**Ctrl** + click on the curve to add points. To delete a point, select it and press the **Del** or **Backspace** key on your keyboard. Right-clicking on or near the curve will open a context menu containing numerous commands that can be used to affect the curve's shape or the points' tangents.

How the curve is interpreted depends on the **Symmetry** option beneath the curve. If this option is disabled, the left and right ends of the curve will be spaced identical to the rounding's new edges. In other words, the center of the curve will represent the center of the rounding. If this option is enabled, the right end of the curve will represent the center of the rounding. Hence, only a symmetrical cross-section can be created for the rounding because that half of the rounding not represented by the curve is a mirrored version of the curve displayed.

The **Constant Cross Section** option attempts to keep the volume of the rounded profile constant throughout the rounding. Depending on the course of the selected edges, disabling this option might help create a better result.

Generally speaking, when using custom cross-sections you should avoid creating roundings that extend over sharp corner points, e.g., points at which three selected edges meet. In such instances, the rounding of the corner will not be harmonious. Rounding should take place in two steps: first for only two edges that meet at the point, then the third perpendicular edge can be rounded.

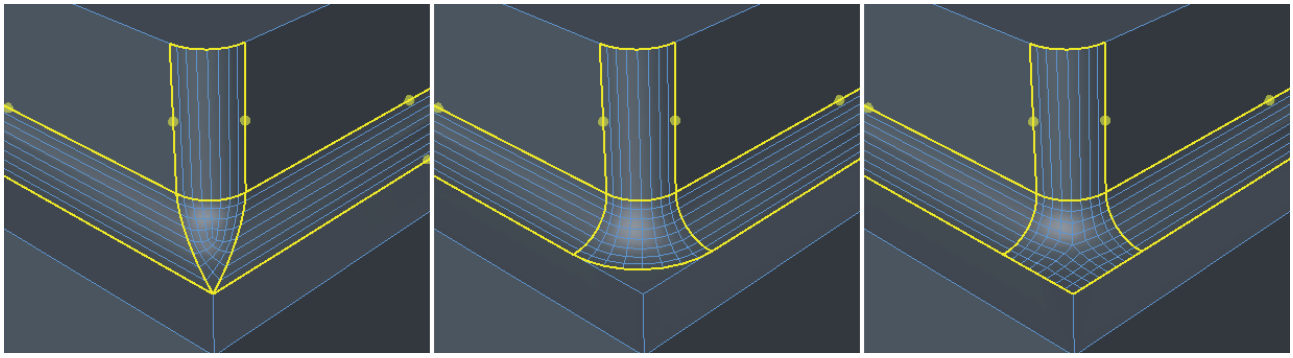
Another option for individual rounding is offered by the **Shape** setting's **Profile** option. A spline must be dragged into the **Bevel** tool's **Profile Spline** field.



The spline must be two-dimensional and open, e.g., exactly on the object's XY plane and its **Close Spline** option is disabled. The profile plane is defined in a separate menu. Note that the **Subdivision** value is grayed out when the **Profile** option is selected. The number of subdivisions is now defined by the profile spline's intermediate points. The **Depth** value defines the degree to which the shape will be created.

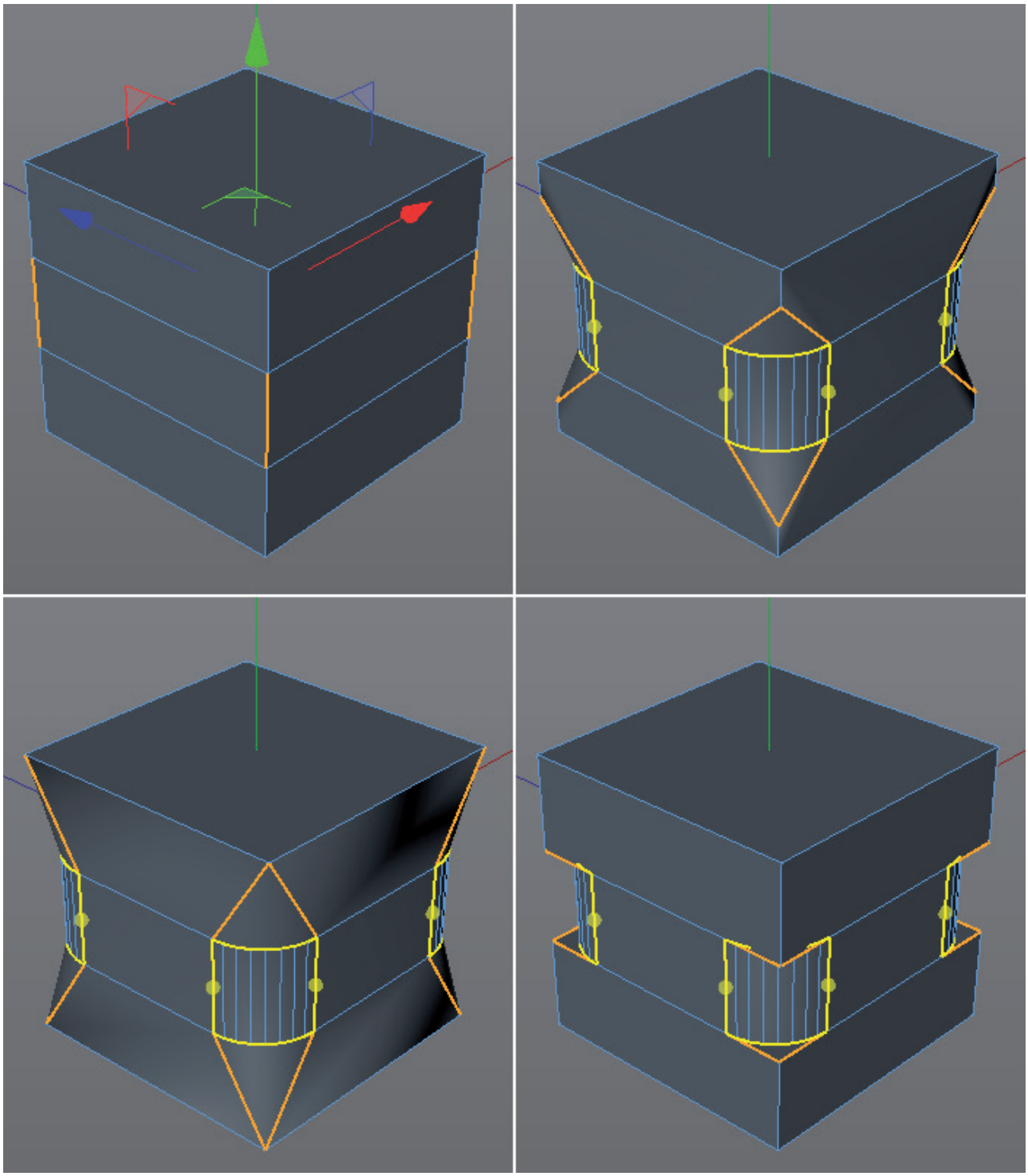
The Topology menu's settings are used to define the type of subdivision, primarily the rounding of neighboring sections on the object.

A miter is always created where at least three polygons meet at a single point that is part of a rounded edge. The **Mitering** setting defines how these polygons will be used to create the miter.



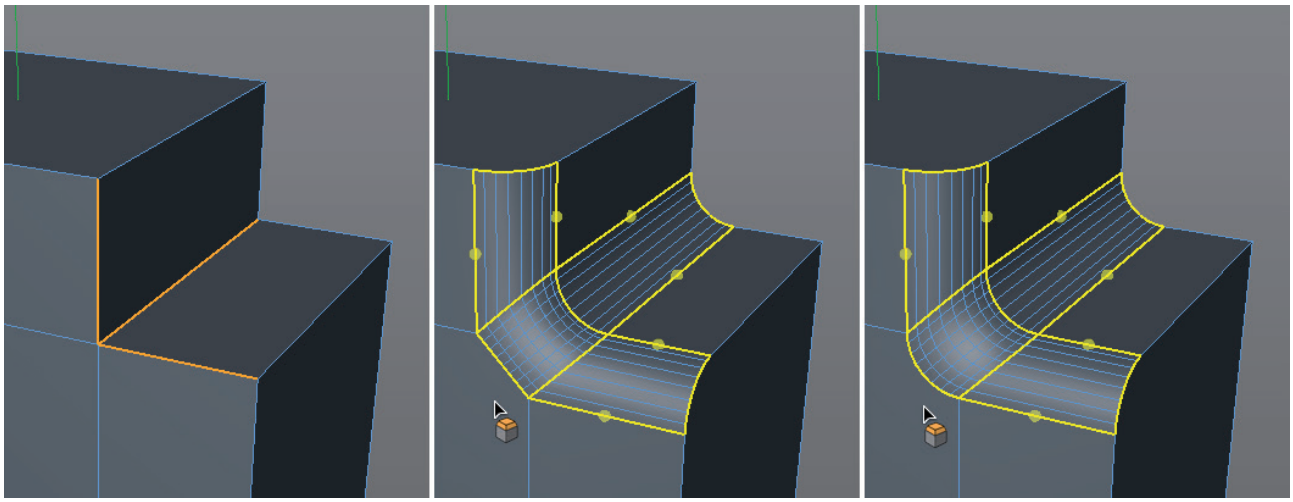
Only the **Uniform** option will create N-gons within these neighboring polygons. Miters and transitions from rounded to non-rounded regions can easily be recognized in the Viewport because the **Bevel** tool will highlight them in orange. The rounding itself will be colored yellow.

The **Ending** options define the type of transition from selected edges to neighboring edges.



If the **Default** option is selected, the length of the transition will be dependent on the **Offset** value. If the Extend option is selected, the entire length of the neighboring edge will be used for the transition from the rounded to the non-rounded region. Without a transition, the rounded region will look added on or imbedded.

The **Inset** option is only designed for use if three rounded and two unselected edges meet at a single point.



The following options define if and where N-gons should be created in the roundings. The **Rounding N-gons** option affects the rounded edges and **Corner N-gons** affects the regions in which multiple rounded edges meet at a single point. Because N-gons also consist of triangles and quads internally, which we cannot directly affect, you should avoid creating N-gons at corners so the quality of the rounding in these regions stays predictable.

The remaining two options affect the Phong angles that can be evaluated by Phong tags if the Break Phong Shading option is enabled. Edges that lie along a miter or rounding can then be excluded from Phong shading, which will result in a noticeably hard surface shading along this edge.

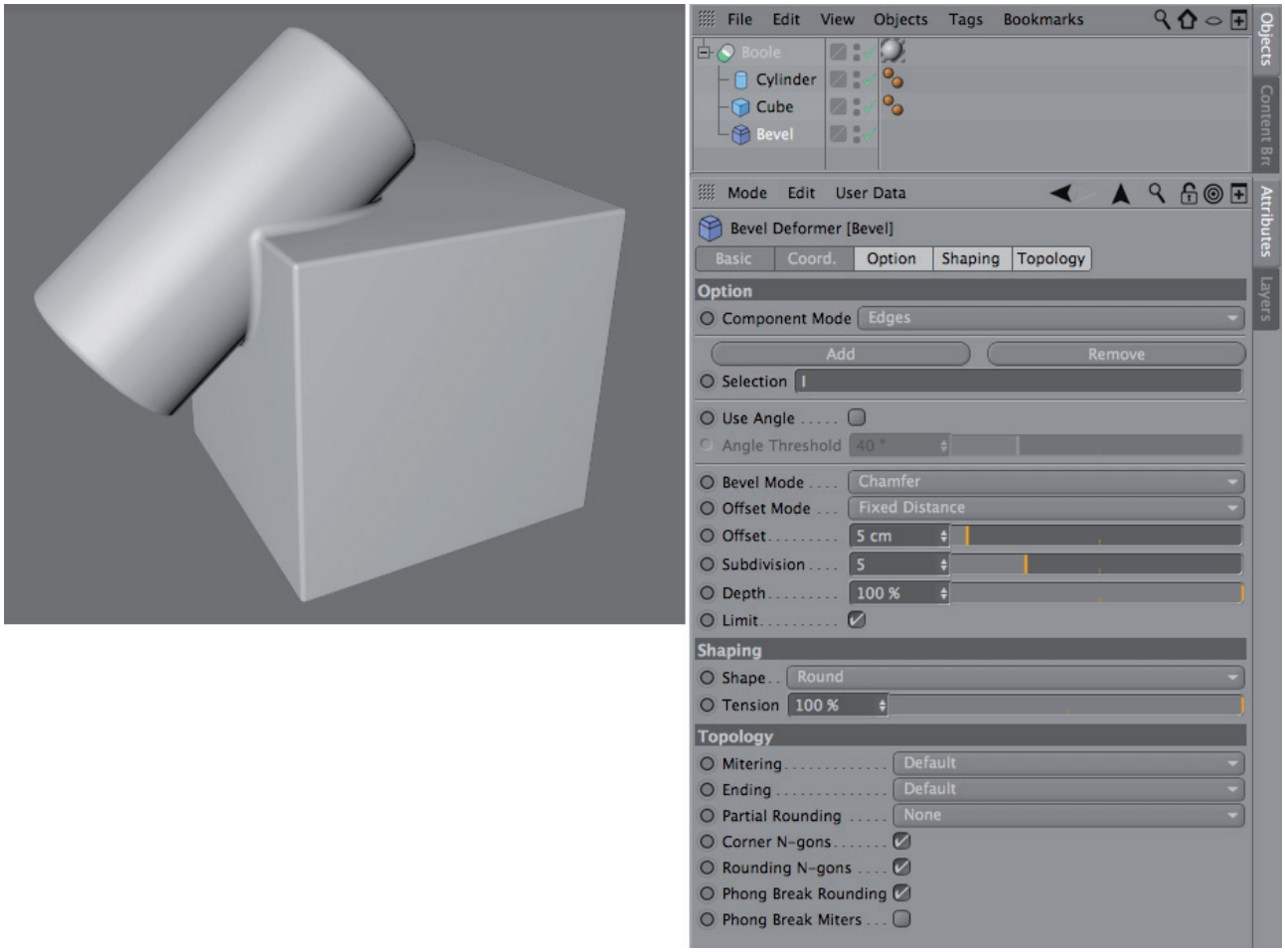
This tool can be applied directly in the Viewport. However, fine-tuning has to be done using the settings in the ***Attribute Manager*** that were just described. Hotkeys are also available that can make options available directly in Viewport that are not available in the ***Attribute Manager***.

Normally, you will first select points, edges or polygons in the Viewport that you want to round. If polygons are selected, the **Bevel** tool's effect will be a combination of the **Extrude Inner** and **Extrude** tools. The surfaces will be moved in the direction of the surface Normals and will form button-like structures or raised surfaces. If the **Extrude** tool has negative values, sunken surfaces will be created. The **Offset** value defines the scaling of the polygon in percent. In any event, most of the option's effect can be controlled using the yellow handles in the Viewport. This is one of the most important options, especially when rounding points or edges as long as no custom cross-section is used.

If multiple edges should be rounded simultaneously, you can **Cmd**/**Ctrl** + drag on individual handles after the rounding's radius has been defined. This way, the rounding of individual edges can be modified. Note, however, that changing the mode settings in the ***Attribute Manager*** will reset these individual modifications. Therefore, this type of modification should be done after all other settings are final.

5.4.6.5 The Bevel Deformer

Another practical tool is the **Bevel Deformer**. This object can, for example, be found in the **Create / Deformer** menu and must be made a Child object of the object whose points, edges or polygons it should affect.



You can define which elements should be beveled in the **Component Mode** menu. If the **Use Angle** option is enabled, the bevel can, for example, be restricted to the edges that lie at an angle larger than that defined in the **Angle Threshold** setting. The bevel can be made even more precisely using set selections. To do so, select the object edges you want to bevel, then select **Set Selection** from the main **Selection** menu and enter the name of the selection in the Deformer object's **Selection** field. This also works with parametric primitives. This is done as follows:

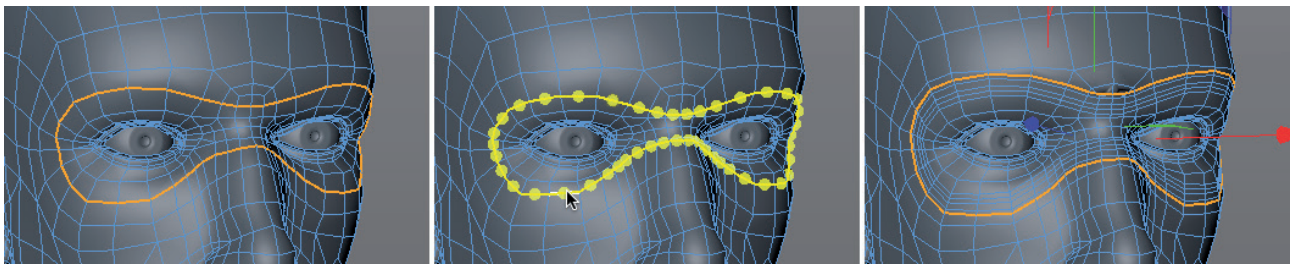
Create a copy of the parametric object and convert it to a polygon object (**Mesh / Convert / Make Editable**; or press the **c** key). Set the desired selections and drag the respective Selection tags onto the original object that is a Child object of the **Bevel Deformer**. The converted polygon object can then be deleted.

The **Boole** object can also create a hidden selection that can then be smoothed using the **Bevel Deformer**. To do so, the **Create Single Object** and **Select intersections** options must be enabled in the **Boole** object's menu. You will get better-quality results if the **Hide new edges** option is enabled, which will generate N-gons. This hidden selection will be named '**I**' (capital 'i') and can be used as the selection name in the **Bevel Deformer** after it has been placed as third element in the **Boole** object's hierarchy.

5.4.6.6 The Slide Tool

Points or edges are not always where we need them to be. Subsequently moving them always carries the risk of inadvertently modifying the object's shape. It would be great if the surrounding edges can be used as a type of guide along which these elements can be moved. This would restrict the movement of the points and edges.

This is exactly what the **Slide** tool does. To simultaneously move multiple edges, they must be selected first.



Points can generally only be moved individually. If **Ctrl**/**Cmd** is pressed while the **Slide** tool is implemented, the selected point will automatically merge with the neighboring point in whose direction it was moved when the mouse button is released. This behavior is similar to that of the **Stitch and Sew** tool, which will be explained below.

Similar to the **Bevel** tool, the **Slide** tool also offers different **Offset** options, when in **Use Edge** mode. If set to **Fixed Distance**, all edges will be moved with equal offset values. If set to **Proportional**, the offset will be a percentage relative to the neighboring edge lengths. Enabling the **Limit** option will prevent the sliding element from overlapping neighboring points.

Furthermore, the **Shift** setting can be used to define the vertical distance from the surface.

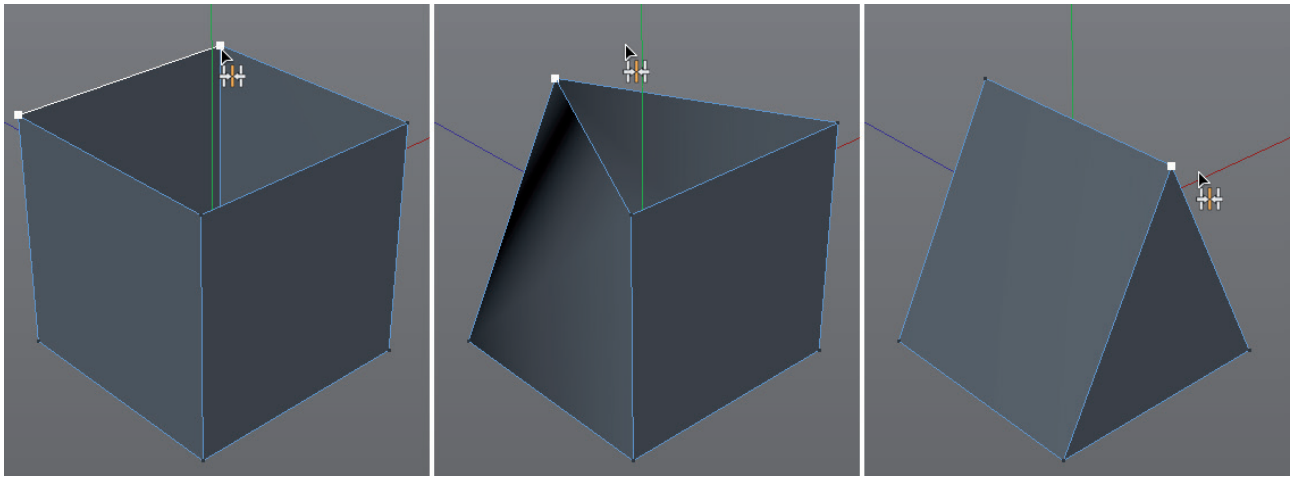
If **Ctrl**/**Cmd** is pressed while an edge is being slid, the original edge will be copied and the copied edge will be moved. The tool's **Clone** option has the same effect.

Enabling the **Preserve Curvature** option will draw a hidden circular curve through the original edge points and the adjacent edges. The sliding movement will then follow this curvature and not the adjacent edges. When used in combination with the **Clone** option and multiple **New Transform** instances, curvatures can be created that are similar to what can be created using the **Bevel** tool. The advantage of using this method is that a rounding can be restricted to one side of a given edge.

5.4.6.7 The Stitch and Sew Tool

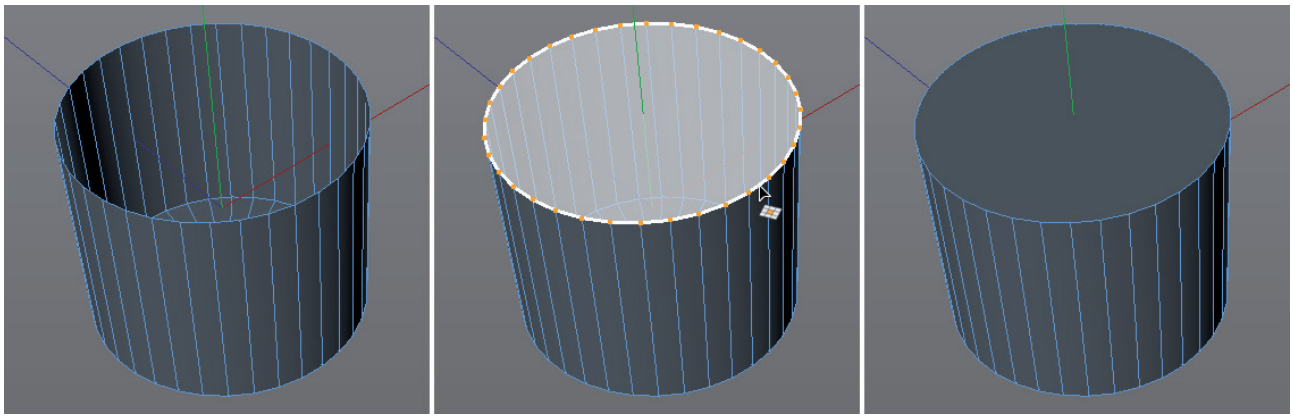
This tool is not used to add details. It is used to merge existing points. For example, the open ends of a structure can be closed or superfluous points can be eliminated by merging them with neighboring points.

To get a predictable result, no points or edges should be selected before activating this tool. Points or edges can then be grabbed directly and dragged to neighboring points or edges to which they will merge. If **Ctrl/Cmd** is pressed simultaneously, the dragged element and the target element will snap to their mathematical center when the mouse button is released.



5.4.6.8 The Close Polygon Hole Tool

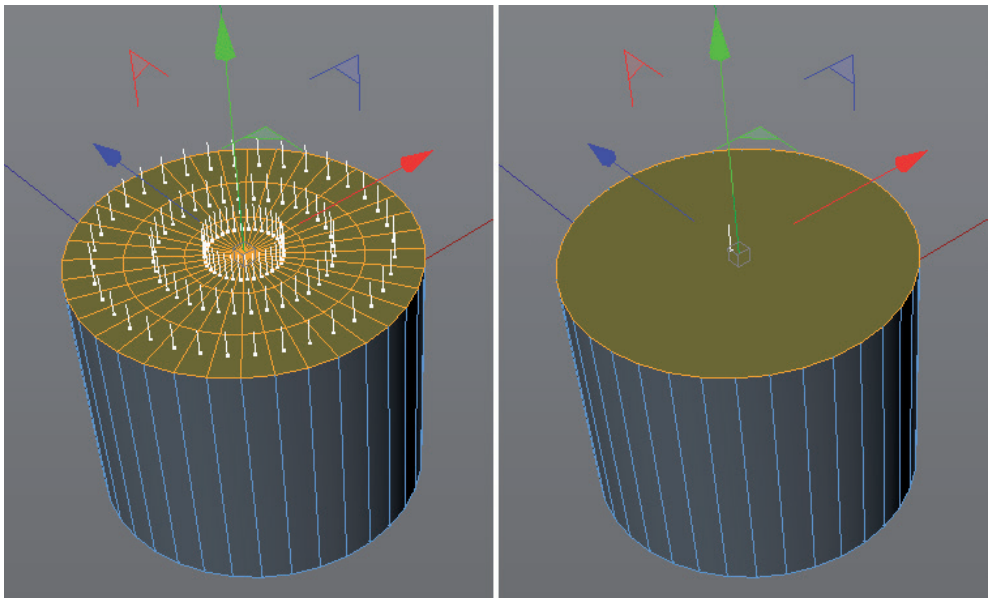
We have already discussed tools that are used to create individual polygons. However, if you want to close a polygon hole, this tool has one deciding advantage: only a single mouse click is required. As soon as the cursor lies over a polygon hole, a bright preview surface will appear. To confirm, simply click with the left mouse button to close the hole.



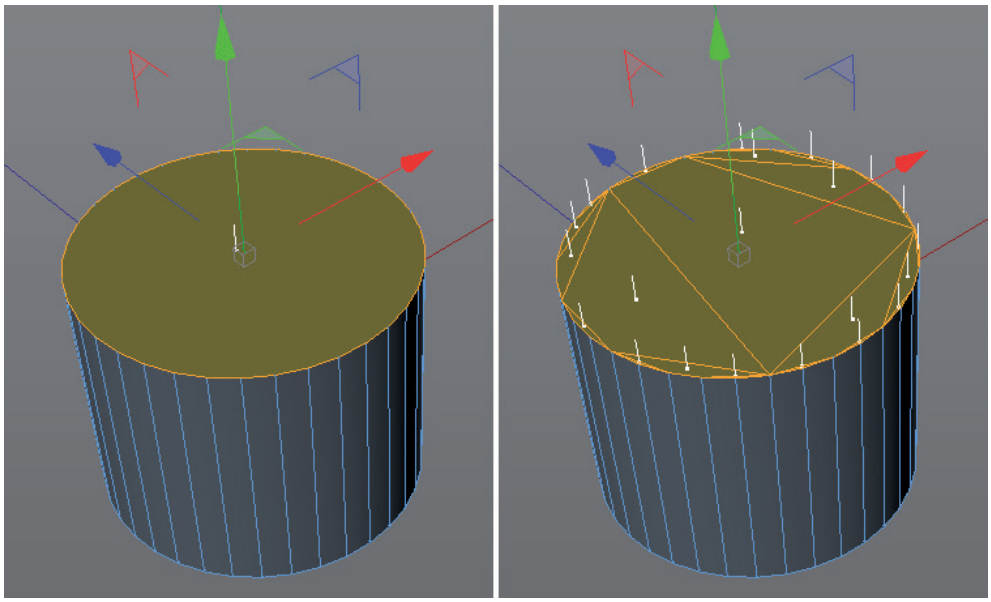
A single N-gon surface will be created, which means that the hole to be closed should lie on a 2D plane, if possible.

5.4.6.9 Melting and Removing N-gons

N-gons can be very useful for restricting the number of visible polygons. This is why several tools offer the option of generating these special types of polygons automatically. The **Melt** command can also be used to subsequently combine selected polygons to an N-gon. However, the points within the selected group of polygons will be lost. Principally, what happens is that the selected surfaces are deleted and the original outer edges are closed by an N-gon. This command also works in **Use Point** and **Use Edge** mode in which it will delete the selected elements without creating holes. In any case, this command should only be applied to regions of geometry that are flat to avoid modifying the object's shape.



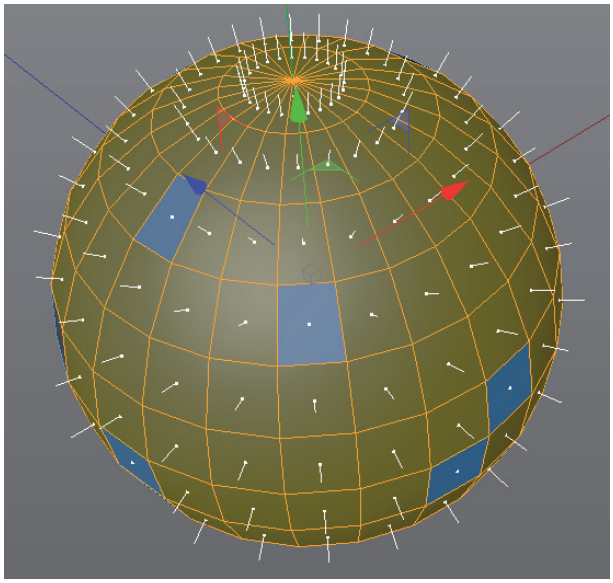
The **Dissolve** command does the opposite of the **Melt** command. It turns N-gon regions into triangles and quads. However, because N-gons themselves to not define the number and arrangement of these surfaces, melting and subsequently dissolving will, as a rule, not restore the surface to its original state.



If no N-gons are selected, all N-gons on the selected object will be transformed.

5.4.6.10 Rotate, Move, Scale Normals

We've already seen Normals in the form of white lines in the Viewport if the Polygon **Normals** option is enabled in the Viewport's **Configure** menu. They should always stand perpendicular to the outer side of an object's surface – and this is the case for all parametric Primitives and spline modeled objects. However, if polygons are created manually, it can occur that the Normals on a given surface point in the exact opposite direction. The object's shape is not affected but the behavior of many tools as well as the object's appearance when lit or textured can be dramatically affected. These types of errors should be corrected as quickly as possible. To do so, select the surface in question and select the **Reverse Normals** command from the **Mesh** menu. Surfaces with incorrectly oriented Normals can also be recognized in the Viewport by their blue (back side/inner) color. A polygon's front/outer side is orange.



If multiple polygons have reversed surfaces, the **Align Normals** command can be used. Cinema 4D will analyze the Normals and correct them automatically. This should be done in Use Model mode or make sure that no polygons are selected. Otherwise only the currently selected surfaces will be analyzed.

Note that **Normal Tags** will often be imported when objects from CAD applications are imported. These save the orientation of the Normals for all surface vertices so the surface shading doesn't have to be dependent on the angle at which neighboring polygons lie to one another. If the **Rotate User Normals** and **Move User Normals** options are enabled, information saved in these tags will also be updated.

5.4.6.11 Moving, Scaling, Rotating Along Normals

If, for example, an **Extrude** command was used to move a polygon, it will be difficult to subsequently correct this polygon's vertical distance from the surface. In such instances, moving along the direction of the Normals can help. This command can be controlled interactively via the **Move** value in the ***Attribute Manager***. If no selection has been made, even an entire object can be uniformly enlarged or made smaller. This can, for example, be very helpful when creating a hull around an object. All Normals must be aligned uniformly.

The same method can be used to rotate or scale polygons around their Normals. If multiple surfaces are rotated simultaneously, the result can often be unpredictable.

5.4.6.12 Split and Disconnect Commands

If polygons have been selected, they can be moved separately without affecting surrounding regions by selecting **Disconnect** command (Mesh menu). Selecting this command will double the number of points at the edge of the selection, which means that all surfaces will remain part of a single object.

Use the **Split** command to transfer the selected polygons to a new object. The originally selected polygons will remain at their original location. This function works like the **Copy** and **Paste** commands.

5.4.6.13 Connecting Objects

The **Connect Objects** does the opposite of what the **Disconnect** command does. First, select all objects whose points and polygons you want to connect to a single object. Then select the **Connect Objects** or **Connect Objects + Delete** command. This also works with spline objects but the different interpolation types and intermediate point settings will be lost.

If the **Connect Objects** command is used, the original objects will remain unchanged. If **Connect Objects + Delete** is used, the original objects will be deleted. Only the newly created combined object will remain.

5.4.6.14 The Optimize Command

Especially when importing objects from other programs it's not always certain that an object does not have any double structures. Cinema 4D objects can also benefit from optimization. This command checks an object's selected regions for collapsed surfaces or unused points and can also merge points automatically if they are at nearly identical locations.

Collapsed surfaces are created when all corner points lie on a straight line. The surface is valid but will not be rendered. Unused points are often created when polygons are deleted. The deleted surfaces' corner points are not automatically deleted. And Cinema 4D itself uses several objects that contain double points, e.g., the cylinder primitive. If this object is then converted to a polygon object (made editable), the edge between the Caps surfaces and the cylinder's walls will be double. Also, commands such as **Disconnect** can cause double points to be created that are not always visible. The **Optimize** command has its own menu, which includes a **Tolerance** value for finding double points. If an entire object should be optimized, switch to **Use Model** mode or make sure that no part of the object is selected when calling up the **Optimize** command.

▶ *See: Exercises for modeling tools*

SUMMARY: MODELING TOOLS

- Existing points or edges can be connected using the **Bridge** tool.
- The **Bridge** tool can also be used to create polygon tunnels or connections between selected polygons. This is done interactively in the Viewport using the mouse.
- The **Extrude** tool can be used to extend edges or surfaces. An object's original shape will be extended and enlarged.
- The **Extrude Inner** tool can be used to increase the number of polygons on a given surface, and can also be used to scale a selected region.
- The **Bevel** tool offers several options for rounding edges or points. When beveling polygons, a combination of **Extrude** and **Extrude Inner** is used.
- The **Bevel Deformer** offers the same functionality and can also be used with parametric objects and its effect can be edited at any time.
- The **Slide** tool can be used to move points or edges along the adjacent edge. Additional options make it possible to create rounding, similar to the **Bevel** tool.
- Points or edges can be merged using the **Stitch and Sew** tool.
- The **Close Polygon Hole** closes holes in surfaces using N-gons.
- Selected polygons can be turned into N-gon using the **Melt** tool.
- The Remove N-gons command converts N-gons to triangles and quads.
- A surface's Normals should generally point outwards. Incorrectly oriented Normals can be re-aligned using the **Reverse Normals** command.
- The **Align Normals** command can be used to automatically check and correct orientation.
- Polygons can be moved, rotated or scaled relative to the direction of the Normals.
- The **Split** command copies selected polygons and uses them to create a new polygon object. The original object will remain unchanged.
- The Disconnect command doubles the number of points at the edge of a polygon selection and makes it possible to move or scale the disconnected surfaces separately without affecting neighboring structures.
- Separate objects can be combined to a single object using the **Connect Objects** command. Applying the Connect Objects + Delete command will delete the originally selected objects.
- Double points can be merged using the **Optimize** command.

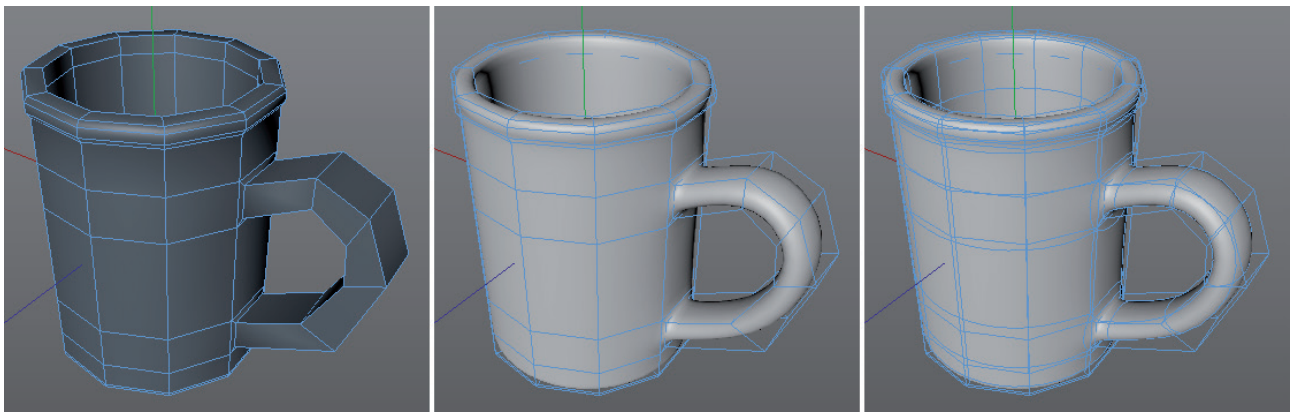
5.4.7 The Subdivision Surface Object

At this stage we've already gotten to know all important modeling techniques. Splines make it easier to create organic shapes as long as these shapes don't have to subsequently be distorted. Polygon modeling offers total freedom for arranging surfaces but quickly becomes confusing and complex when creating organic shapes.

This disadvantage is compensated by the **Subdivision Surface** object. This is a generator, which generally has polygon objects as sub-objects. The **Subdivision Surface** object gives the polygon object a higher subdivision and also gives it a more rounded shape. This effect reflects the previously described Subdivision function, which also has a **Subdivision** option. The advantage here, though, is that the subordinate polygon object itself is not subdivided and that this subdivision can be adjusted at any time via the **Subdivision Surface** object or even removed entirely. This is an interactive process that does not modify the original object.

This means that the polygon object can be edited and that the subdivided and smoothed version of the **Subdivision Surface** object will constantly be updated. Hence, two versions of the polygon object exist: a normal, generally roughly modeled shape and a smoothed version of the polygon object.

Various display types are available in the Viewport's **Options / Configure** menu, depending on your own preferences for the modeling process.



In this menu's **Display** tab in the *Attribute Manager* you will see the **Isoline Editing** option. If this option is disabled, an object's actual shape will always be displayed when in **Use Point**, **Use Edge** or **Use Polygon** mode when a polygon object is selected that is a sub-object of a **Subdivision Surface** object. The smoothed and subdivided geometry will be displayed independently of this.

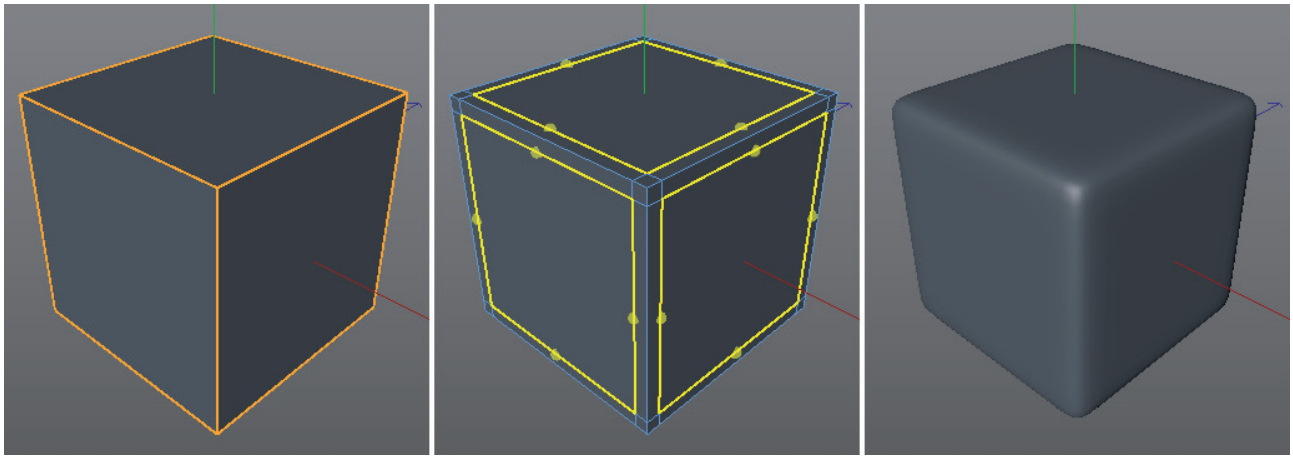
If **Isoline Editing** is enabled, the object's original shape will not be displayed and you can edit the points, edges or surfaces of rounded **Subdivision Surface** shape directly. This mode gives a good impression of the modified shape but don't forget that the original shape still exists and serves as a basis for the **Subdivision Surface** geometry.

The intensity of the Subdivision Surface object's subdivision can be defined separately for Viewport display and for subsequent display for rendering, i.e., for stills or animations. Don't be fooled by the low values. Each time the value is increased by a single unit, the number of polygons on the smoothed object will be quadrupled (approximately). Increasing the value will make the shape correspondingly more organic but will also increase render time accordingly.

It's difficult to recommend the best setting because the effect varies greatly depending on the sub-object's pre-existing number of polygons. When modeling the polygon object, note that adding **Subdivision Surface** will make the object a little smaller.

Technically speaking, the polygon object's edges will be used as tangents on the smoothed shape. The shorter a polygon object's edge is, the smaller the rounding will be on the **Subdivision Surface** shape. This means that the interval between points on the polygon object can be used to control the strength of rounding.

As you might remember, the **Bevel** tool's **Partial Rounding**/**Full** option can be used to create parallel edge loops. This function can now be used to directly control the **Subdivision Surface** object's rounding.



Of course, you can also add similar subdivisions using the **Extrude Inner**, the various **Knife** tools or **Edge Cut** tools.

When working with **Subdivision Surfaces**, you should generally only model polygons using quads. This ensures the most homogenous result for the smoothed geometry. Triangles will also be rounded by the subdivision but will create a different polygon pattern, which will be visible on the **Subdivision Surface** object, among others. For the same reason, you should avoid using N-gons because they can also contain triangles. If the use of n-gons or triangles can't be avoided, try to only use these in areas where no rounding takes place.

5.4.7.1 Type Settings

The Subdivision Surface object can be calculated using one of several methods, which can be selected from the **Type** menu. The **Catmull-Clark (N-gons)** option is optimized for objects that contain N-gons in addition to the standard triangles and quads. However, N-gons that have openings or have extreme curvatures can produce flawed results. If the **Catmull-Clark** option is selected, N-gons will also be supported but they will be converted internally to triangles and quads prior to the smoothing. This mode can produce better results for problematic N-gons. This option also uses the algorithm that is used by numerous other programs, which makes it easier to exchange these objects with these applications.

Since there are differences amongst various applications with regard to their standard Catmull-Clark algorithms, a publicly accessible library was published whose modes carry the moniker **ObenSubdiv**. Some of these modes don't really differ from the results produced by Cinema 4D but they do offer special functions only offered by OpenSubdiv. **OpenSubdiv Bilinear**, for example, does not smooth any sub-ordinate objects and only increases their subdivision. This can be useful when working with mechanical objects whose angled shapes should be preserved but still require a higher subdivision for deformations.

OpenSubdiv Loop is also a special mode that can be used to automatically break down existing quads into triangles. This is useful if an object needs to be made up exclusively of triangles for subsequent editing.

OpenSubdiv Catmull-Clark reflects the standard behavior described above and is almost identical to Cinema 4D's own **Catmull-Clark** algorithm. Differences only become apparent when working with weighting. Weighting is explained in more detail in a later section in this curriculum.

The **OpenSubdiv Catmull-Clark (Adaptive)** mode is particularly useful for higher levels of subdivision on objects used for interactive viewing in the Viewport. This mode offers the additional **Adaptive (GPU) Tessellation Level** setting that can be used in addition to the **Subdivision Editor** value to calculate additional subdivision levels via the graphics card. In many cases this is faster than using the CPU. This additional subdivision is done in critical regions of the object only and will not affect other surfaces on the object. When rendered, only the **Subdivision Renderer** setting will be applied. This mode will automatically switch to **OpenSubdiv Catmull-Clark** mode for rendering, which can result in slight deviations for the surface smoothing.

Deformations will not affect the additional GPU subdivision. For example, if a character is animated, the original object will first be deformed and this state will be smoothed and subdivided by the GPU. This mode can only be used if **Enhanced OpenGL** is enabled in the Viewport.

Many of the OpenSubdiv modes offer additional options such as the smoothing of triangles or edges of geometry (boundaries). Boundaries are outer edges of surfaces such as windows or planes. The **Boundary Interpolation** option **Edge and Corner** makes it possible to disable the rounding for the four corners of a plane, for example, so the object's shape is preserved in spite of other sections being smoothed. The **Boundary Interpolation** reflects the typical behavior in Cinema 4D.

The **Triangle Subdivision** setting can be used to smooth the region between neighboring triangles and quads. The **Catmark** option reflects the typical Cinema 4D behavior. The **Soft** option can be used to smooth banding in these regions.

The **Edge Crease** setting affects the calculation of edge weighting for adjoining edges with different weighting. The **Chalkin** option offers the most harmonious results compared to the **Uniform** option.

5.4.7.2 Subdivision Surface Object's Special Features

Subdivision Surface Weights can be added to the points, edges and polygons of the subordinate polygon object. To do so, select the desired element using the **Live Selection** tool and take a look at the selection tool's **Subdivision Surface** menu in the *Attribute Manager*.

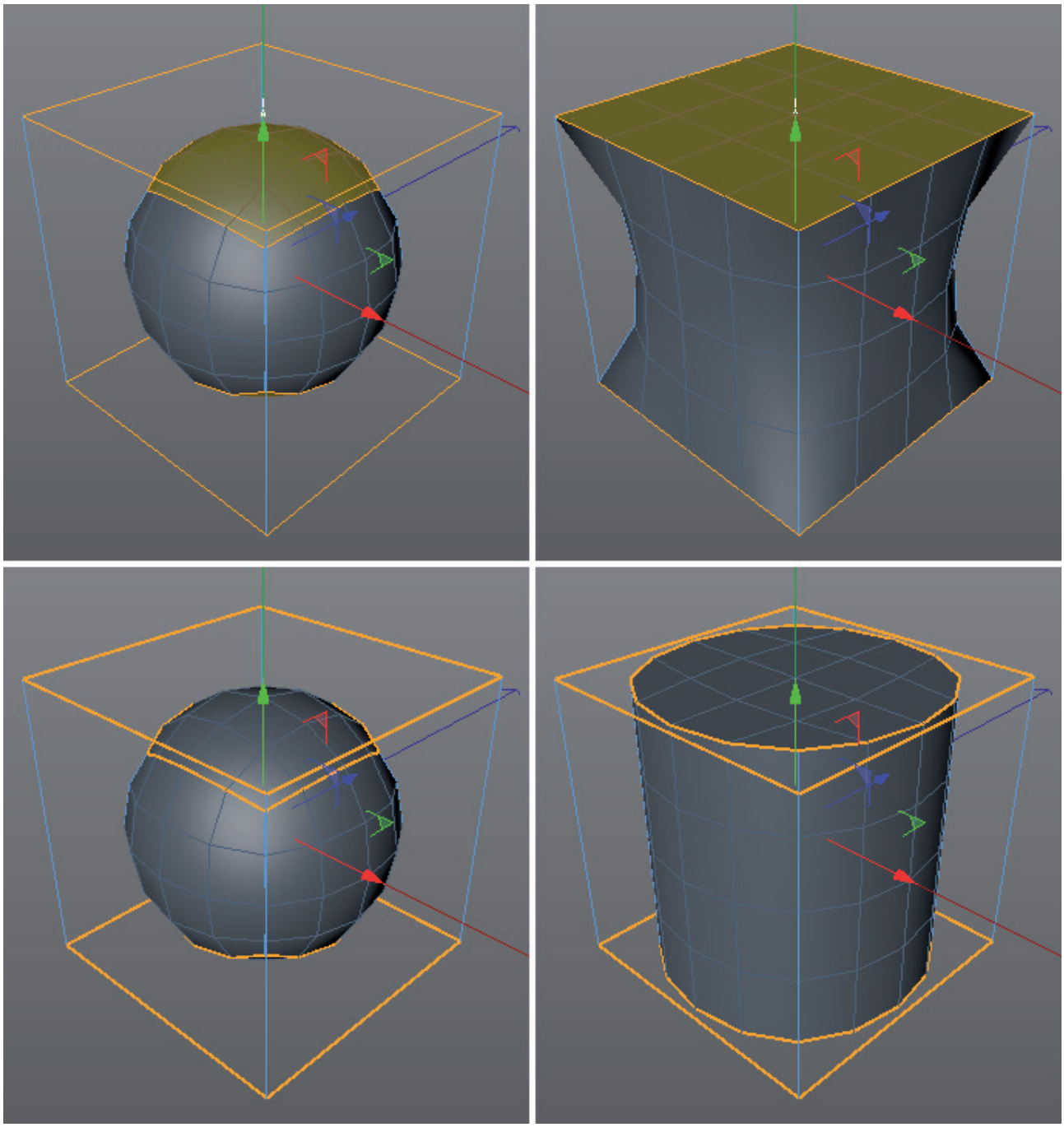
The **Strength** value can be set to a value between -100% and +100%. Depending on the **Mode** option selected, this value can be assigned to selected points, edges or polygons absolutely (**Set** option) when the **Set** button is clicked, or the value can be subtracted from or added to existing weighting (**Add**/**Sub** option, respectively).

Points and edges can be weighted separately. When polygons are weighted, all points and edges will also be weighted accordingly.

The percentage weighting of points results in a direct attraction or repelled of the **Subdivision Surface** shape at that location. Only weighting points will therefore result in a series of zigzag spikes.

If only edges are weighted, the Subdivision Surface shape will be attracted or repelled in the direction of the edge's center.

If all four edges of a quad are weighted with 100%, the **Subdivision Surface** geometry will contract so strongly in this region that circles or ellipses can result. If in addition the edge points are also weighted with 100% or if the entire polygon was weighted, a hard structure in the shape of the polygon will result.



All of these effects have no effect on the polygon object's original shape. All weighting information is saved in a separate tag next to the polygon object (in the ***Object Manager***). Selecting the **Subdivision Surface Weight** tag will cause the corresponding **Subdivision Surface** to be displayed in a different color in the Viewport.

Red regions have weighting not equal to null. All other regions will be colored blue.

If this tag is deleted, the original **Subdivision Surface** shape will be restored.

Because a Subdivision Surface object can also subdivide and round multiple polygon objects simultaneously, as long as they are sub-objects of the top polygon object, the **SDS** tag also has a second function that lets the subdivision be defined individually.

If the **Change Subdivision** option is enabled, separate values can be defined for **Subdivision Editor** and **Subdivision Renderer** values. This option is completely independent of the saving of weighting. This means that the **SDS** tag can also be called up directly over the ***Object Manager's*** **Tags**/**Modeling tags** menu.

If you don't need exact weighting values, the weighting can be added interactively without using the Live Selection tool. To do so, select the respective points, edges or polygons on the polygon object and press the period key ('.') while you click and drag left or right in the Viewport. The value that can be defined ranges between the **Interactive Minimum** and **Interactive Maximum** values, which can also be found in the Live Selection tool's **Subdivision Surface** menu. If Interactive Minimum is set to -100%, negative weighting can be created when doing so interactively.

Even though affecting Subdivision Surfaces in this way is very practical, adding more edges where a stronger attraction is needed offers even more control over rounding. The repel effect is only possible using weighting.

▶ *See: Exercises for Subdivision Surfaces*

SUMMARY: SUBDIVISION SURFACES

- The **Subdivision Surfaces** object rounds the subordinate polygon object by interactively adding subdivisions.
- The various **Type** settings offer a wide range of subdivision options, including the use of GPU for calculating subdivision.
- Objects within the polygon object's hierarchy will also be smoothed.
- Individual subdivision levels for the Viewport and for rendering can be defined for the Subdivision Surface object directly.
- **SDS** tags can be used to define different subdivisions to different sub-objects.
- The Viewport's Configure menu can be used to define the display type for polygon objects when editing points, edges or polygons.
- The scale of rounding on a Subdivision Surface object can be influenced indirectly via the sub-object's edge length.
- The Live Selection tool can be used to add individual weighting to points and edges that will result in an attraction or repelling of the Subdivision Surface shape at the respective element's location.
- This weighting can also be created interactively by pressing the period key and clicking + dragging in the Viewport.
- Weighting is saved in the **SDS** tag. Deleting this tag will also remove all weighting for the object.
- The original polygon object can be removed from the Subdivision Surface object at any time to modify rounding.

5.5 Deformations

Deformations are used non-destructively in Cinema 4D, which means that they can be edited at any time or disabled entirely. A deformed object's original state can be restored at any time.

In a standard deformation, an object's points are moved and the attached surfaces move accordingly, which results in a deformation of the object. As a result, soft deformations such as bends or twists look good when an object has a sufficient number of points. Also, the points should be positioned as uniformly as possible across the surface.

Some deformations that are often specific to a particular animation go a step further and can also change an object's structure. For example, the **Explode** deformer can be broken used to break down an object into its individual parts (surfaces).

Most of the **Deformer** objects can be used for animation but **Deformer** objects can also be used for modeling.

An object's deformed shape can, for example, be frozen and converted to a correspondingly shaped polygon object, which can in turn be modified. The deformation, however, cannot be edited interactively or animated. Therefore, it's generally better to maintain the Generator object as long as possible, similar to how the **Subdivision Surface** object or the various spline modeling objects (**Extrude**, **Lathe**, **Loft**, **Sweep**) are maintained.

The difference between these objects is, however, that deformations must generally be grouped under the object to be deformed.

This bears the advantage that multiple deformations can be used to affect a single object.

Groups can be created if multiple objects should be deformed at the same time. The deformers will not directly be made sub-objects of the object to be deformed but will all lie at the same hierarchical level, e.g., beneath a **Null** object.

Each Deformer object has its own settings that can be modified in the ***Attribute Manager***. Some offer additional handles in the Viewport that can be used interactively.

The most commonly used Deformer objects are **Bend**, **Bulge**, **Squash** and **Stretch**, **Shear** and **Twist**.

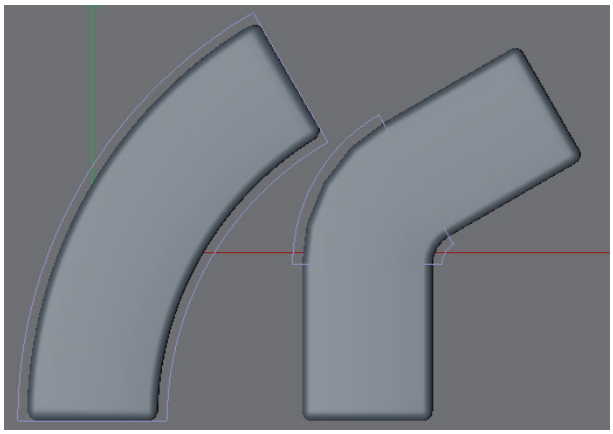
5.5.1 Fine-Tuning Deformation Regions

The first thing that should be considered before an object is deformed is which regions on the object should be affected. A deformation doesn't always affect the entire object. Cinema 4D offers a hotkey for many Deformer objects for just this purpose. First, select the object to be deformed then press the **Shift** key when you select the Deformer object. This will not only automatically make the Deformer object a sub-object, but will also rotate and scale the Deformer to include the entire object and orient it along the Y axis of the object to be deformed. This can be seen on the **Bend**, **Bulge**, **Squash** and **Stretch**, **Shear** and **Twist** objects by their purple bounding box display around the entire Parent object.

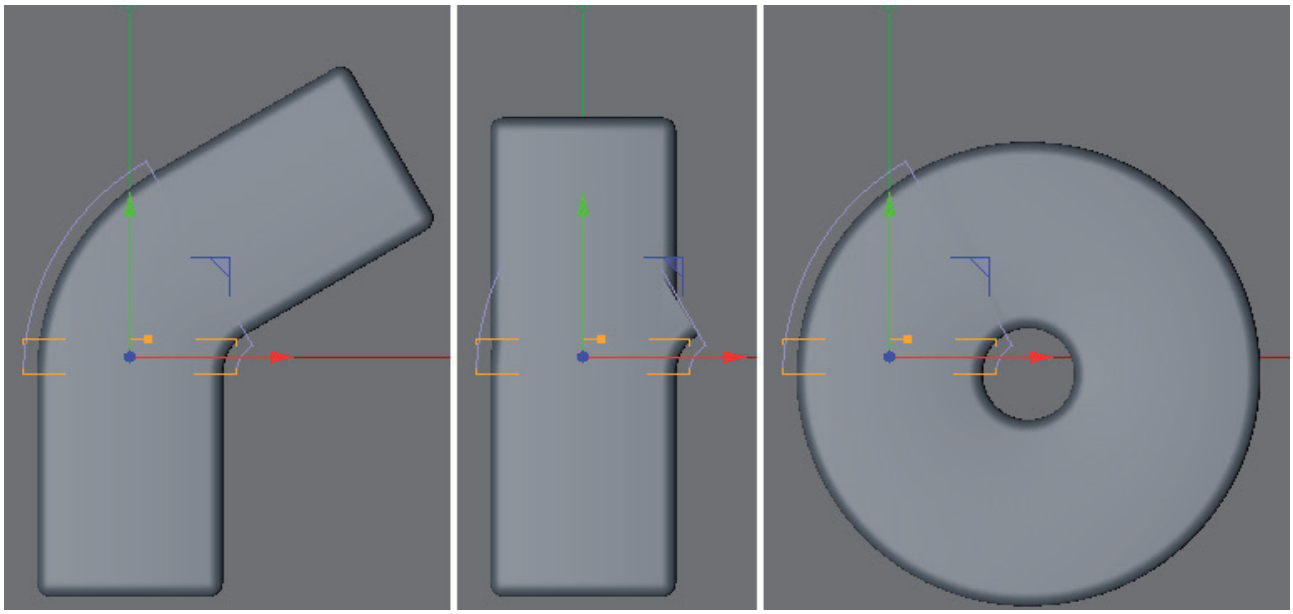
If the Deformer object's Y axis, and thus also the handle, should be arranged along a different axis of the object to be deformed, rotate the Deformer object roughly in the corresponding direction and click on the **Fit to Parent** button in the ***Attribute Manager***.

Moving the handle in **Use Model** mode will, depending on the Deformer object, affect the direction or strength of the deformation effect.

Generally speaking, the size of the purple bounding box should be adjusted to fit the size of the region to be deformed. This is done using the **Size** values in the *Attribute Manager*.



The bounding box may also have to be moved along its Y axis to make it fit properly. The deformation's effect also depends on the **Mode** option selected.

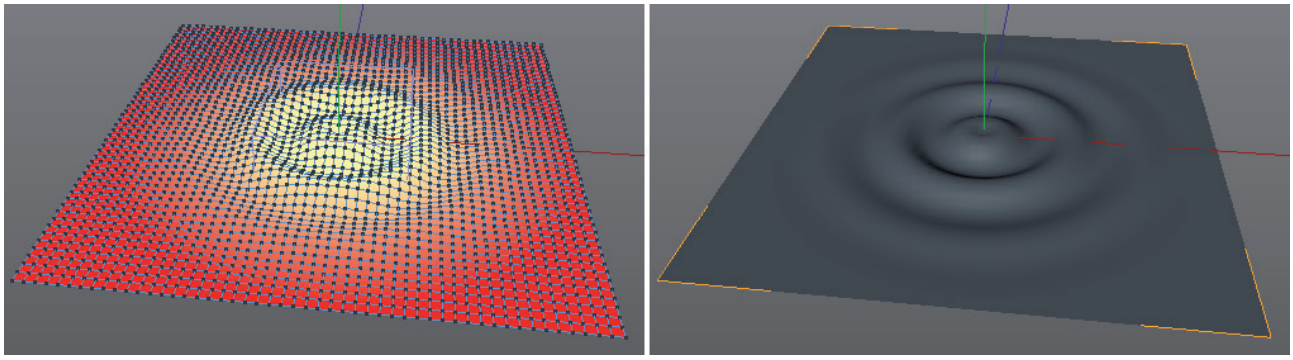


If the **Limited** option is selected, the deformation will start at the floor of the purple bounding box that lies across from the handle. The deformation will then end at the end of the handle. All regions of the object to be deformed that lie above and below the bounding box will not be deformed. However, these regions will be adapted accordingly. For example, a rectangle with a sufficient level of subdivision that is bent to the side will be straight at the top but nevertheless slightly angled in the direction of the bend.

If **Within Box** is selected, the deformation will only the points that lie within the purple bounding box will be moved. All other regions of the deformed object will remain unaffected. If Unlimited is selected, the deformation will extend beyond the bounding box and affect the entire object.

Deformations can also be restricted using Fields. Respective settings can be found in the **Falloff** tab. Only very few Deformers such as **Bevel**, **Spline Wrap** and **Spline Rail** don't offer these options. Fields can take on different shapes like cube, sphere or cylinder and the deformations can be restricted to parts of the objects that lie in these volumes. More complex Field types also allow the use of particles, selections, textures or splines and can be animated automatically. Fields can also be used by Selection tags, vertex maps or Effectors within MoGraph.

As long as you're working with a simple polygon object that needs to be deformed, another option is available that we already discussed, **Vertex Maps**. These can, for example, be created using the Live Selection or Bruch tool. In conjunction with Deformer objects, Vertex Maps can be used to calculate the deformation as a percentage relative to the vertex weighting of each surface point. Points with a Vertex Map weighting of 0% will, for example, remain unchanged.



The **Restriction** tag, which can be found in the ***Object Manager's*** menu in the **Rigging** tag group, is used to create a connection between the deformation and the Vertex Map. Assign this tag to the Deformer object and drag all Vertex Maps tags that are created into this tag's corresponding table.

5.5.2 The Most important Deformation Objects

Many deformations are very specific and designed to do certain things during an animation. Some can be used in a wide variety of ways and are suited both modeling and animation. We will discuss a few examples below. Other Deformer objects can also be discussed briefly, if necessary.

5.5.2.1 The Bend Deformer

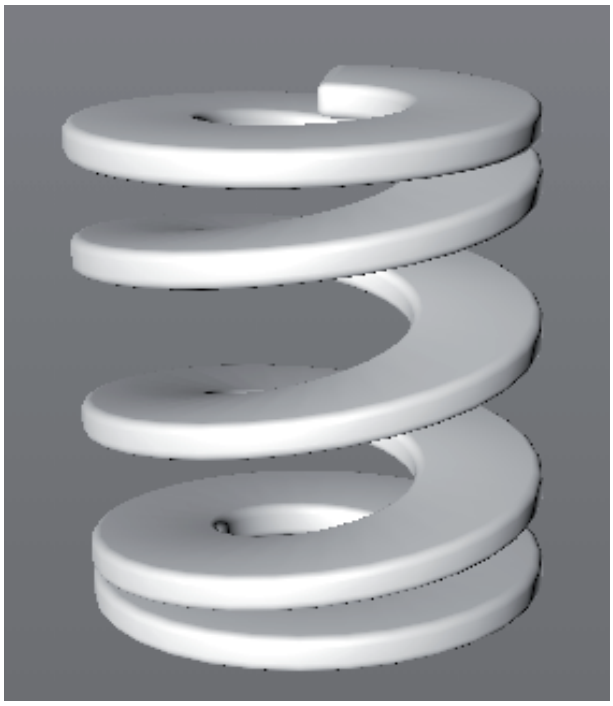
This is the most commonly used deformer object, which is used to bend objects in any direction. Objects can even be bent into circles. Often, the object seems to be elongated, which is due to the points being pulled apart on the outside of the bend radius. This effect can be limited by enabling the **Keep Y-Axis Length** option.

5.5.2.2 The Bulge Deformer

This object can be used to supplement constrictions or enlargements. The **Curvature** value can be used to define constrictions. A value of 0% will create a linear transition; larger values up to 100% will result in correspondingly more rounding. Values in excess of 100% will exaggerate the curvature and can result in multiple protrusions being created. The **Fillet** option only affects the transition between the deformed and non-deformed parts of the object. If the option is enabled, an organic, rounded transition will be created.

5.5.2.3 The Shear Deformer

This Deformer object moves the ends of the deformed region against each other. The standalone effect looks quite simple but can, for example, be used in conjunction with the **Bend** deformer to create interesting new shapes. The example from the **20_BendDeformer** example project, for example, shows a bent cube that is deformed into a helix spring using a **Shear** deformer.

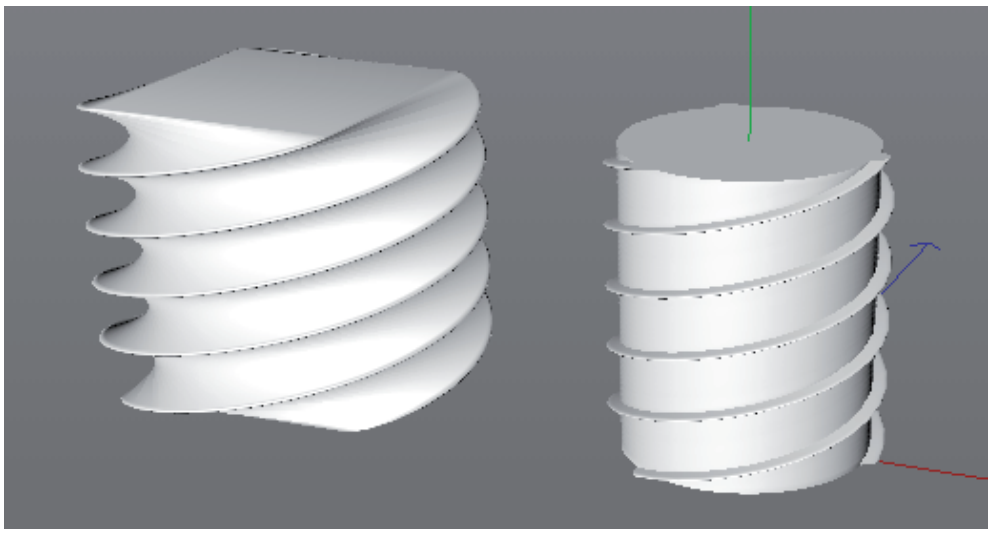


5.5.2.4 The Squash and Stretch Deformer

Similar to the **Bulge** deformer, this object can also be used to generate constrictions or enlargements. However, these will only be calculated in one direction, which means that the constriction will not appear on the upper part of the deformed region. Otherwise the settings are very similar to those of the **Bulge** object.

5.5.2.5 The Twist Deformer

This Deformer twists objects around its Y axis. This makes it easy, for example, to create screw threads using a Cube. The **Project 21_TwistDeformer** shows an example of this.



5.5.2.6 Possible Combinations

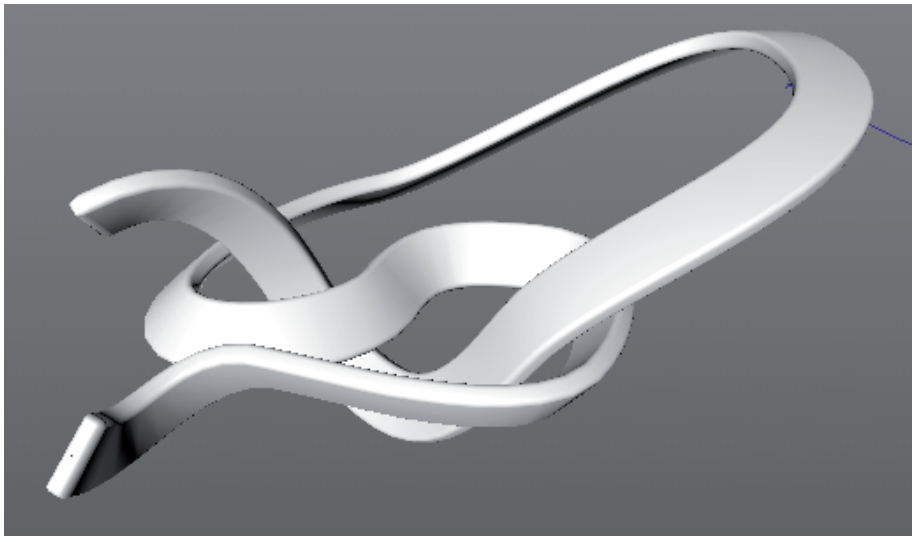
Always keep in mind that all deformations can also be mixed. In doing so, the order in which they lie in the ***Object Manager's*** hierarchy plays an important role. The Deformers will always be applied from top to bottom. For this reason, the **Bend** deformer should always lie at the end of the hierarchy. Otherwise the object's deformed regions can end up protruding outside of the bounding box, which can produce unwanted results.

Note also that the **Limited** mode can also affect the deformation on the sides, and therewith outside of the bounding box. Therefore, there are limits, for example, to the number of times an object can be bent. The Project **22_BendDeformer2** shows such an example, in which a narrow cube was bent four times at 90°. The last **Bend** object extends too far into the previous bend, which can result in a faulty shape. What can help in such instances is restrictions in Vertex Map or Field regions, rearranging the objects or using the Within Box mode (see Project **23_BendDeformer3**).

Even more effective in such cases, however, is the use of the next Deformer:

5.5.2.7 The Spline Wrap Deformer

If the shape you want to create is very complex, using a spline object can make it easier to create. The spline can be subsequently applied to objects as a deformation in conjunction with the Spline Wrap (see Project **24_SplineBend**).



First, the spline must be linked to the **Spline Wrap** via drag and drop into the **Spline Wrap** object's **Spline** field. The **Axis** setting directly below must also be set to the right option. It defines the direction of the world axis. Select a direction that corresponds to the orientation of the object being deformed. For example, if you've modeled an arrow that points along the world X axis in the Front view, set **Axis** to **+X**. If the object stands vertically pointing upward, **+Y** would be the correct option.

The **Fit Spline** option will stretch the object along the entire length of the spline. The **Keep Length** option will try to maintain the object's original size. If the spline is longer than the deformed object, the Offset slider can be used to control the object's position along the spline.

The object's rotation on the spline can generally be adjusted using the **Banking** value, which is located in the **Rotation** menu. In this regard, using a **Rail** spline would be more flexible. It works according to the same principle as the **Sweep** object. In the **Spline** menu you will find the corresponding **Rail** field into which a spline can be dragged.

5.5.3 Freezing Deformations

If you want to freeze a deformation in the geometry, i.e., you're sure that the deformation is final and no more changes will be made, select the object and then the **Current State to Object** command in the **Mesh/Conversion** menu. A new, simplified polygon object will be created that has the exact shape of the deformed object – but no longer needs a Deformer object to maintain its shape.

▶ *See: Exercises for Deformers*

SUMMARY: DEFORMERS

- As a rule, the deformation quality depends on the number of points on the surface to be deformed and their dispersion across the surface.
- Generally speaking, Deformer objects are made sub-objects of the object they deform.
- If the **Shift** key is pressed when a Deformer object is selected from the menu it will automatically be made a sub-object of the currently active object and its bounding box will also be fitted accordingly.
- If a Deformer object's Y axis should follow a different direction on the object being deformed, rotate the Deformer roughly in the corresponding direction and press the **Fit to Parent** button in the ***Attribute Manager***.
- If multiple objects should be deformed simultaneously they must be grouped at the same hierarchical level as the Deformation object(s) or within a **Null** object.
- The region to be deformed can be affected using the Deformer object's **Falloff** setting or **Mode** options.
- **Restriction** tags containing **Vertex Maps** or Selection tags can also be assigned to Deformers in the ***Object Manager***.
- Multiple Deformation objects can be combined. The order in which they lie in the ***Object Manager's*** hierarchy is important. The top-most Deformer will be applied first and all other in the order in which they lie.
- Deformations can be frozen into the geometry. To do so, select the object to be deformed and select **Mesh/Conversion/Current State to Object**.

5.6 Modeling Objects and Help Functions

This section first describes the most important **Modeling** objects. These object to not generate any geometry themselves and take effect after another object has been made a sub-object of the Modeling object. In the second part of this section, helpful tools for duplicating or randomly transforming objects will be described.

5.6.1 The Modeling Objects

The majority of these objects are located in the **Create / Generator** menu or in the top icon menu of the Subdivision Surface object. The OpenVDB functions can be accessed via the **Volume** menu in Cinema 4D or in the respective icon menu. For most of these objects, their effect can first be seen after another object or object group has been made a sub-object of the Modeling object. An exception to this is the Instance object, which uses a **Reference Object** field to link to an object. The Volume Builder can also access objects via links that aren't directly subordinated. The advantage of using these objects is that the original is not modified and can be restored by disabling the **Modeling** object, for example. These objects are Generators that basically work like **Subdivision Surface** objects.

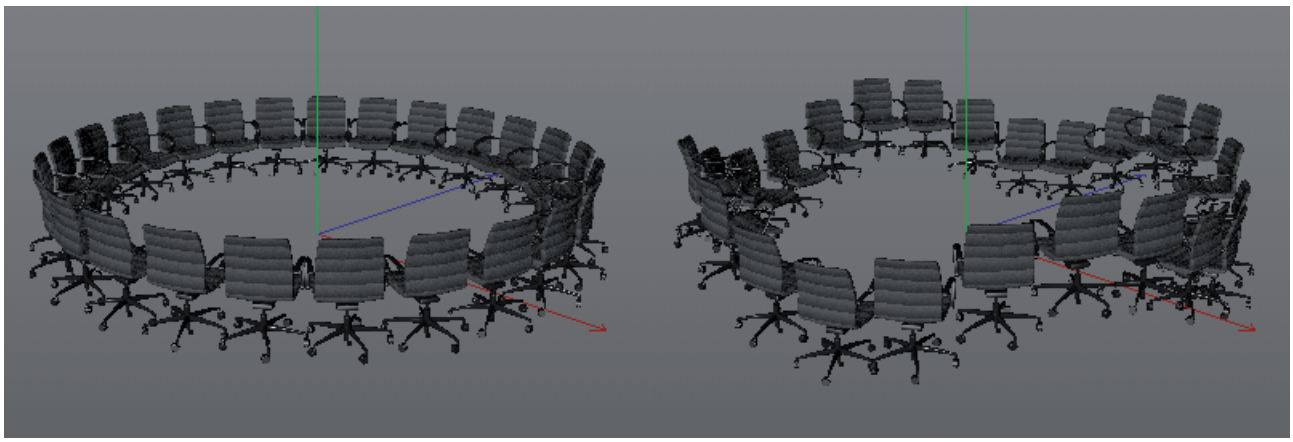
All Modeling objects, with the exception of the **Volume Builder**, can be made editable by pressing the **C** key or using the **Make Editable** icon at the top of the left icon palette to make its points, edges and polygons accessible for editing. However, this does not work if Render Instances are used. This will be discussed shortly.

Note also that several Modeling objects can also be stacked in complex hierarchies, which is common when using the **Boole** objects.

5.6.1.1 The Array Object

This object can be used to circularly arrange object copies. The sub-object will be duplicated in accordance with the **Copies** value and positioned in a circle around the **Array** object's Y axis. The original object's Z axis as well as that of the copies will be oriented outwards.

The Amplitude setting can be used to create a wavy arrangement of copies on the **Array** object's XZ plane.



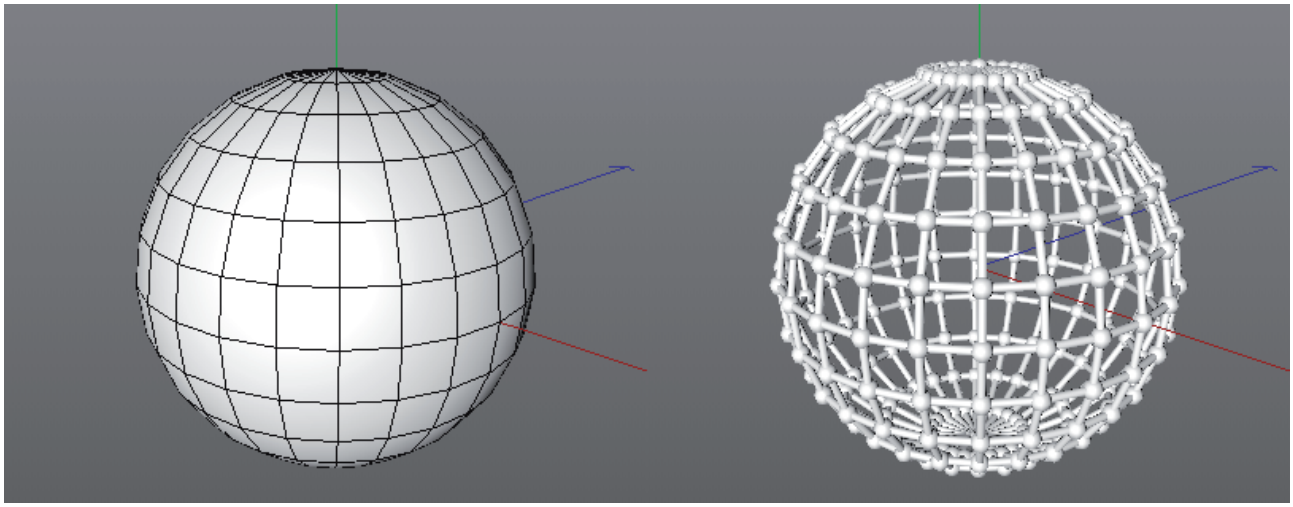
The **Amplitude** value defines the maximum deviance in the arrangement along the **Array** object's Y axis. The **Array Frequency** value defines how often the copies can oscillate on the maximum and minimum amplitude. The **Frequency** value is only important when animating. Values greater than 0 will result in a movement of the **Array** copies in the circular motion.

Because it's so easy to create a high number of copies using this object, the amount of memory required for complex objects can increase quite quickly. In such cases, enabling the **Render Instances** option will help. This will optimize the copies so only the original is maintained for rendering. This only works if all objects can remain identical and only deviate with regard to position, size, direction of movement and maybe material. Instances cannot be modified individually because they don't have their own points or polygons. More information about instances will be made available when the **Instance** object is discussed.

The **Array** object can, for example, be used to duplicate chairs of one type around a table. If you want to have more control over the placement of object copies, take a look at the **MoGraph Cloner** object.

5.6.1.2 The Atom Array Object

This **Modeling** object can be used to make the inner polygon structure of a sub-object visible. The object's points will be replaced by spheres and its edges will be replaced by cylinders.

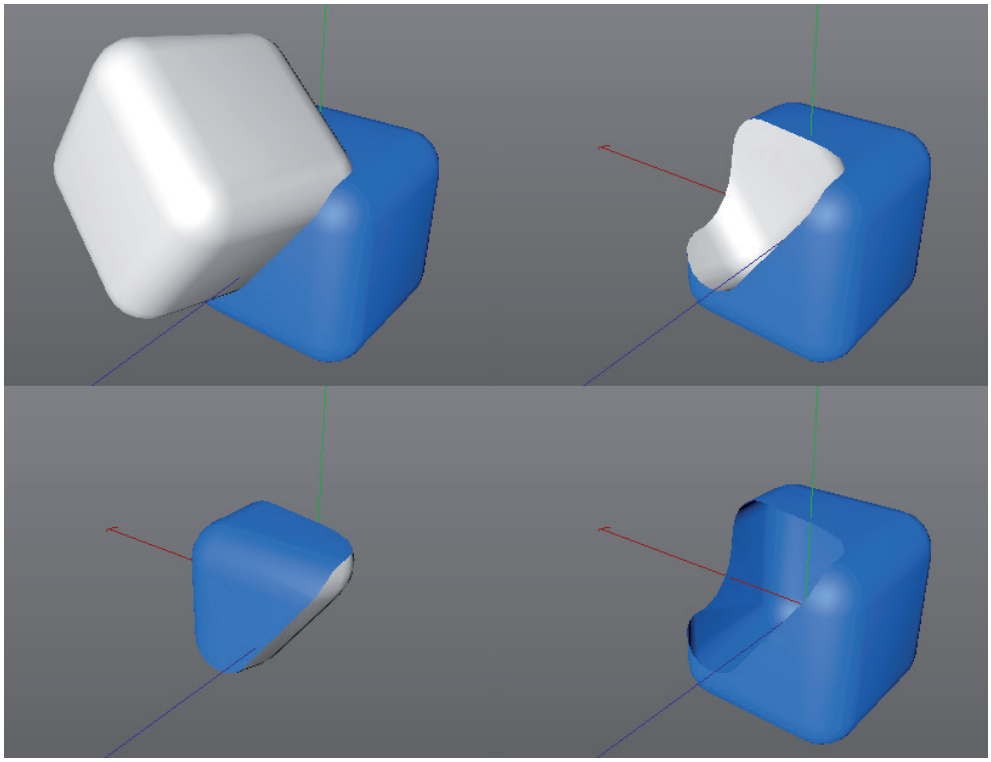


The original object's display will be replaced by a type of grid. The radii of the spheres and cylinders can be defined separately, whereby the radius of the spheres should never be less than that of the cylinders. If the **Atom Array** object should be made editable, the **Single Elements** option can be used to convert each sphere and cylinder individually. Otherwise they will be converted together as a single object.

The **Atom Array** object can, for example, be used to give objects a wireframe look or, for example, to create a simple traverse structure, which is commonly seen in trade show or stage construction.

5.6.1.3 The Boole Object

This object is very useful for combining objects. Objects can be combined, subtracted from one another, or the intersection of two objects can be created.



This object always requires at least two sub-objects that intersect with each other. At least one of these objects must be a closed volume (such as a sphere or a cube).

The order in which the objects are arranged in the **Boole** object's hierarchy is also important. The top object will be considered object **A** and the second object will be considered object **B**. Additional objects can be arranged below object **B** as long as these don't touch or overlap.

How objects **A** and **B** will interact is defined in the **Boole** object's **Boolean Type** setting. **A subtract B**, for example will subtract the part of **B** that intersects or overlaps **A** will be subtracted from **A**. This is an easy way to create holes, notches or cutouts.

Other modes can be used to add or combine **A** and **B**. If **A without B** is selected, the common areas of the **A** and **B** intersection as well as **A** will be deleted.

Enabling the **High Quality** option will activate an algorithm that will generally create a better result and a cleaner arrangement of edges. However, in some cases, disabling this option can produce better results or even make the operation possible if the algorithm can't do so.

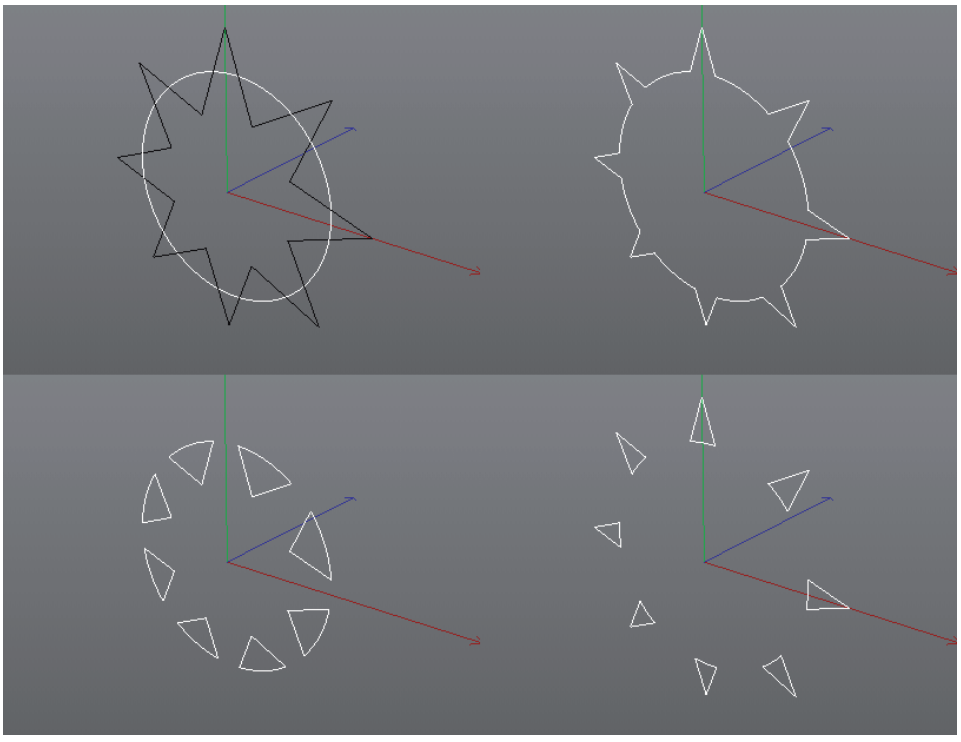
If **Create single object** is enabled, the resulting shape will be optimized automatically. Double points will be merged using the **Optimize Points** value, which defines the maximum distance within which points should be merged.

Enabling the **Hide new edges** option will activate the creation of N-gons on the overlapping objects, which makes the resulting geometry look cleaner. Enabling the **Create Phong breaks at intersections** option makes the transition between both shapes more pronounced by breaking the shading in these regions. The resulting shape will look like two conjoined shapes. The **Create single object** option must also be enabled. Otherwise all parts of the object will be shaded separately. Enabling the **Select intersections** option will automatically select the edges of the converted **Boole** object that lie at the edge of the overlapping **A** and **B** objects. This selection can be used prior to converting the **Boole** object because it carries the name '**I**' (capital 'i'). This makes it possible, for example, to apply a **Bevel Deformer's** bevel to a **Boole** object that has not been made editable. However, the **Create Single Object** option must be enabled and you should also enable the **Hide new edges** option.

Again, a major advantage of using the Modeling objects is that the original objects remain unaffected. You can exchange, edit or move the sub-objects at any time to achieve a different result. And you're not restricted to using polygon objects – you can also use spline objects or Primitives.

5.6.1.4 The Spline Mask Object

This object is much like the Boole object but only works with two-dimensional splines.



This means that the splines used should, for example, lie entirely on the XY, ZY or XZ plane. In addition, the splines' depths must all be equal. If both splines lie on the world XY plane, for example, there can be no difference in the Z coordinates of any of the points. Splines with tangents should also be checked to make sure that the tangents lie on the same plane as the splines.

The **Mode** menu is pretty much the same as that of the the **Spline Boole** function that was already discussed in the **Spline/Boole Commands** section.

The Axis menu's option should reflect the plane on which the splines lie. The **Spline Mask** can also be used to **Create Caps** surfaces, i.e., fill the resulting outlines with N-gons. The **Spline Mask** can also be used as a normal spline object, e.g., for creating geometry using the **Extrude** tool. More than two splines can be placed in a **Spline Mask**. For some modes such as **A subtract B** or **B subtract A**, the order in which the splines are arranged in the **Spline Mask's** hierarchy affects the result. Since the **Spline Mask** works interactively, the splines can be resorted or a new mode selected if the original result is not what was desired.

5.6.1.5 The Connect Object

This object works like the Optimize function but is interactive. The sub-objects are constantly checked for overlapping points if the **Weld** option is enabled. The **Tolerance** value defines the maximum distance between points that should be welded. When working with complex hierarchies, the top object in the hierarchy can be linked with the **Object** field, which means that the object does not have to actually be made a sub-object of the **Connect** object.

The **Phong Mode** defines how the shading on the resulting object should behave. Often, objects will have a different Phong setting before they're connected. If set to **Manual** you can manually assign a Phong tag to the **Connect** object and define the settings yourself. If **Average** is selected, a median value will be ascertained for the shading's edge angle based on all Phong tags. The **Lowest** and **Highest** options will use the Phong tag's smallest or largest edge angle, respectively, for the connected shape. Only the **As Breaks** option will calculate individual Phong shading breaks like the **Unbreak Phong Shading** command (**Mesh/Normals** menu) would do. If each object should keep its own material, leave the **Textures** option enabled. If disabled, you can assign a new texture to the **Connect** object that will cover the entire shape.

If **Center Axis** is active, the resulting geometry will be centered on the connect object. Otherwise the objects will stay in place.

The **Connect** function can, for example, be used to connect individual objects that you want to smooth as a single unit using a **Subdivision Surface** object.

5.6.1.6 The Instance Object

Instance are like copies of an object but bear the advantage that they are not completely independent of the original. If the shape of the original object is modified, all instances will be updated accordingly. The connection between instances and the original object is made via a **Reference Object** field. Objects do not need to be made sub-objects of the Instance object.

If the original object is already selected when the **Instance** object is selected, the original object will automatically be placed into the **Instance** object's **Reference Object** field. Otherwise the original object can simply be dragged and dropped from the ***Object Manager*** into the **Reference Object** field. The Instance object itself merely defines the position, the orientation and is a repository of sorts for the various settings used to create instances.

In instance contains all points, edges and surfaces of the original object, which means it can also be deformed individually using **Deformer** objects or have **Subdivision Surface** or spline modeling objects assigned to it.

If very many copies of a given object are needed, the **Render Instance** option can be enabled. This restricts the way in which instances can be used because they will no longer contain points, edges or polygons, but the advantage is that much less memory is required to work with and render the scene. If the only difference between the original object and the instances is position, scale and rotation, enabling Render Instances can be enabled without causing

any adverse effects. Individual materials can also be assigned to instances as long as the original object does not have a texture assigned to it.

Multi-Instances take this concept much farther. They can be used to create very large numbers of identical copies. Internally, the memory required will be kept to a minimum by combining all copies (instances) to a single object. In addition to the limitations for **render instances**, **multi-instances** can also not be used together with **dynamics** collisions but can therefore be used to create a great number of copies while maintaining performance and greatly minimizing memory requirements.

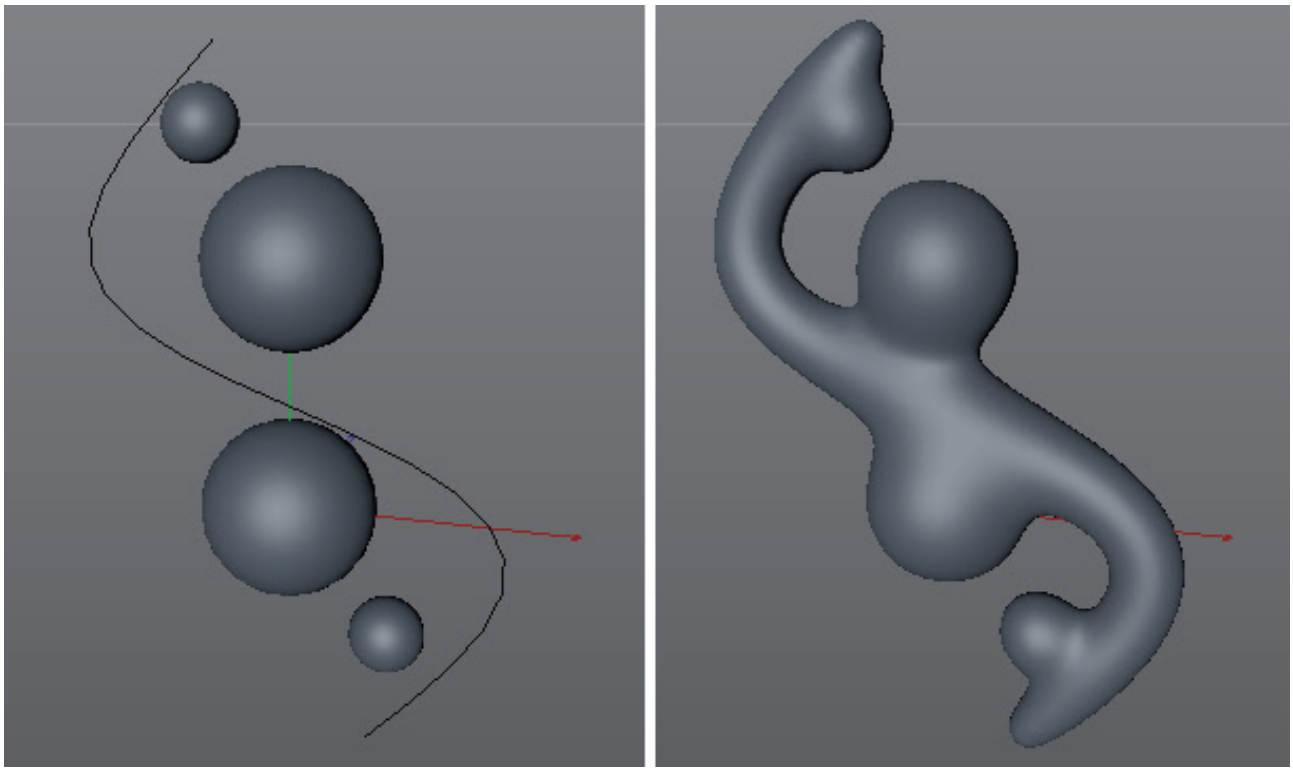
To define the copies from the count and position, a **Positions Source** must be defined in the **Multi-Instances mode**. This can be an **Emitter** or a **Thinking Particles Group**. A **MoGraph Matrix** object can also be linked here but a **MoGraph Cloner** object offers the same functionality and can therefore be used more flexibly since it offers additional options.

Another advantage of Multi-Instances is that you can switch between the duplicated objects' different display types in the **Viewport Mode**. Another option for duplicating objects is the **Duplicate** command, which we will describe later.

This can replace the use of individual **Instance** objects. The **MoGraph Cloner** object offers even more control over the number, animation and placing of copies.

5.6.1.7 The Metaball Object

This object is used to create spherical polygon shell around sub-objects. As a rule, Sphere objects or spline objects are used in conjunction with the Metaball object. If splines are used, a cylinder-like shape will be created along the spline's path. These shapes can flow into one another. Two spheres that lie close together will flow together when they are made sub-objects of a **Metaball** objects. The closer these spheres are moved together, the more they will connect until they form one large sphere. This behavior is similar to water drops or mercury drops that flow together. Any number of spheres or splines can be made sub-objects of a **Metaball** object. However, the number of polygons created will increase accordingly.



The size of the resulting shape depends, for example, on the spheres' radii and also on the **Metaball** object's **Hull Value** setting. Larger values will result in the effect adhering closer to the sub-objects' original shapes, i.e., tighter fitting hulls. Hence, the precision of display and with it the number of generated polygons can be defined separately for the Viewport and for rendering. The unit of measure for Editor Subdivision and Render Subdivision define the average

edge length of the new surface. The shorter the edges, the more points and surfaces that will be created, and the rounder the shape will be – with correspondingly more memory being required. In order to achieve good results even with low subdivision, the orientation of the Normals can be optimized by enabling the **Accurate Normals** option.

Enabling the Exponential Falloff option will cause the objects to connect at a later stage, i.e., when the distance between them is shorter. If this option is disabled, an inverse square calculation will be made, which produces a more gradual, organic transition.

5.6.1.7.1 The Metaball Tag

A **Metaball** tag can also be used to affect the shape of individual objects, e.g., a sphere, within a **Metaball**. This tag can be found in the *Object Manager's* **Tags / Modeling Tags** menu.

This tag contains a **Negative Influence** option. If enabled, the object will not connect with neighboring objects but will repel surrounding objects. This can, for example, be used to create indentations. The **Strength** value is a multiplier for the **Metaball** object's **Hull Value** setting. Larger **Hull Value** settings produce correspondingly tighter fitting hulls.

As already mentioned, splines can also be used in conjunction with the **Metaball** object. Also, any type of polygon object can be used. Their points (or a spline's intermediate points) are assigned a spherical radius that can be defined using the **Metaball** object's **Radius** value.

In order to better maintain the shape of a **Metaball** object's Child object without affecting the organic merging of neighboring shapes, assign a Metaball tag to the object and set **Type** to **Triangle**. If **Type** is set to **Line**, the object's edges can possibly be interpreted as cylinders, which can in turn merge with each other or with other objects. If the **Solid** option is enabled, the object will be rendered as a solid object. In combination with the **Negative** Influence option, subtractive Boolean operations can be simulated. This works especially will if **Type** is set to **Triangle**.

5.6.1.8 OpenVDB System

These functions can be used to fill objects or volumes evenly with voxels. Subsequently, a polygon hull can be placed over these voxels. The main advantage for modeling is that the number and arrangement of polygons of the referenced objects is no longer relevant. Voxels make it possible to create any complex shape or volume using simple cubes with constant edge lengths. Since the voxels' shapes are also subtracted or overlapping regions can be combined, this makes for a robust alternative to the **Boole** function, which doesn't always produce the desired result if complex objects are used.

A positive side-effect is that voxels can also be used to restructure Sculpt objects since sculpting works best on evenly subdivided surfaces.

The basic process is easy. Group the object you want to fill with voxels under a **Volume Builder**. Here you can also define the size of the Voxels and therewith the detail of the object display. Then make this a Child object of a **Volume Mesher** object, which generates the new polygons. The voxels themselves can only be displayed in the Viewport but not rendered there. The **Fog** mode in the OpenVDB system is designed to display volumes, in this case gaseous volume, as is required for simulating clouds or fire. No geometry is generated. These types of cloulds can, for example, be defined vis Fields. However, separate simulation systems can more often create more realistic results, which can then be imported vial the **Volume Loader**, which can be found in the main Cinema 4D **Volume** menu. Finally, the **Vector** mode can be used to calculate vector fields that can, for example, be used to affect the flight paths of particles or the direction of movement within dynamic simulations. These vectors can be evaluated via a **Field Force** object from the **Simulate / Forces** menu and included in a simulation.

5.6.1.8.1 Volume Builder

This object generates voxels within volumes or within an area of influence of the subordinated objects when the **Distance Field (SDF)** option is used. Slines or particles, for example, can also be assigned voxels if a volume around these elements is defined using a **Radius** value. All subordinated objects will appear in the **Objects** list in the **Volume Builder** and can be **unified** (Union), **subtracted** (Subtract) or **intersected** (Intersect). Only the bottom most object in the list will not offer these options since it is used as the basis for all calculations.

By selecting the objects in the list, additional settings can be made available, e.g., the **mesh points** can be used as volume cells that are scaled across the **mesh point radius**. The **Smooth Layer** can be used to round hard edges withing the voxel volume and to smooth transitions. The **Reshape Layer** can, for example, be used to increase the volume's size. This can, for example, be applied to particles to get more precise control over the convergence of particle volumes as the particles near each other.

Generally speaking, the **Smooth Layer** and the **Reshape Layer** only affect the underlying entries in the Objects list. The quality of the volume is primarily defined by the **Voxel Size** setting. Small voxels produce longer render times and a more detailed result on the respective objects.

Clicking on the small triangle next to this option will make additional options available, which let you optimize the number of voxels that is generated. Imagine the voxels as annual rings on a tree. If only the tree's bark should be depicted, the voxels on the inside of the tree trunk don't need to be generated. The **Interior Voxel Range** and **Exterior Voxel Range** settings are used to define the number of voxel layers on the inner volume and the outer bark. Increasing the number of **interior voxel layers** can, for example, be used to depict very rugged or perforated volumes or 3D Noise structures. Otherwise, the exterior voxel layers will be more important.

The **Spline Voxel Range** and **Particle Voxel Range** settings can be used to create hollow (sphere, tube, etc.) structures.

The **Volumetype** menu is used to define the type of calculation that will take place. If **Signed Distance Field (SDF)** is selected, an area filled with uniformly placed voxels will be created around polygons, points or splines. Depending on the overall number, especially of interior voxel layers, it can occur that the interior of the volume remains empty. This mode is well-suited for modeling and its Boolean functions offer new options for modeling organic or technical elements.

If **Fog** is selected, the settings for the voxel area to be filled are no longer available because the entire volume will always be filled when in this mode. This mode is therefore particularly well-suited for depicting flames, explosions or clouds, etc.

To display these types of effects in renderings, **ProRender** in combination with a special volume material must be used. The **Vector volume type** behaves similarly, where vectors will be created within the assigned volume, perpendicular to the splines. The orientation and length of the vectors can be modified using additional filters, Fields or a cross-product calculation with other vectors, for example. In the end, the Volume Builder can be assigned to a **Force** object and then, for example, be combined with particles or dynamic simulations.

5.6.1.8.2 Volume Mesher

This object can be used to create polygons if it is a Parent object of a **Volume Builder**. The **Voxel Range Threshold value** can be used to select the voxel layer that should be used to generate the polygons.

If you enable the **Use Absolute Value (ISO)** option, you can define the value that a voxel must have so it is evaluated for the generation of polygons. Depending on the value defined, it can happen that no polygons are created if no voxels with a corresponding value are found.

The **Adaptive** value can be used to reduce the density of polygons that are created. Finally, Vertex maps can also be created for the shape if the **Create Curvature Map** option is enabled. Depending on what is selected in the Curvature Direction setting, the regions curved inwardly or outwardly – or both – will be assigned weighting in the **Vertex Map**. This information can, for example, be evaluated via a **Vertex Map** shader in order to limit dirt or wear on the surface on raised or lowered surfaces.

5.6.1.9 The Symmetry Object

Many real-world shapes and objects are symmetrical. Symmetrical objects are created by modeling only one half of an object and simply making it a sub-object of a **Symmetry** object. All you have to do then is define the correct mirror plane. The mirror plane uses the **Symmetry** object's axis when mirroring elements. As long as an object's points lie on this plane, they can be welded with the mirrored half. The **Weld Points** option must be enabled. The **Tolerance** defines the radius within which points will be welded. If **Symmetrical** is enabled, the welded points will be positioned exactly on the mirror plane. Otherwise it can occur that a point on one half snaps to a point on the other half and therefore not lie at the center. Especially when **Brush**, **Magnet** or **Sculpt** tools are used to model an object in conjunction with the **Symmetry** object, points that lie on the symmetry plane can shift causing the cohesiveness between the original and mirrored shapes to be lost. Enabling the **Clamp Points on Axis** option will prevent this from occurring. If mirrored polygons that lie on the symmetry plane are extruded, polygons that lie entirely on the mirrored plane will be removed if the **Delete Polygons on Axis** option is enabled. Enabling the **Automatically Flip** option will ensure that an editable mesh is made available on the side of the **Symmetry** object that lies nearest the camera. Depending on the angle of view you can edit the shape of both halves of the entire object. Clicking on the **Flip** button will switch the original and the mirrored copy, which prevents you from having to move the camera itself.

5.6.1.10 LOD Object

Also known as the **Level of Detail object**, this object can automatically modify the visibility of objects, the quality with which they are displayed and even the number of surface polygons for objects. These modifications make it easier for the graphics card to process the scene, especially for complex scenes, and speeds up the display in the Viewport. Similar techniques are used in computer games to increase the frame rate and achieve a more fluid display of complex scenes.

As a rule, objects of which you want to display a reduced version depending on their distance from the camera, for example, should be made Child objects of the **LOD Object**. In the **LOD Object's** settings you then define which objects should be modified and how. Use the **LOD Mode** setting to define which objects should be affected. If **Children** is selected, the **LOD Object** must contain at least two different Child objects. The more complex and detailed objects must be positioned correspondingly higher in the hierarchy. A typical example would be for a high-resolution tree model at the very top, followed by a model that has the thinnest branches removed since these would not be visible at a distance. Next, a model of the tree with only a trunk and a textured polygon shell that depicts the leaves and so on.

If **Manual Groups** is selected, you define the number of steps, i.e., number of model versions for the reduction of detail, manually. Separate **Object** lists can be added via drag & drop. In each list, all objects must be listed that should be displayed simultaneously. The order of the objects in the **LOD Object** hierarchy no longer plays a role here.

If **Simplify** is selected, the object order will also not play a role. Here you can also use the **Level Count** setting to define the number of detail levels. These will apply to all Child objects equally.

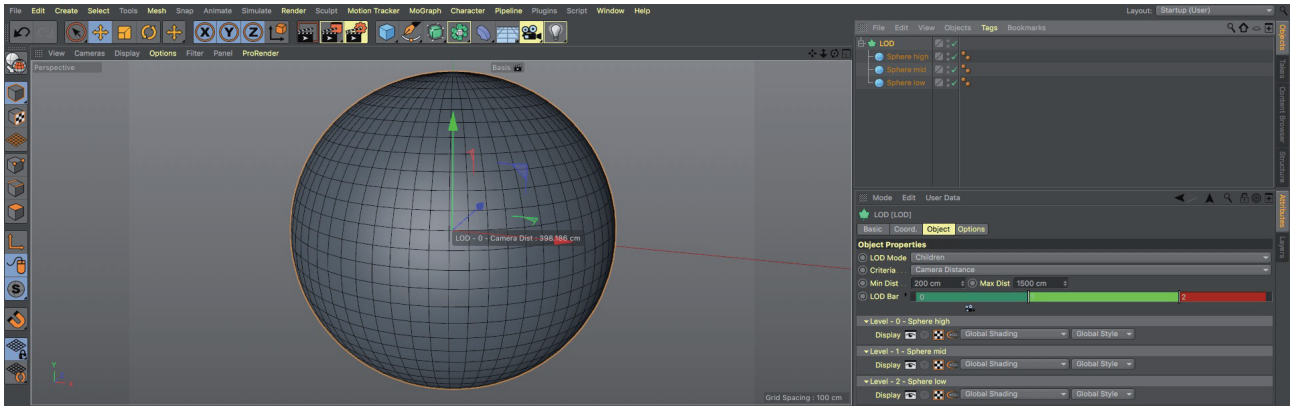
All three **LOD Mode** settings have in common that the display type and quality for the objects can be controlled using the menus and icons in the **LOD Object's** Display menu. The **Eye** icon can be used to hide or show the objects. The next icon can be used to switch the backface culling, i.e., the display of hidden polygons or lines. The third icon is used to control the display of colors and textures if the object has materials assigned to it. The corresponding menus can be used to define additional shading methods and line display types. The effect is paramount to adding and configuring **Display** tags.

5.6.1.10.1 LOD Criteria

After an **LOD Mode** has been selected, the criteria can be defined that determine how the objects' visibility or the general object display should be modified. If **User LOD Level** is selected, you can manually define which objects should be visible using the **Current Level** setting. The **User LOD Value** setting works similarly but uses a percent value for the modifications. A colored gradient uses vertical lines to show the value at which a switch will occur. These lines can be clicked on and dragged to a different position.

The **Screen Size H** and **Screen Size V** settings define the current display size of the object horizontally or vertically, respectfully, depending on the view size. A smaller circle below the **LOD Bar** setting shows the respective result. The vertical lines in the bar in turn show the location at which switches occur between objects and can also be clicked on and dragged to a different position.

The **Screen Surface** option works similarly with the difference that the entire screen surface in the view will be defined in relation to the object's screen surface.



The **Camera Distance** option is the most commonly used option since this is the technique used for game engines. Object levels are switched based on their distance from the camera.

You define the **Min Dist** and **Max Dist** manually. Here, fine-tuning is also done using the **LOD Bar**, which also offers a **camera** icon that can be used to adjust the distance of the currently active camera. However, after the mouse button is released, the camera will return to its original position. Nevertheless, this function is very useful for precisely setting the transitional vertices for the object switch. Vertical lines are also available here in the **LOD Bar**.

The last criterion is the **Global** option and refers exclusively to the three pre-defined levels of detail **Low**, **Medium** and **High**, which can be found in the view's **Options > Level of Detail** menu.

5.6.1.10.2 Simplified LOD Mode

If only the objects' visibility was modified in the **Children** and **Manual Groups** modes, a new look for the objects can be created using **Simplify Mode**. With this mode, additional simplified object versions don't have to be created.

If **Simplify Mode** is set to **Full Objects**, objects' shapes will be displayed unmodified.

If set to **Decimated**, polygons will be omitted accordingly. This omission can be defined as a percentage using the **Strength** value and will also work on procedural objects. The polygons will be modified according to their index numbers. A random omission or sorting, e.g., along an axis, will not take place but can be simulated under certain circumstances using the **Voronoi Fracture** function in MoGraph. An object's fragments can be sorted in a specific direction in 3D space. If the fragmentation is done per object polygon, this resorting of polygon numbers can be done.

If set to **Convex Hull**, the Child objects will be hidden and replaced by new geometry that works like a shrink-wrap around the original shape. Concave areas and gaps between objects will be omitted. Alternatively, all Child objects can be covered with such a shrink-wrap. The gaps between objects will then remain. To do so, enable the **Per Object** option.

If **Bounding Box** is selected, only the object's bounding box will be displayed. This is simply a cube that is scaled to the size of the object, so that all of its vertices are encompassed. Here you can also select the **Per Object** option to affect each object individually.

The simplest display quality is the **Null** mode, which replaces the objects with a simple Null Object. These can be displayed as a circle, rectangle, star, etc.

5.6.1.10.3 LOD Options

If the **Progressive** option is enabled, lower LOD levels will be maintained when the LOD level is raised. This can, for example, be used when approaching an increasing number of objects, letting an increasing amount of detail appear.

The **Always Shoe Unassigned Object** option refers to the **Manual Groups** mode and will permanently display all objects that are not linked with any **Object** lists.

Normally, the same objects and levels of detail that the DOF Object uses in the Viewport will be output for rendering. If the reduction of an object's complexity makes sense for faster Viewport performance is of secondary nature. The best possible state should always be visible. Enable the **Don't Affect Render** option to do so.

Alternatively, the **Render LOD Increment** value can be used to increase the LOD level for rendering.

Camera clipping is particularly interesting when using LOD in combination with Instances or MoGraph clones. It can be used to evaluate the camera's viewing cone to make only the objects that lie within the camera's field of view active. Using **Cone: Safe Distance**, the objects' visibility can be expanded to a defined region. **POV: Safe Distance** can also be used to exclude a spherical region around the camera from omitting objects.

The **Use Camera Clip in Render** option can be used to activate these effects for rendering. The **Polygonize Objects** option is a special option and only effective in **Children** mode. Since the **LOD Object** uses a new hierarchy for the objects internally, it can occur that individual links are broken, as can happen when using Deformers. Of this should occur, enabling the **Polygonize Objects** option can help. The only disadvantage is that only the polygonised version of the Child objects appear after the **LOD Object** is converted.

5.6.1.11 Polygon Reduction

Polygon reduction makes sense whenever an object's polygons have to be optimized and the object is not a parametric object or an object created using Generators. To do so, make the object to be reduced a Child object of the **Polygon Reduction** tool. After a brief calculation time, which can be followed in the progress bar at the bottom of the Viewport, the degree of reduction **(Reduction Strength)** can be defined. The resulting number of triangles, edges and vertices will be displayed simultaneously but can also be entered directly to better control the reduction. However, an exactly defined value can only rarely be achieved.

If more than on Child object exists, the **Reduce all generator children as one object** option can be used to reduce all Child objects as one object. If the objects should be reduced individually, the values for the triangle, edge and vertices counts will omitted. You will then only be able to work in percentage using the **Reduction Strength** value. It is generally recommended to enable this option when simultaneously reducing multiple objects with different polygon densities because lower-resolution objects will be more apt to maintain their original shapes longer.

If the objects contain holes or open polygon edges, these regions can be better protected against extreme modification by enabling the **Preserve 3D Boundaries** option. In addition, the **Boundary Reduction Angle** value can be used to define the angle up to which vertices will not be removed by the algorithm if no change in rotation takes place.

The **Preserve UV Boundaries** works similarly in that it maintains the vertices that lie on the edges of UV polygon islands and thus prevents assigned textures from slipping or smearing.

In addition to UV coordinates, polygon reduction also takes vertex maps into consideration and calculates their weighting for the new vertices on the reduced object.

5.6.1.12 Help Functions

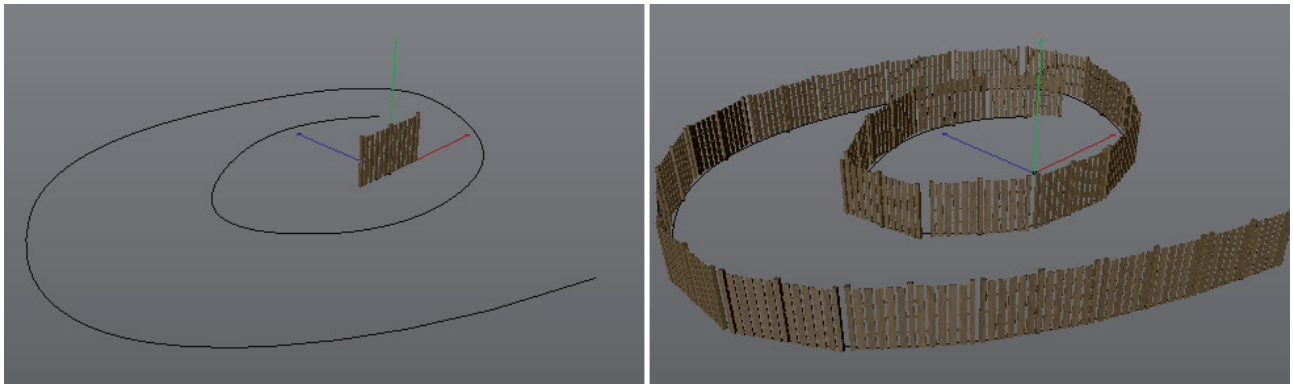
In the Cinema 4D **Tools** menu you will find numerous helpful functions for everyday work. We will discuss the most commonly used tools for duplicating and randomly arranging objects.

5.6.1.12.1 The Duplicate Function

The **Duplicate** tool will automatically affect the currently selected object when selected. This tool can be used interactively as long as it remains selected. You can experiment interactively with the number and arrangement of duplicates before exiting the tool.

The number of copies is defined using the **Copies** value in the **Duplicate** tab. The **Clone Mode** setting lets you define the type of duplicates that should be created. If **Copies** is selected, individual copies of the original will be made. If **Instance** or **Render Instances** is selected, instanced objects will be created whose function and difference to copies we've already been discussed.

The **Options** tab's **Mode** setting defines the arrangement of the duplicates. You can select from a **Linear** arrangement, a **Circle** arrangement (works like the Array tool) and an **Along Spline** arrangement.

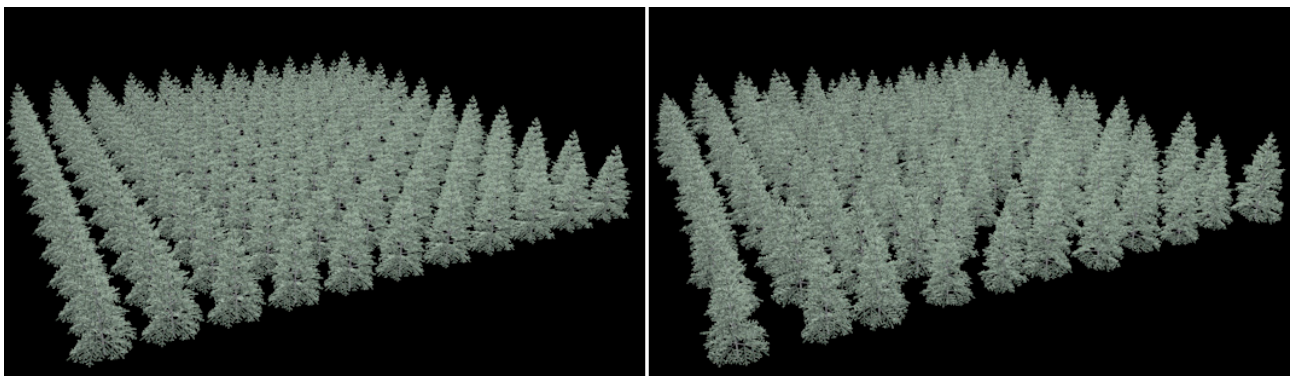


The **Per Step** option defines how the subsequent settings will be applied. For example, the **Move** values can be assigned each relative to the previous duplicate (**Per Step** enabled) or absolute for all duplicates as a whole. Hence, if you want to stack 10 cubes on top of each other with a total height of 2 meters, the **Linear** mode must be used, **Per Step** must be disabled and **Move** values of 0m, 2m, 0m defined. Otherwise the gap between neighboring cubes will each be 2 m.

In addition to the **Move** settings you also have **Scale** and **Rotation** settings that can be used to affect the duplicates. Duplicates will be generated when the **Apply** button is clicked. You can subsequently modify all settings until you achieve the result you want.

5.6.1.12.2 The Randomize Function

Equal spacing, precise placement, exact scaling and perfect rotation are not always desired or necessary. In fact, slight variations often look more realistic and natural.



These variations can be created using the **Randomize** tool, which is located in the **Tools** menu. This tool will automatically affect currently selected objects when selected. This will not work if a sub-object is selected.

Use the **Move**, **Scale** and **Rotate** values to define the maximum deviations and confirm your entries by clicking on the **Apply** button. If you want to modify the result you can use the **Seed** value, on which the random calculation is based, to do so. You can modify the values as long as the Randomness tool remains active. Once you exit the tool the result will be permanent.

▶ *See: Exercises for modeling objects*

SUMMARY: MODELING OBJECTS AND HELP FUNCTIONS

- As a rule, modeling objects work like Generators, which means that objects they affect must be made sub-objects of these objects.
- The only exception to this rule is the Instance object to which an object must be assigned. The **Volume Builder** can also be used to link objects.
- Modeling objects to not affect original objects. The original object's original state can be restored at any time by disabling the modeling object or by removing the original from the modeling object's hierarchy.
- The **Array** object can be used to create any number of copies and will arrange them circularly. The copied objects' Z axis will always be oriented outwards.
- The **Atom Array** object can be used to give an object a wireframe appearance. The object's points will be displayed as spheres and its edges as cylinders.
- The **Boole** object calculates intersections and can be used to subtract parts of shapes.
- The **Spline Mask** works like the Boole object but with two-dimensional splines. The **Spline Mask** itself can be used like a spline object, e.g., in conjunction with an **Extrude** object.
- The **Connect** object optimizes the points of sub-objects or linked objects and can be used to weld adjoining shapes.
- An instanced object acts like a copy of an object. The instance is linked to the original object and all modifications made to it will also be made to the instance.
- The **Render Instance** is a special type of instance that greatly reduces the amount of memory required, which makes it possible to use very many instanced objects. Contrary to the normal instances, a **Render Instance** contains none of the original object's point information and can therefore not be deformed individually or converted to a polygon object.
- The **Metaball** object interprets an object's points as spheres that merge like drops of water or mercury.
- The display quality of the Metaballs can be defined individually for the Viewport display and for rendering.
- The calculation of each object to which a **Metaball** object is assigned can be controlled using the **Metaball** tag's settings.
- The **Symmetry** object interactively creates a mirrored copy of its sub-object. If the original object's points lie near the mirror plane they can be clamped in place welded to the mirrored half.
- If more than just a few object copies are needed, these can be created using the Duplicate function.
- The objects can be places on a line, circularly or on a spline curve.
- Properties such as Move, Scale and Rotation can be interpolated via the number of copies.
- The Random function can be used to randomly position, scale and rotate selected objects.

6 Creating Test Renderings

The following sections will cover lighting and defining surface materials. Not all effects are displayed in full quality in the Viewport.

Test renderings are important when creating materials and when positioning lights. As a rule, these are renderings that are done either directly in the Viewport or that are made using lower quality than the final rendering. Even though this is not directly related to the light or material systems, it's nevertheless important to discuss the primary techniques for creating test renders for your scenes so you can check your scenes while working through this curriculum. Even though many effects can be simulated in the Viewport directly via OpenGL, refractions in transparent materials, area shadows and more complex reflections, for example, will only be visible if raytrace rendering is used and can therefore not be test-rendered in the Viewport.

The test rendering's quality is defined in the **Render Settings** menu. We will take a brief look at these menus and explain the most important settings. The remaining settings will be explained later in conjunction with other rendering processes. The **Render Settings** dialog window can be opened using the **Render** menu's **Render Settings** command or by clicking on the icon in the top icon bar.

6.1 The Render Settings

This dialog window contains all settings used to define the quality and type of rendering. Many of these settings are not relevant at this point. We will concentrate on those settings that are important for creating test renderings.

6.1.1 Selecting the Right Renderer

The type of renderer used can be selected from the **Renderer** menu at the top left of the dialog window. **Standard** is the default renderer used by Cinema 4D, which offers a normal quality output and also includes raytracing. Additional effects such as **Global Illumination** or **Caustics** can be added by clicking on the **Effect** … button and selecting from the menu that appears. The **Physical** render mode simulates, among other things, real camera systems and offers optimized render algorithms. These include, for example, motion blur or depth of field. **Hardware OpenGL** mode can be compared to the results you get when rendering in the Viewport.

ProRender is an unbiased GPU renderer that uses the graphics card for rendering. In contrast to **Hardware OpenGL**, **OpenCL** is used, which can also be used for raytracing. If the scene was lit realistically and the materials defined as physically correct as possible, this renderer will produce very realistic results without much tweaking. In addition, the calculation of VDB volume effects, e.g., as for fog or clouds, can currently only be handled by ProRender. This renderer is much faster than the **Standard** and **Physical** renderers. Another advantage is the iterative rendering of the scene. This can quickly give a good first impression of the scene. This expeditious feedback makes it possible to more quickly assess lighting, materials and other scene elements. For example, the effects of moving a light source can be seen almost instantly. Since this renderer can also be used directly in the Viewport, it is possible to work in a permanently rendered state, which is updated automatically after each modification is made.

Other installed renderers such as **Redshift, V-Ray** or **Octane** will also appear in the Renderer menu, where they can be activated.

6.1.2 Output Settings

These settings affect the resolution of images and animations for rendering. The settings at the bottom of the menu are for defining the number of frames and frame rate for animations. At the top of the menu you will see a white arrow that, when clicked upon, displays numerous presets for common image and video formats. These values can also be entered manually using the **Width** and **Height** settings. You can even convert a resolution in centimeters or millimeters to pixels. Simply select the corresponding conversion type from the drop-down menu next to the **Width** setting. You can also enter a DPI or pixel per cm value in the **Resolution** setting. The Cinema 4D resolution will automatically be adapted.

If a resolution has been defined manually, the **Lock Ratio** option can be enabled, which will automatically adjust the **Height** or **Width** value if only one of the two is modified, thereby maintaining the correct aspect ratio. Even if the final resolution is not used for test rendering, it's better to define the right resolution at the start so the correct aspect ratio is always maintained for testing. This way, it's easy to determine what parts of the scene will be rendered when borders and darkening (safe frames) are included. The degree of darkening can be defined in the view's **Configure** menu.

The aspect ratio is the ratio of an image's height to its width and can also be defined using the **Aspect Ratio** setting. A drop-down menu next to this setting lets you choose from commonly used aspect ratios such as 4:3 or 16:9. You might be surprised that pixels don't always have to be square. Some older video formats, for example, are rendered with distorted pixels, whose distortion is corrected when broadcast. This aspect ratio can also be defined. However, for standard images or for high-definition television images, an aspect ratio of one will be the right choice. The same applies to the **Fields** setting for animations, which is located near the bottom of the **Output** menu. These options produce an interlaced rendering. This is not suited for still images and not necessary for many types of state-of-the-art output methods. If you have doubts, check with your client or a production supervisor whether or not an interlaced output should be generated.

The **Frame Range** menu defines the range that will be rendered. If **Current Frame** is selected, only the frame currently open in the Viewport will be rendered, regardless of the animation's total length. If **All Frames** is selected, the entire animation range, e.g., as defined in the **Project Settings** menu, will be rendered. If **Preview Range** is selected, only the range defined in the **Timeline** will be rendered, which can be any range within your animation. If **Manual** is selected you can manually enter the range using the **From** and **To** settings. The **Frame Step** value defines which frames should be omitted when an animation is rendered. As a rule you will use a value of 1, which will render all frames. However, a higher setting of 3 or even 5 may be enough when checking a scene's lighting, for example, which will render only every 3rd or 5th frame, respectively. If you change this value, don't forget to set it back to 1 before the final render.

Next, we have the **Frame Rate** setting, which should match both the output medium as well as your Project's settings. This setting has no affect when rendering still images. Below the **Image Resolution** setting you will see the **Render Region** setting. This setting can be used when rendering complex still images at a high resolution. This setting lets you render a specific part (region) of the image without having to arduously render the entire image for testing. Click on the small arrow to make additional options available. Use the four border values to define the region you want to render. The region within these four border values will be rendered. The surrounding part of the image will remain black. If you are already using an **Interactive Render Region** in the Viewport, this region's border valued can be copied and pasted here by clicking on the **Copy from IRR** button. This makes it easier to target a specific region. What **IRR** is and how it works will be explained after the **Render Settings** section. At the far bottom of the Output menu you will find the **Annotations** field. Here you can type comments or information regarding anything from deadline dates to required settings and anything that can be helpful to you or others working with this file in the future.

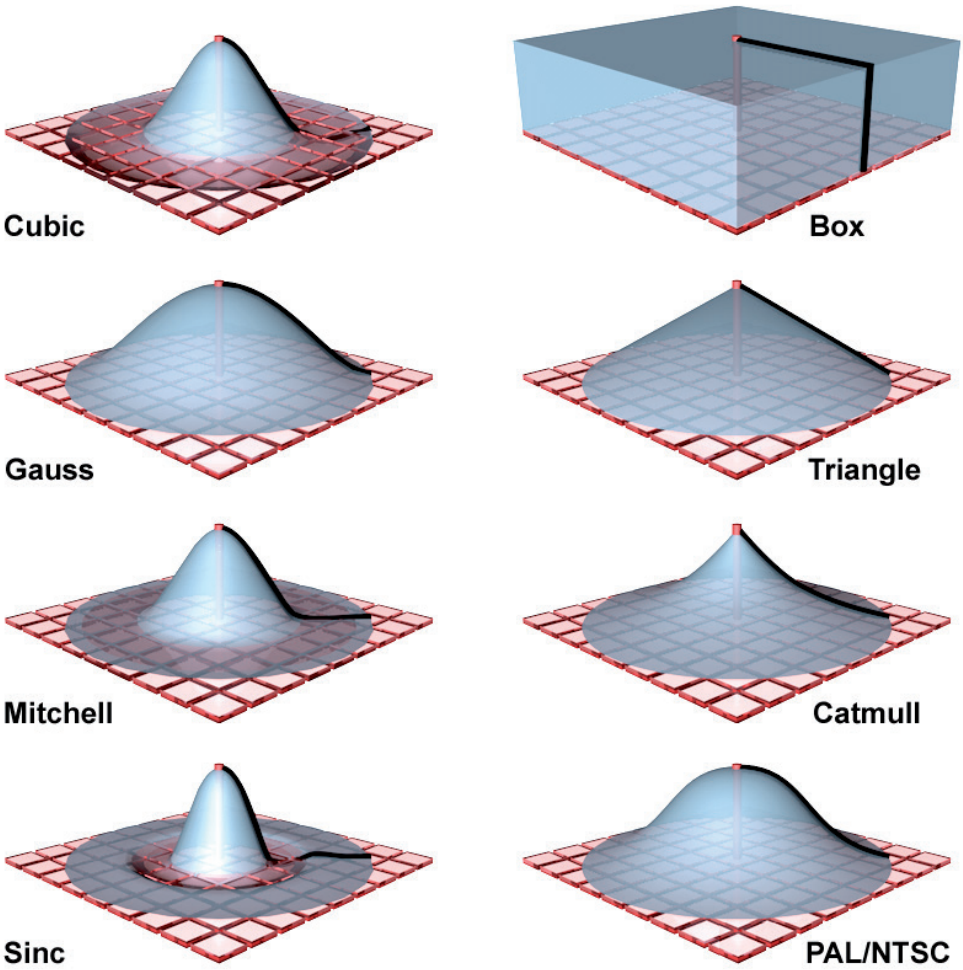
Animations can then be output directly as videos. You've already learned how to define an animation's length and frame rate. A video format, which is often defined via a data rate, can be defined in the the **Save** function in the **Render Settings**. The **Adapt Data Rate** option ensures that the data rate will be adapted correctly even if modifications are subsequently made to the image resolution. An increase in resolution will result in a proportional increase of the data rate to ensure a constant compression quality.

6.1.3 Antialiasing Settings

A rendered image's quality depends to a large part on the type and degree of antialiasing. As you know, pixels can often be seen on angled edges when they are displayed on a monitor. This effect can be reduced by antialiasing when the **Standard** renderer is used. If set to **None**, no edge smoothing will take place. Edges as well as materials will look correspondingly rough because transparencies and reflections also benefit a lot from good antialiasing. This setting can, however, be useful for test renderings to save time.

If **Geometry** is selected, the objects' outlines will be smoothed. This can increase the output quality quite a bit but it will not affect materials. Therefore, final renderings should always be done using the **Best** mode so both objects and materials are blurred.

The **Filter** setting offers various antialiasing algorithms.



The **Cubic (Still Image)** and **Gauss (Animation)** are already optimized for their respective purposes. Animations have more natural look when the Gauss filter is used. All filters generally define how image pixels' sub-pixels should be calculated. Antialiasing subdivides each rendered pixel into smaller units – the sub-pixels. The finer the sub-pixel subdivision, the more information that can be rendered for a given edge – and the longer the image will take to render.

Depending on the filter selected, neighboring sub-pixels will be blended more or less strongly. This blending is based on curves – similar to splines. Many of these curves focus the interpolation of the sub-pixels at the pixel's center, which results in a sharpening of the pattern, which makes this method suited for still images. This is especially true for the **Cubic**, **Mitchell** and **Sinc** filters and can result in extreme sharpening of contours and stark contrasts in the rendered image. In some cases it can lead to flickering in animations simply appear as a very obviously sharp edge in the image. If this happens, enable the Clip Negative Component option to prevent extreme sharpening.

Gauss and **PAL/NTSC** on the other hand have a harmonious filter progression and are generally well-suited for animations and produce smooth images. The remaining filters produce a result somewhere between smooth and additional sharpness. Enabling the **Custom Size** option will let you manually define the effect on the sub-pixels. The **Filter Width** and **Filter Height** values define the number of pixels that should be included for the selected filter for the X and Y direction of the rendered bitmap. These values can be increased up to a value of 4, which will produce an extreme sharpening. If **Custom Size** is disabled, the **Filter Width** and **Filter Height** fields will automatically display the values Cinema 4D will use for rendering. This is a good reference for defining your own antialiasing values.

Selecting the **Best** option will make additional settings available. Because this option also affects materials, the **Threshold** value can be used to fine-tune this effect. This value defines the minimum color deviance of neighboring color pixels for which additional smoothing via **Antialiasing** is activated. This means that the additional sub-pixels will only be generated if the color deviation exceeds the **Threshold** value. The default value of 10% should be sufficient in most cases. In some instances, this value might have to be lowered to increase precision.

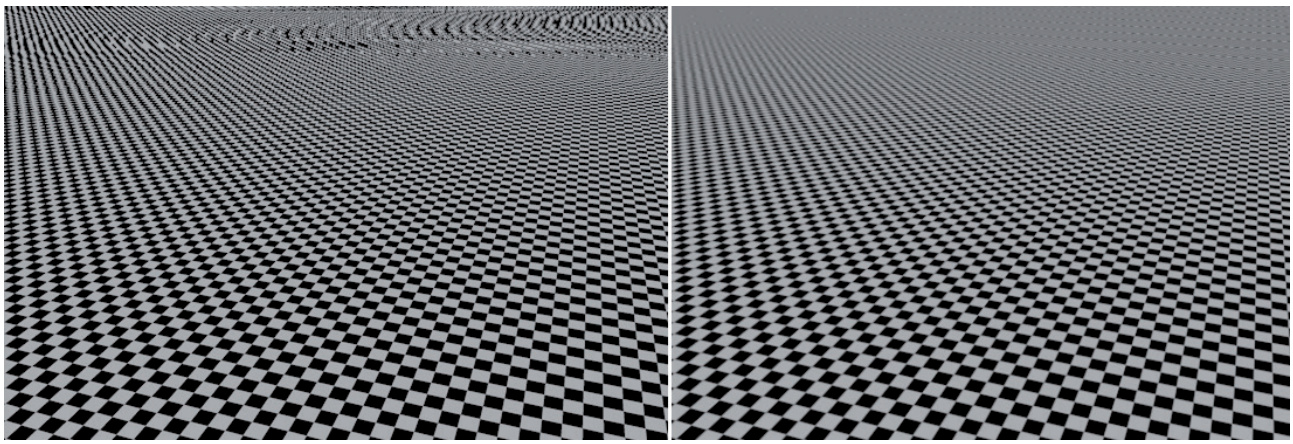
The actual intensity of the antialiasing is defined by the **Min Level** and **Max Level** settings. These settings define the number of sub-pixels that are subdivided in each image pixel. A setting of 4x4 means that a pixel will be subdivided four times in its height and in its width, which means that 16 sub-pixels per pixel will be generated.

Min Level defines the minimum strength of this effect for all image pixels. If you experience problems on simple geometry or for example on shadows, decrease either the **Threshold** value – if the problem is related to color – or in crease the **Min Level** value. **Max Level** defines the maximum limit for sub-pixel subdivision. The setting affects transparencies, shadows and finer texture details, for example. Here, the quality can also be improved by increasing the value. Larger values will also lead to correspondingly longer render times. Therefore, simply setting both values to their maximum values doesn't make any sense. Slowly increase values to find the best compromise between quality and render time.

We will also discuss how antialiasing strength can be defined individually for each object using a **Compositing** tag. These tags are located in the ***Object Manager's*** **Tags** menu or you can right-click on the respective object in the ***Object Manager***. The **Consider Multi-Passes** option must be enabled to include this tag's effect.

When images are saved their individual properties can be saved separately. For example, shadows, reflections or highlights can be saved as separate layers, which, of course, is very useful for the post-production phase. These layers that make up these images are called **Multi-Passes** in Cinema 4D. Enabling the Consider Multi-Passes option will, for example, ensure that antialiasing is rendered in high quality for Alpha channels.

The abbreviation **MIP** stands for a process that helps reduce image noise for finer structures via blurring. This is especially helpful when fine structures or patterns are viewed from flat angles when a lot of image information is included in very tight pixel regions, which can lead to noise effects.



The **MIP Scale** value defines the effect's global strength for rendering. This is a type of multiplier for all **MIP** and **SAT** settings, which can also be defined in the materials themselves. A value of 200% would result in a doubling of all **MIP**/**SAT** settings. Generally speaking, the higher the setting the more antialiasing that will be applied to objects in the distance (from the camera). The bottom-most setting in the Anti-Aliasing menu is the Small Fragments setting. Cinema 4D uses one of two different methods for rendering. The **Scanline** method scans the image row for row. The time required to do this correlates directly to the number of polygons beneath the scan line. This method provides the highest quality results but also requires long processing times, e.g., if a large number of polygons lie near the horizon line within a small area.

The second method is the **Raytrace** method, which is much faster even in regions with a high number of polygons. However, it does not provide the high-quality results of the **Scanline** render method. The Hybrid method applies the advantages of both methods. Cinema 4D will determine which render method should be used at which locations in the image. For example, high-res objects that are too small to be seen from the current angle of view will be rendered using the faster **Raytrace** method. Since these objects lie so far from the camera, the lower quality rendering will not be noticeable. Objects that lie closer to the camera will be rendered using the **Scanlie** method.

If the **Physical** renderer is used, antialiasing works somewhat differently. This renderer has its own menu in the **Render Settings** when it's selected. If **Sampler** is set to **Adaptive**, which calculates the number of steps per pixel to optimize rendering, the **Sampling Subdivisions** value defines the number of sub-pixels in which each pixel should be divided. The value is applied to the power of two. A value of 0 will already result in a subdivision per pixel, i.e., one calculation per pixel. However, these calculations do not yet produce a color calculation for the pixel but are passed on to the next parameters that use the **Sampling Subdivisions** information to process additional calculations within the sub-pixel. **Shading Subdivisions (Min)** and **Shading Subdivisions (Max)** define the minimum and maximum number of samples per sub-pixel. These values are also applied to the power of two, whereby **Shading Subdivisions (Min)** is always applied and, depending on the results of the previous **Sampling Subdivisions** calculation, the calculation can be increased to the Shading Subdivisions (Max) value on a per-sub-pixel basis. If a low Sampling Subdivisions value was defined, too little information per pixel will simply be available to determine an increase and it can happen that the defined maximum limit of shading samples is not reached, which in turn reduces the resulting quality, even though a relatively high **Shading Subdivisions (Max)** was defined.

The **Shading Error Threshold** value helps determine when **Shading Subdivisions (Max)** should kick in. The lower the value, the more often the **Shading Subdivisions (Max)** will be applied. In the end, the ascertained per-pixel colors will be calculated together via the **Filter** setting, which can be found in the **Physical** renderer's **Antialiasing** menu in the **Render Settings**.

When using **ProRender**, the sampling and antialiasing settings become even more simple. If **ProRender** is the active renderer, a **ProRender** menu will be made available in the **Render Settings** menu, whose **Offline** and **Preview** tabs share many of the same settings. The **Offline** settings are for the final rendering, i.e., for achieving the best possible render results. The **Preview** settings are designed to achieve the fastest possible rendering with the least amount of quality loss.

The **Anti-Aliasing Grid** can be defined for both types of rendering, which defines the number of sub-pixels each pixel should be divided into. Each sub-pixel then contains the number of samples defined by the **Anti-Aliasing Samples** value, which will in turn be calculated together using the defined **Filter**. This principle is the same for all three raytracers.

The antialiasing functions are also available for the **OpenGL** renderer to smooth reflections and transparencies. In addition to the normal **Anti-Aliasing, Supersampling (Brute-Force)** can be used to internally render a larger image, which is scaled down for output. Since multiple pixels are combined into a single color value when the image size is reduced, the overall image quality can be improved, especially when using transparencies.

6.1.4 Additional Options

The **Options** menu contains general settings that you won't have to modify in most cases. Nevertheless, it's a good idea to get familiar with these functions. The first options define whether or not **Transparency**, **Refraction**, **Reflection** or **Shadow** should be rendered. One or more of these options can, for example, be turned off for test renderings to make them faster.

Certain settings can also be restricted. Reflections, for example, can be restricted to Floor and Sky objects, or only soft shadows can be rendered. We will explain in detail what **Shadow Maps** are and how these work. For now, all you need to know is that these are simple bitmaps that are used by light sources to render soft shadows. These bitmaps can be recycled through multiple test renders to save render time. For example, the rendering of an animation in which no lights or objects are animated and only a single camera motion takes place can be sped up by enabling the **Cache Shadow Maps** option. The shadows don't change and can therefore be reused.

Blurriness is the blurring of transparencies and reflections. A frosted glass, for example or a brushed metal surface are good examples. Enabling or disabling the **Global Blurriness** option does so globally for the Project. If **Active Object Only** is enabled, only the currently selected object will be rendered. Enabling the Textures option renders textures (images) that are used in the materials.

If a texture's path has changed, Cinema 4D might not be able to access it anymore. Enable the **Show Texture Errors** option to make sure a corresponding error message is displayed when the render process is started.

The **Default Light** is the light that illuminates your scene until you actually place a **Light** object into it. Normally, the **Default Light** will be deactivated (turned off) as soon as you place a **Light** object into your scene. If this option is enabled, the Default Light will, for example, be disabled if **Global Illumination** is activated. This is useful, for example, if you want to illuminate your scene using only luminous materials. Generally speaking, this option can be left enabled. Later in the curriculum we will discuss how lights can be used to emit volumetric light, which can then be used as a cone of light or as rays of light that shine between clouds. The **Volumetric Lighting** option makes these effects possible.

Spline modeling objects as well as parametric Primitives can be displayed in varying levels of quality. The Viewport's **Options/Level of Detail** menu offers several options from which to choose. The level of display quality can also be defined individually using the **Display** tag. In the *Object Manager*, right-click on the object to which you want to assign the tag and select it from the **Render Tags/Display** menu. The **Use Display Tag LOD** option must be enabled for these tags to be used for rendering.

HUD is the abbreviation for the Viewport's Head Up Display. It can be used to display various types of information such as the name of the Viewport, object settings or even the current frame number directly in the view. This display can be configured in the Viewport's **Options/Configure** menu. Enable the Render HUD option if you want this information to be displayed in the rendered image(s). A Doodle is an image layer that can be used to make notes or doodles. This can be done using the **Doodle** brush in the **Tools/Doodle** menu. Enable the **Render Doodle** option to include the **Doodle** layer in the rendered image(s).

Sub-Polygon Displacement is a material effect that can be used to increase the subdivision of an object's surface or deforms it using a texture. Enable this option if you want a material to affect a given object accordingly. The material property **Subsurface Scattering** behaves in a similar way. This effect simulates light that passes through an object. Fingers through which a white light shines red or a candle are good examples. Then there are the **post effects,** which can be activated separately for rendering. These effects include **Depth of Field**, **Color Correction**, **Sharpen Filter** and many more. The options for effects also depend on the selected renderer. For example, the Physical Renderer and Pro-Render can render blurs even withour separate effects based on the physical settings of the camera used. The **post effect** options make it easy to enable or disable a wide variety of effects.

The **Identical Noise Distribution** option affects image noise that can result from using low sampling values. This can, for example, result when **Area** shadows are rendered, when using selected blur effects or when using **Global Illumination**. You will learn more about this later. If this option is disabled, the structure of the noise will look different in each image of an animation, which will more closely resemble the grain of real-world film. If this option is enabled, the noise pattern will be static and will look like image interference that moves with the camera. Therefore, enabling this option for rendering still images doesn't make much sense.

As you will see with the first test renders, the image is essentially assembled using little squares. These are called **Buckets** in Cinema 4D and represent the area on the image that is currently being rendered by the processor(s). The order in which the **Buckets** should render is defined using the **Bucket Sequence** setting. These settings are more cosmetic in nature than anything else and do not affect the render time or quality in any way. The size of the **Buckets** can be defined manually or automatically. You can leave the **Automatic Size** option enabled or disable it and define a custom size using the available options. Note that larger **Buckets** can increase memory requirements accordingly. If you are using a computer with multiple processors or cores, it's recommended that you don't use **Buckets** that are too large. Especially when making small test renders, it can happen that not all **Buckets** fit in the image, which will slow rendering. This problem is less acute when the **Physical Renderer** is used because it uses one **Bucket** per CPU. When rendering with **Team Render**, a special form of rendering across a network that spreads rendering across multiple computers, more **Buckets** will be used for rendering, depending on the number of computers and processors being used.

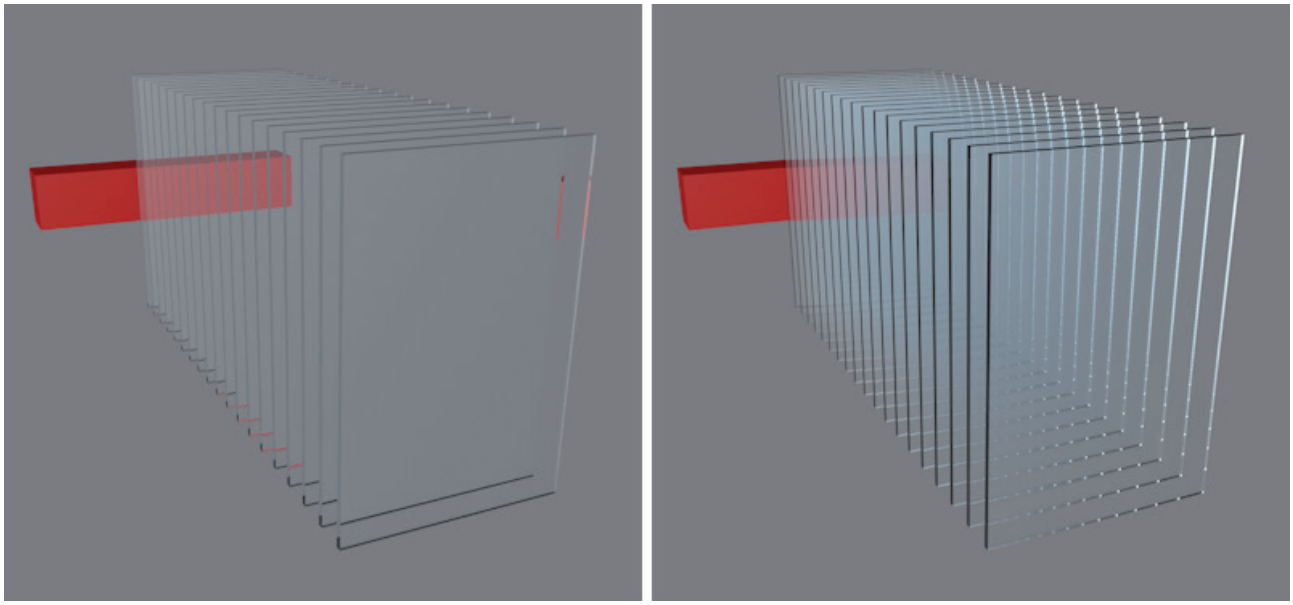
6.1.4.1 Material Override

If objects already have materials assigned to them that are too elaborate to render for a quick test rendering or for testing the equal distribution of light, you can create a new material and assign it via the **Material Override** function. Existing and already assigned materials will then be ignored for rendering. Individual materials can also be excluded from this function by dragging them into the **Materials** list. The overriding of Materials already assigned to objects can also be restricted. For example, if the **Transparency** option is enabled in the **Preserve** menu, all of the materials' transparency settings will be maintained while all other material properties will be overridden by the **Material Override** function.

6.1.4.2 More Advanced Functions

In the right column you will find several numeric values that can be edited. These affect the rendering directly and must be adjusted in some cases. The **Ray Threshold** value defines the point at which the rays will end, which primarily affects the material properties. As soon as the ray hits a surface, for example, that is less reflective or transparent than the **Ray Threshold** value, the ray will no longer be emitted. The rendering will continue with the next pixel. The drawback is that material properties can also end up being suppressed. The best-quality results are produced if the value is set to 0%. However, values that are even slightly higher can help reduce render time without loss of quality.

The **Ray Depth** value also affects materials for rendering – primarily with regard to transparencies. The **Ray Depth** value defines how many transparent polygons can be penetrated by a single ray before the calculation of the ray ends. This not only applies to glass or water materials but also to objects masked with **Alpha** masks. For example, if you want to line up several panes of glass behind each other or want to create a masked leaf texture, it might be necessary to increase the **Ray Depth** value.



Otherwise, the default value of 15 will be good enough for most scenes. The **Refection Depth** setting works similarly but it affects rays that are reflected by a mirrored surface. For example, if two mirrors are facing each other, the calculated ray will, in theory, bounce back-and-forth indefinitely between these surfaces. This process can be stopped after just a few refractions without producing a noticeable difference.

The **Shadow Depth** setting works like the **Refraction Depth** setting but affects the calculation of shadows. As long as the calculated ray has not reached the value defined for **Shadow Depth**, shadows will be calculated for the objects visible at that location. Increasing this value should only be necessary in extreme cases for scenes with a high number of reflective objects.

The **Level of Detail** setting has already been explained briefly. This setting affects the number of polygons on parametric objects depending on their distance from the camera. As a rule, you will always want to have the best quality when rendering so the default value is set to 100%. The **Global Brightness** setting is a multiplier for the intensity of all lights in a given scene. For example, if a scene's overall brightness is too high, reducing the **Global Brightness** value can help darken the scene. The **Motion Scale** value scales the motion blur for the **Multi-Pass'** output of the **Motion Vector**, which can be used to add motion blur to an animation in post-production. **Multi-Passes** will be explained later in more detail.

6.1.5 Viewport Rendering

Now you know which function many of the important settings for affecting render quality have. We will use the **Standard** renderer for our first test renders. Leave the **Output** set to its default value of 800 x 600, which represents an aspect ratio of 4:3. **Range** should be set to **Current Frame**, set **Filter** in the **Anti-Aliasing** settings to **Cubic (Still Image)** and leave all other settings at their default values. The same applies to the **Options** menu.

To start the first test render, click on the top frame of the Viewport you want to render to make it the active Viewport. Next, click on the **Render View** icon or select the corresponding command from the **Render** menu. You can now watch the scene render in the Viewport.

If you only want to test render a part of the scene you can draw a frame around the region you want to render. To do so, select the **Render Region** command from the **Render** menu. Then simply draw a frame around the region you want to render. Several of the Render menu's commands can also be accessed using the icons in the top icon palette.

If you only want to render a specific object in the scene, you can select it in the ***Object Manager*** and use the **Render Active Objects** command in the **Render** menu or icon palette to render the active object(s) only.

6.1.6 Interactive Render Region

Test rendering can be quite time-consuming especially when materials are being created and fine-tuned or when positioning lights. This is where the **Interactive Render Region** (**IRR**) function comes in handy. As you probably already assumed, this function can be accessed in either the **Render** menu or in the top icon palette. This function always renders in the view for which it was initially used.

A rectangle will appear whose size can be adjusted by dragging the points on its edge and at its corners. The region within the rectangle will be rendered each time an object is moved or a material setting is changed. The respective modification will be rendered automatically.

Since re-rendering complex scenes can be quite arduous a times, the quality of the **IRR** function can be adjusted by clicking and dragging on the tiny triangle at the rectangle's right edge. Drag the triangle downward to reduce quality for faster rendering. If the triangle is dragged to the top of the rectangle, the same high-quality results will be rendered as are produce by the **Render Region** function.

Right-clicking on the rectangle's frame will open a menu from which you can select. If **Alpha Mode** is enabled, all image regions in which no real objects lie will be rendered black. Enabling the Lock to View option will lock the IRR to the view for which it was initially used. If this option is not enabled, the **IRR** will automatically appear in the currently active view. **Anti-Aliasing** applies additional render options available from your graphics card to improve render quality. As a rule, this will not result in any noticeable increase in render times. You can click on **Interactive Render Region Settings** to open a dialog window in which this and other settings are accessible.

The display quality can be adjusted using the **Detail** slider or a numeric value can be entered manually. The **Enable** option lets you enable or disable the **IRR** at any time. Re-selecting the **Interactive Render Region** function will also re-activate the **IRR** automatically. The **Gadget Overlay** defines whether or not handles and lines that lie within the **IRR** will be visible. This makes it easier, for example, to edit Primitives when they lie within the **IRR**.

Clicking on the Save button will save the contents of the IRR as a TIFF image. A dialog window will open in which you can select the location to which you want to save your IRR rendering. A quicker way of doing this is by simply selecting the corresponding option after right-clicking on the **IRR**'s frame.

The position and size of the **IRR** can be copied to the **Render Settings**. This was already discussed in the **Output** settings' section. If the **Render Region** option is enabled in the **Render Settings**' **Options** menu, only the contents of the **IRR** will be included in the final render. All regions outside of the **IRR** will be rendered black. This is what will be seen when the scene is rendered to the **Picture Viewer**. This reflects the final rendering, which, as a rule, also includes the saving of the image(s). We will discuss this in detail later. For now we will stay on the topic of test rendering.

6.1.7 Test Rendering with ProRender

In many cases, **ProRender** can be more useful than the **Interactive Render Region** since it offers a special mode that lets you activate continuous – practically in real-time – refreshing directly in the Viewport. Activate **ProRender** in the **Render Settings** and set the perspective view to the active **ProRender** view in the view's ProRender menu. Additional HUD elements will be displayed at the bottom of the Viewport. The element on the right lets you switch between preview and offline quality. These settings were already covered in the **Antialiasing** section. As a rule, you can leave it set to preview quality in order to get the quickest results.

The **Start ProRender** HUD element will start rendering the view. This is not a one-time process – the renderer will refresh the rendered view until the **Start ProRender** HUD element is clicked again.

Note that **ProRender** must first translate the scene into OpenCL language and materials and shaders might have to be baked before rendering starts. Therefore, it can take a while for **ProRender** to start the render process on the graphics card when it's started for the first time. After that, the process generally works much faster than when rendered via the CPU.

6.1.8 Making a Preview

The **Make Preview** command is located in the **Animate** menu or can be accessed in the Render. This render method is well suited for animations because it renders a range of frames. The **Preview Mode** menu offers several options from which to choose. The **Full Render** option uses the **Render Settings** menu's settings. **Hardware OpenGL Preview** reflects the Hardware OpenGL Renderer in the **Render Settings menu**.

The **Preview Range** options define the frame range that will be rendered. These options should be familiar to you from the **Render Settings** menu.

Selecting the **Manual** option will let you define a frame range to render. The **Image Size** value defines the image's width in pixels. The height will be adjusted automatically in accordance with the aspect ratio defined in the **Render Settings** menu. The **Frame Rate** setting should match that of the **Project Settings** menu's setting. For example, if you want to omit frames and you're not worried about the animation's timing you can, for example, lower the **Frame Rate** setting.

You can also select from different file formats, depending on which operating system you're using, to save your preview animations. Click on the **Options** button to define the movie's codec. Click on **OK** to start rendering. Render times depend on the animation's length and the quality with which it's rendered. Selecting the **Make Preview** command again will stop the current rendering. After the defined frame range has been rendered, the corresponding movie can be found in your user directory under **Preview**.

SUMMARY: RENDER SETTINGS

- Graphics cards are not able to display all effects or properties in the Viewport. These include shadows, transparencies, reflections and Global Illumination, among others. This is why test renderings are very helpful for checking render and scene quality before the final rendering is made.
- The Render Settings menu's options are used to define render quality for both still and animations.
- The resolution defined by the Render Setting menu's Output options and the resulting aspect ratio are responsible for creating the darkened regions within the Viewport. The darkened regions will not be visible in rendered images or movies.
- Antialiasing interpolates colors between pixels to smooth edges such as those on diagonal edges. Pixels are subdivided into smaller sub-pixels that are then calculated contiguously using filter curves.
- Depending on the type of filter selected, a sharpening or blurring of the image will result. Still images benefit from sharpening while animations look more natural when blurring is used.
- Numerous light and material properties can be enabled or disabled in the Render Settings menu, e.g., to help speed up rendering.
- The Ray Depth value defines the maximum penetration of rays through, for example, transparent surfaces, and the Reflection Depth value defines the maximum number of reflections for rendered pixels.
- Test renderings can be started manually; the current view, selected object(s) or the content of an defined render region can be rendered.
- The Interactive Render Region updates automatically when changes in the scene are made; the IRR can be scaled using the handles on its edge.
- All functions can be accessed in the Render menu or using the icons in the icon palette.
- The **Override Material** function can be used to override individual or all properties of assigned materials, e.g., when doing test renderings.

7 Lighting

Lighting an object makes it visible in the Viewport. This is why the **Default Light** remains active until a **Light** object is placed into the scene. The Default Light can be edited in the view's **Options** menu. Selecting the **Default Light** command will open a small dialog window with a preview sphere. You can click and drag on the sphere to change the direction of light, which will update in the Viewport accordingly. However, the **Default Light** does not cast shadows and its color or intensity cannot be modified. Furthermore, only a single **Default Light** can be active. To light objects or a scene correctly, **Light** objects must be added. These can be created using the **Create**/**Lights** menu or via the tip icon palette.

7.1 Setting Up Correct Lighting

Real-world lighting is very complex. It can, for example, be reflected an indefinite number of times between surfaces, each time transferring color information back-and-forth. Real-world lighting can also be refracted or bundled. And then there's the processing of light by the camera lens and so on. The topic of lighting can fill volumes. This is why it's easier to demonstrate using traditional photography as an example.

In Cinema 4D, several settings must be checked to make sure the lights correctly illuminate the scene's objects.

- First, it's important that the display quality is set to **Gouraud Shading** in the perspective view. This is the only mode in which lighting effects from lights can be displayed. In **Quick Shading** mode, for example, only the **Default Light** will be used, even if other lights are present in the scene.
- The next step is to make sure the display quality in the Viewports is correct. If available, Enhanced OpenGL should be enabled in the perspective view. This will use the graphics card to generate the objects' display, which is not only faster but also produces better results than when using the CPU. Highlights in particular will look much better. Additional options are available that even let you display shadows in reduced quality in real-time in the Viewport. Especially the depiction of reflections will be greatly improved.
- You should also make sure that the objects' colors are set up correctly. If they are too dark, too much light will be absorbed. The surfaces will look too dark, even when illuminated by intense light. You should therefore check the **Color** settings in the Cinema 4D **Project Settings** menu. This menu is located in the main Edit menu or can be accessed quickly by pressing **Cmd**/**Ctrl** + **D** on your keyboard. A good neutral color is a gray with the following values: RGB=229/229/229; HSV=0/0%/90%. This represents a loss of light intensity on the illuminated surface of 10%, which is a good median value.
- The **Linear Workflow** option should be enabled in the **Project Settings** menu. This ensures that the brightness range will not be unnaturally amplified by overlapping gamma curves.
- Make sure all objects have the right **Phong** tag settings so their surface shadows look correct.
- Make sure that all objects have slightly rounded edges to avoid highlights on sharp edges and corners.

7.2 How Light Affects a Scene

It might sound trivial but light's first and foremost function is to illuminate our environment. But if this were all, we could simply make do with a light positioned in front of the elements we want to view. This would look like a picture taken with a camera flash in which much of the scene would be too dark.

This type of lighting is probably the worst type for any scene. Even though all objects are illuminated, they also have stark shadows cast from the front, which can result in all three-dimensionality being lost for all shapes. This also happens when persons are photographed from the front with a bright flash and their facial features end up looking flat.

Light should be used to help define objects' shapes, especially since these 3D objects are rendered to 2D images. Images must be given as much depth information as possible so these images look convincing to the viewer. This, for example, also includes additional shadows that would otherwise not be present in real-world situations.

Shadows are much more than just the absence of light. They represent a correlation between objects. A shadow on the floor tells us that an object is standing on that floor. If the shadow were not there it would be difficult to discern size ratios or the position of objects in relation to each other on a 2D plane. Proper illumination is very important for composition and should also be done as precisely as possible – just like modeling. A poor illumination can make the best model look bad. Good lighting, however, can be used to make a poorly modeled object look better.

7.2.1 Common Light Effects

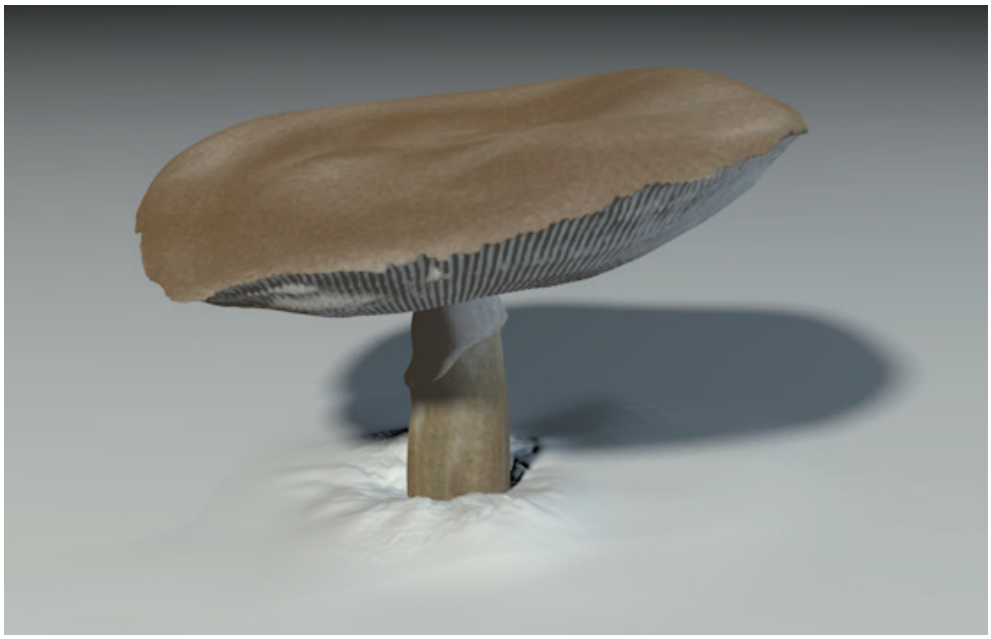
The number and arrangement of lights depends on the scene but light effects can be split into specific categories.

7.2.1.1 The Primary Light

As a rule, this light is the most intense light source in the scene and therefore defines the overall lighting for the scene. For outdoor scenes, this light would be used as the sun, for example. This light should not be positioned frontally in the scene. It should be placed to the side, above or below the camera. This will create better-looking shadows.

7.2.1.2 Fill Light

This light supplements the primary light's shadow casting, for example to prevent hard shadows from being created. Of course a hard, dark shadow can be created intentionally but as a rule, completely black, hard shadows will not be created if a primary light is already in the scene. The fill lights prevent stark contrasts between areas illuminated directly by the primary light and these objects' back surfaces.



7.2.1.3 Backlight/Effect Light

Depending on the environment you're working with, it might be necessary to accentuate objects differently. This is especially true when working with depth of field. You are probably familiar with effect created in portrait photography when a light is positioned behind the person being photographed, which adds contrast to the subject's hair or silhouette. This type of light brushes by the subject and does not illuminate it directly. That's why these lights are mostly positioned behind or above the subject.

Supplemental or effect lights can also be added to create an effect on a specific object such as the twinkle of an eye or the sparkle on a freshly washed car. Hence, lights do not necessarily have to be used to only illuminate objects – they can also be used to add shine and reflection.

7.2.2 The Differences between real-world Lights and Cinema 4D Lights

It should first be noted that all effects described above can be created using Cinema 4D lights. These light objects go even farther and offer additional options that real-world lights don't. For example, lights can be created that don't cast shadows or their highlight and shadow parameters can be separated.

On the other hand, natural light is much more complex in its dispersion. Although theoretically also possible to create in 3D, such precise simulations would take far too long to render. As a rule, we want relatively fast results and we are willing to make compromises to achieve these results.

The lights in Cinema 4D are **direct** lights. This means that the light is emitted directly onto a given object and illuminates its surface. No refraction or forwarding of light takes place. This effect is called **diffusion** or **dispersed** light and must often be rendered as a separate effect using **Global Illumination**. However, this has nothing to do with the light source itself or its placement.

In fact, material properties play a large role in the dispersion of light because light can also be interpreted as reflections that reflect back-and-forth between surfaces and dispersed further.

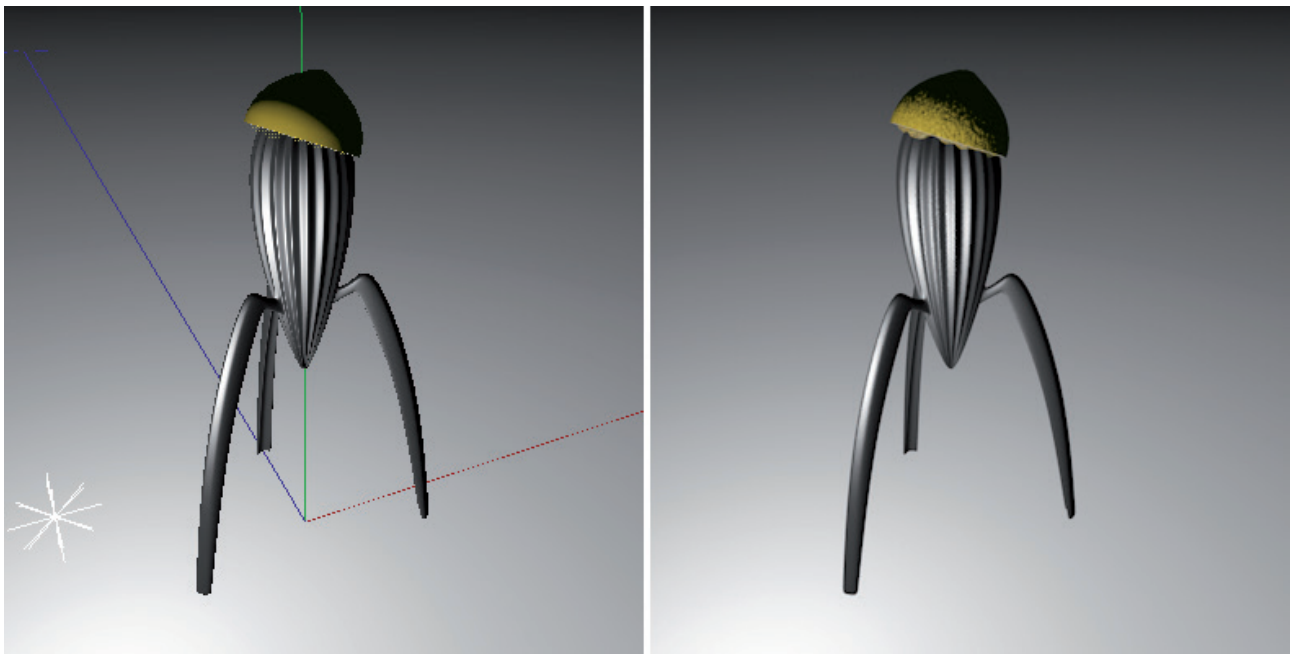
3D lights are also not made up of electromagnetic waves or photon particles. This is why a color spectrum will not be produced by simply shining a white light through a transparent prism structure. Emitted light will also not be bundled on a lens or diverted by large masses. However, Cinema 4D offers functions that can be used to simulate these effects such as **Caustics**, which can be used to simulate refracted or bundled light.

The simplification of light in 3D also bears several advantages, including the variation of shadows, as mentioned earlier. Cinema 4D lights can also be restricted to specific objects or a negative light can even be created, which can be used to darken parts of a scene.

Cinema 4D offers several different types of lights in order to be able to simulate the wide variety of real-world lighting situations. We will take a look at some of these lights.

7.2.3 The Omni Light

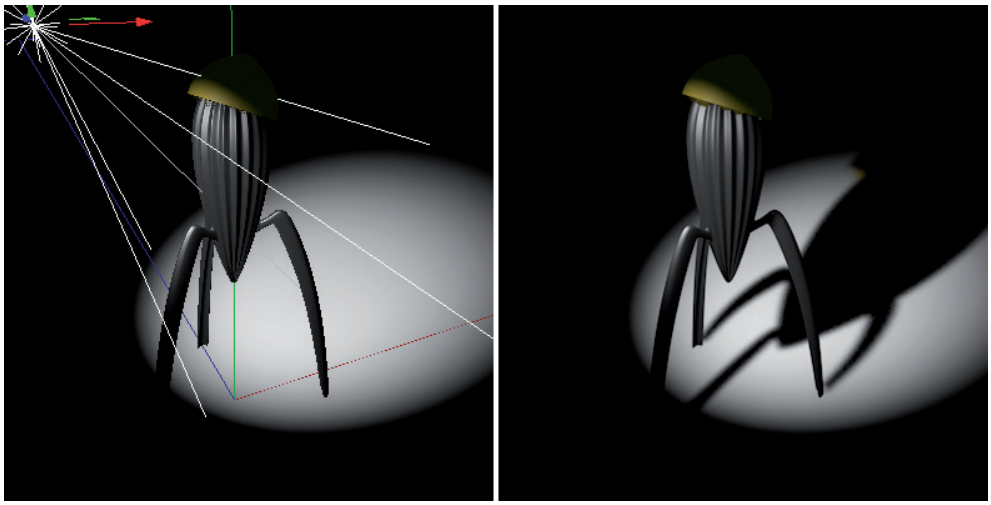
This light emits light evenly in all directions. It can be compared to the light of a candle flame or a light bulb. The light itself merely consists of a position in space, i.e., it has no actual physical size, which means that it can't be reflected in an object's surface, for example. This type of light is very rare in the real world. Artificial light generally has a reflective surface behind or around it, which emits the light in a specific direction. Omni lights are good for creating effects such as specular highlights.



7.2.4 The Spot Light

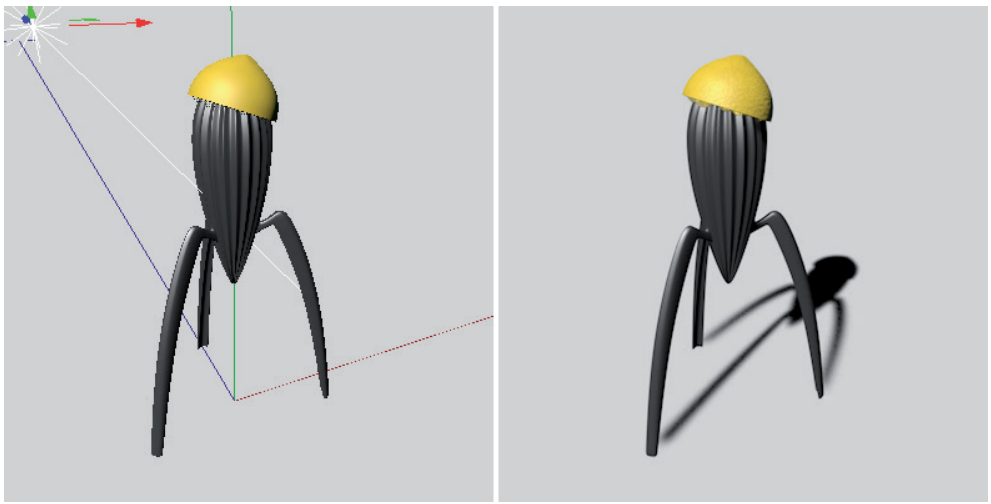
This light has many characteristics of the Omni light. For example, it also has no physical size. However, this light can be emitted in a specific direction, which is similar to a real-world light. The type of light cone can vary. A typical spherical shape can be used or a square spot, which functions like a theater spot light with flaps on the sides.

Another type of Spot light is the **Parallel** light. Light is emitted parallel, like a bundled laser beam that can be endlessly wide. You can also choose between a round and a square shape.



7.2.5 The Parallel Light

The Parallel light can also be implemented without the spotlight characteristic. No surface exists from which the light is emitted. The simulated rays of light will be generated throughout the light's XY plane. Since the light is only emitted in the positive Z direction, all objects behind the light will remain unlit.

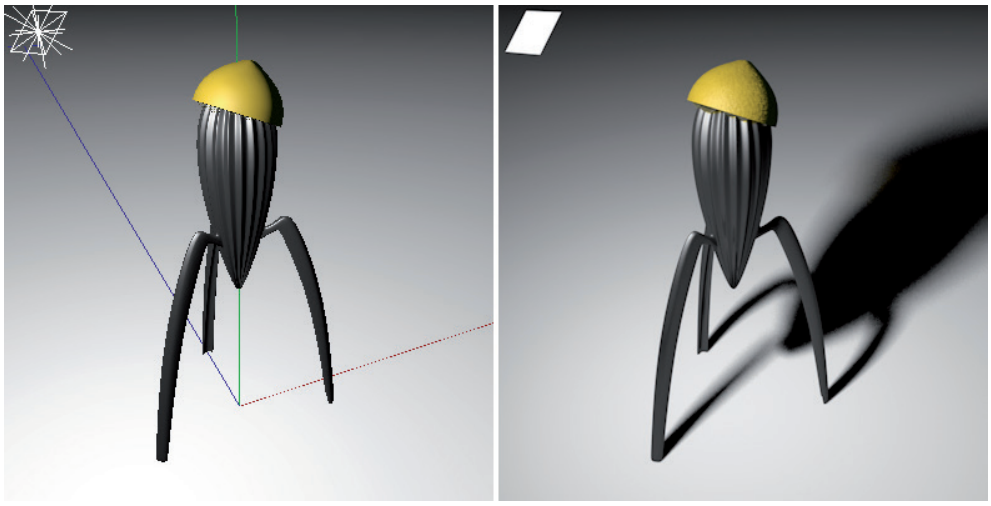


7.2.6 The Infinite Light

This light is very similar to the **Parallel** light. However, the light's position does not play a role for the lighting effect. The Z axis determines the direction in which the light will be emitted. This light is often used for simulating sunlight. This light also has no actual physical size, which means it cannot be displayed in renderings or be reflected as a physical shape in surfaces. This is not the case for the next light:

7.2.7 The Area Light

This light generates the most natural-looking light for the simulation of artificial sources of light. This light can be given the shape of the desired lighting element. It can, for example, be shaped like a plane, like a disc or have a three-dimensional shape such as a sphere. The light is softer and more natural because it is not only emitted from a single point but from an entire surface.

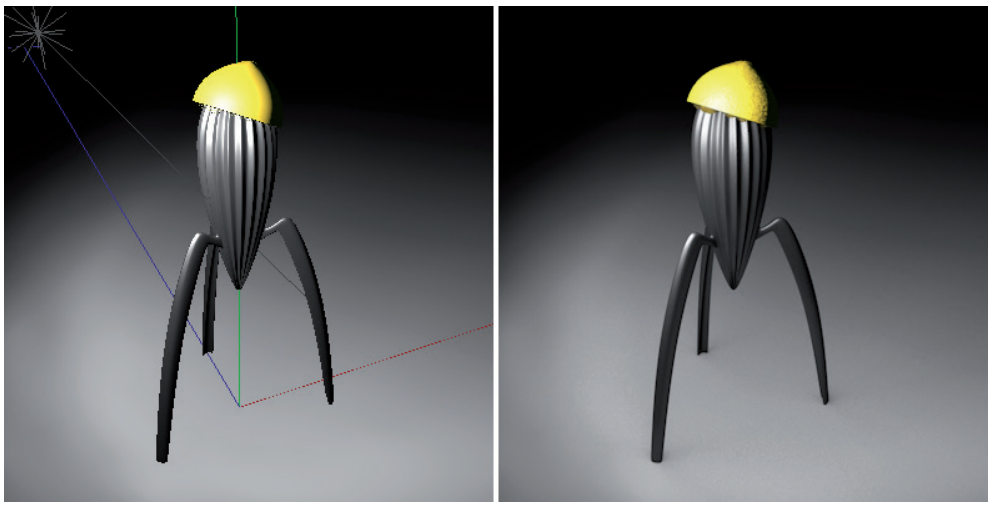


7.2.8 Physical Light

The **Physical Light** is a pre-configured **Area** light for which realistic properties have already been activated. By default, this light has a visible shape, a brightness falloff in the distance and a realistic unit of measure for brightness. This light source also appears as a luminous surface or shape directly in the Viewport and will also reflect in surfaces. Contrary to what its name might suggest, this light also works with the **Standard** renderer and **ProRender**.

7.2.9 The IES Light

This light type reflects the **Omni** light's characteristics but is able to use and IES file's intensity and orientation information. As a rule, IES lights can be gotten for free from lighting manufacturers and ensure a very precise representation of the actual light's emission characteristics. The Cinema 4D **Content Browser** contains several IES files, which can be accessed in the **Presets/Lighting/IES Lights**.



7.2.10 General Light Settings

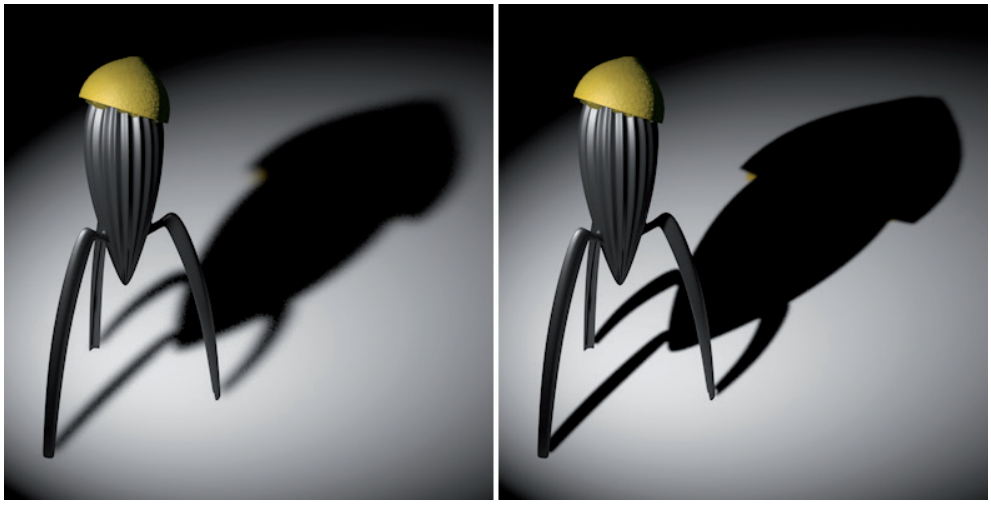
The **Light** objects' **General** tab contains general settings, which can differ depending on the light selected. For example, a light's **Color** or Intensity can be defined in this menu. Click on the small arrow next to the Color setting to make the **Use Temperature** and **Color Temperature** settings available. A light's intensity is defined in percent. A value of 100% will illuminate a perfectly white surface to a degree that the brightest point on the surface will also have a value of 100% white. This brightness is affected by materials assigned to a surface. Don't forget that the brightness of multiple lights is combined, which can lead to a surface being over-exposed. In the end, the brightness of all light sources must be balanced to achieve the best result.

The **Type** setting contains the various light types from which you can choose. The type of light can be changed at any time. You can never go wrong if you select the **Omni** light first and subsequently switch to a different light. In addition, not all light types can be selected using the icons in the icon palette.

Shadows are a separate light property, which means that they have to be defined individually in the light's **Shadow** menu. This also makes it possible to create lights that don't cast shadows. However, if a light does cast shadows, there are several types from which to choose.

7.2.10.1 Soft Shadow/Shadow Maps

The term Shadow Maps in itself indicates that these shadows are created using bitmaps. Before the image is actually rendered, a depth mask will be calculated from the light's angle of view. The distance to each point on the surface, which can be seen (and thereby lit) directly by the light source will be saved as a bitmap. This bitmap will then be projected onto the object from the angle of view of the light. Finally, the shadow will be calculated based on the viewer's or camera's angle of view. The global point coordinates that are ascertained will be compared to the globally converted depth values of the light source's **Shadow Map**. If deviations are found, the corresponding point must lie within the shadow. This sounds complicated but it's something that is calculated very quickly by the computer. The time needed for calculation depends primarily on the **Shadow Map's** resolution. The higher the resolution, the more depth information that must be gathered and the more memory that will be required for that bitmap. Increasing the resolution also increases the sharpness of the shadows' edges and the overall level of detail.



In addition to rendering relatively fast, **Shadow Maps** also have very uniform shadow edges that are very realistic, which is why they are used quite often. However, shadows should be optimized, depending on the type of light used in order to reduce the amount of memory required and simultaneously improving quality.

7.2.10.1.1 Optimizing Shadow Maps

Since the rendering of each **Shadow Map** requires time and memory, the overall number and resolution of these bitmaps should be monitored carefully. This is especially true when using **Omni** lights. These lights emit light in all directions and must therefore calculate **Shadow Maps** in all directions as well. Often, illuminated objects do not lie in

all directions around a light source but only within a light's angle of view. In such cases it's a good idea to point the **Omni** light's Z axis toward the object(s) that should cast shadows, as you would with a **Spot** light. There is a special Viewport option that can help you do this:

Create an **Omni** light and select the **Set Active Object as Camera** command from the Viewport's **Camera** menu. The view will now look at the object from the **Omni** light's angle of view, which is the Z axis. Use the normal **Move** and **Rotate** commands to position and rotate the light so the objects are centered in the view. The light's Z axis will be pointed to these objects. Finally, select the **Default Camera** again from the view's **Cameras** menu. This will take you back to the normal Viewport view.

Of course this method also works with the Spot light sources when you want to point the cone to specific objects or regions of the scene.

The **Shadow Cone** option can be enabled in the light's **Shadow** menu, which will display the cone around the light source's Z axis within which the shadows should be calculated. Enable the Soft Cone option to create a soft fading of the shadow at its edges.

However, if you are using a Spot light type, this will be superfluous. The shadows will automatically be calculated within the light cone's area.

At the top part of the **Shadow** menu you will find the **Density** and **Color** settings. As a rule, these should be set to 100% and black, respectively. You can use fill lights to brighten dark shadows, just like a photographer would do.

The Transparency option should also remain enabled. This ensures that a highly transparent object does not cast the same heavy shadow that an opaque object would. In addition, shadows will assume the transparent object's color, e.g., when light passes through a colored glass.

The bitmap resolution is defined by the **Shadow Map** setting. Custom values can also be used. Note that a higher resolution will have a correspondingly sharper shadow and will also require more memory. The amount of memory required for a given resolution is displayed in the **Memory Usage** value.

The **Sample Radius** value affects the bitmap's interpolation. The larger the radius, the softer the shadow will be. Shadows with high resolutions can be made softer using this setting.

The **Absolute Bias** option should always remain enabled. It ensures that the **Shadow Map's** depth information remains independent of the distance between the light soured and the object casting the shadows. If this option is disabled, the shadow's position relative to the object can change when the light is moved.

The **Bias (Abs)** value defines the amount of correction that is required for the depth mask so the shadow begins exactly at the object's edge, which can otherwise be imprecise due to the limited precision and resolution of the depth mask.

Mathematically speaking, a value of 0 would be desirable but then other disruptive factors would appear that affect the structure of the 3D objects. If the **Bias** value is too small, the polygons' edges will start to cast shadows, which is most obvious on curved surfaces. Depending on the scale of the scene, values just above 0 are better suited. Values that are too high will produce the opposite result – shadows will start farther away from the respective object.

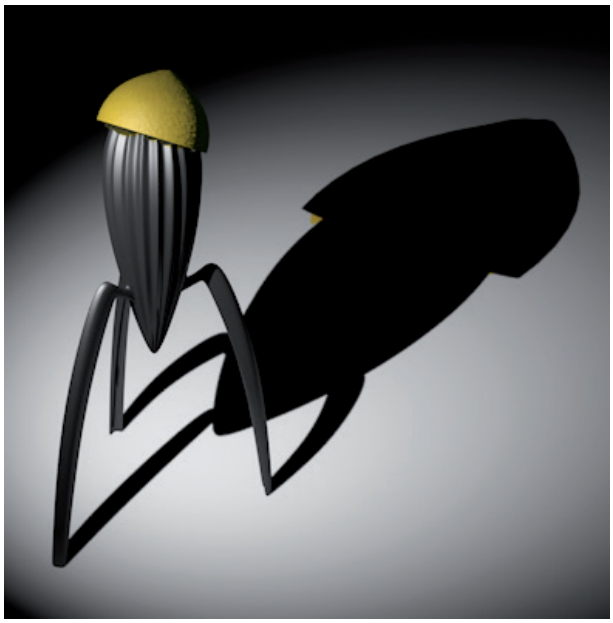
When using a **Parallel** or **Infinite** light, the **Parallel Width** value must also be defined. Because the light from these light types can be emitted from an endlessly large surface, this surface must be restricted to a specific region. Otherwise the **Shadow Map's** resolution would have to be much too high.

The **Parallel Width** value references the distance from the light's position along the X and Y axes of the light's coordinate system. If you notice that, for example, only part of an object casts a shadow, either the light's position or the **Parallel Width** value must be modified accordingly.

The **Outline Shadow** option is a special effect with which only the shadow's outline can be shown. The softness and quality of the shadow can be controlled using the **Shadow Map's** resolution and the **Sample Radius** value.

7.2.10.2 Hard Shadows / Raytraced

This is the right shadow type to use if you want to produce hard shadow edges. This shadow produces only hard shadows and requires no additional settings, as is obvious when you take a look at its available options. Except for the Density, Color and Transparency settings there are no other options that can be defined. Hard shadows require a minimum of extra memory but are slower to render because they are more complex than soft shadows. There are almost no perfect, hard shadows in the real world but they are well suited for depicting shadows cast by bright sunlight or for illustrative renderings.



7.2.10.3 Area Shadow

Area shadows are very realistic because an actual shape is used for the shadow – as is the case with the **Area** light, which also uses an actual shape. Only this shadow type is able to correctly depict core and edge shadows.

The price that must be paid for this precision is the longest render times of any shadow type. This is due to the high number of rays that are emitted by the illuminated surface throughout the scene that have to be sampled. The number of sampling steps determines the shadow's quality, which should be as noise-free as possible.

The **Shadow** menu contains settings that can be used to define the number of samples.

This is an adaptive process, which means that sampling will vary, depending on the region within the scene. The **Minimum Samples** and **Maximum Samples** values only represent the minimum and maximum number of possible samples, respectively. Cinema 4D will estimate a sample count for each image pixel between these two values, thereby using the **Accuracy** value as a reference. The closer this value lies to 100%, the more the number of samples will lean towards the **Maximum Samples** value. The following steps can be followed to achieve a high-quality shadow:

1. Increase the **Accuracy** setting to 100%. This will force Cinema 4D to always use the maximum number of samples.
2. Increase the **Maximum Samples** value incrementally until the test render looks the way you want it to.
3. Set the **Minimum Samples** value to approx. 1/6th of the **Maximum Samples** value.
4. Reduce the **Accuracy** value by about 50% to allow Cinema 4D to vary the sampling accuracy for each pixel.

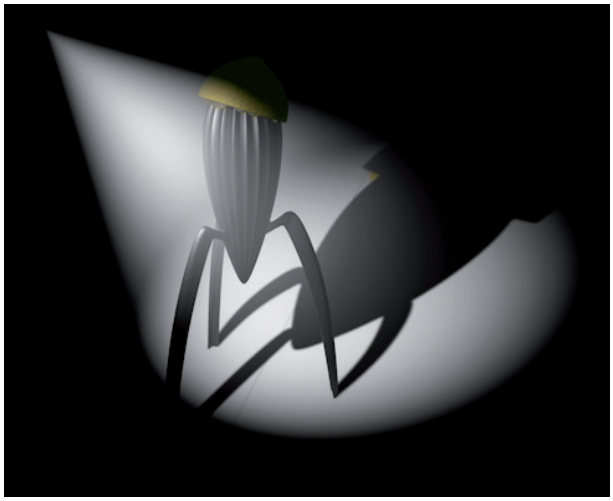
If these steps are followed, a good compromise between quality and render time can be found.

The **Area** shadow's look is also affected by the size of the surface that is used by the light to calculate the shadow. The corresponding settings can be found in the light's **Details** menu. The **Area Shape** setting can be used to select the desired shape. The shape's size is defined using the **Size X**, **Size Y** and **Size Z** values. The larger the surface, the softer its Area shadow will be. The smaller the surface is, the more the shadow will resemble a Hard shadow.

When using an **Infinite** light, the shaded area will be defined somewhat differently. An **Infinite Angle** value will be made available. The larger this value is, the softer the shadow's edge will be.

7.2.10.4 Visible Light

A natural environment is made up of more than just light and illuminated objects. There is also an atmosphere that lies between these elements. This atmosphere is not pure and can therefore interact with light, for example, if it contains smoke or fog, which disperses light. In extreme cases, the emitted light itself will be made visible, like a headlight in fog.



This is called visible light and can be defined using the **Visible Light** setting in the light's **General** menu. This setting is not available for **Area**, **Parallel** or **Infinite** light types.

Regardless which option you choose, this effect will always be controlled using radii that can be modified interactively using the handles in the Viewport or using the **Visibility** menu's settings. The **Outer Distance** value defines the distance from the light source at which the visible light ends; the **Inner Distance** value defines the distance around the light source within which the maximum effect is visible. A falloff will automatically occur between these two radii. If you do not want the intensity to fall off, disable the **Use Falloff** option. Otherwise, the **Falloff** value can be used to define the degree of falloff by the time the **Outer Distance** is reached. A value of 80% will result in 20% visible light remaining at the defined **Outer Distance** and then ending abruptly.

When using a **Spot** light, a radial falloff can be enabled. The **Use Edge Falloff** value can be used to define the peripheral limitation of visible light. The smaller the value, the brighter the visible light will be at the edge and the more defined the visible light will be. Normally, the visible light will assume the color of the light source. You can also define a different color by enabling the **Use Gradient** option and defining a color gradient. The left end of the gradient defines the color at the light source, the right end defines the color at the outer edge of the visible light.

Additional colors can be added by clicking just below the gradient to add color handles. Superfluous handles can be removed by dragging them up and over the gradient. Each color can be modified by clicking on the corresponding color handle and opening the **Color** options by clicking on the small triangle at the left of the gradient bar. Enable the **Colored Edge Falloff** option if the distribution of color should not be dependent on the distance from the light. The left gradient color will then run along the light's Z axis. The right gradient color will appear at the outer edge of the visible light.

Normally, the distance radii have a spherical shape. The **Relative Scale** values can be used to create an elliptical shape.

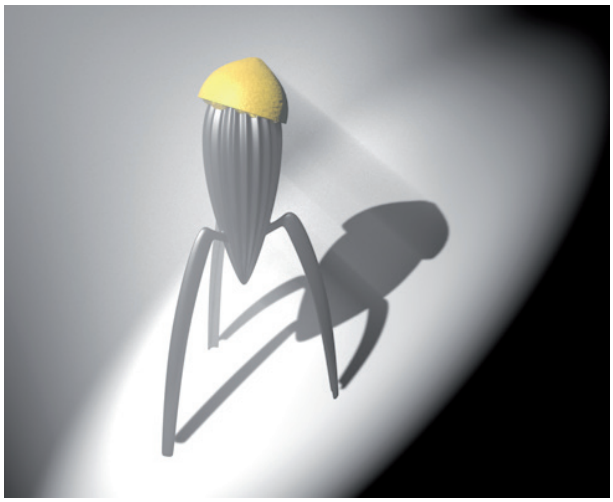
The **Brightness** value serves as a multiplier for the light's intensity and defines the brightness of the simulated fog. If you want to create more of a light fog than thick smoke, reduce the **Brightness** and increase the **Dust** value accordingly.

If 'banding' should occur (subtle brightness and color variances in the visible light), the **Dithering** value can be modified to counter this. Increasing this value will add random noise and mix the visible light's colors.

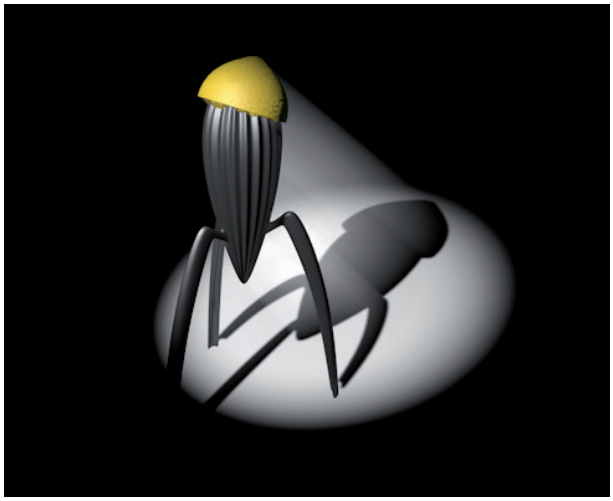
If the **Additive** option is enabled, the brightness of various lights will be added to the visibility properties where they overlap. This can quickly lead to over exposure, which is why this option is disabled by default. Similar effects can result if a visible light cone is viewed from the front. Leave the **Adapt Brightness** option enabled to prevent brightness from being added for fog.

The **Sample Distance** setting is only relevant for **Volumetric** or **Inverse Volumetric** visible light.

If **Visible Light** is set to **Visible** (**General** menu), a very simple algorithm will be used to display a type of fog. Objects within this fog will not cast shadows. Visible rays of light shining through a foggy forest, for example, cannot be realized using this setting. In this case, a volumetric visible light must be used.



The **Sample Distance** value defines the distance at which these samples should react to objects. The smaller the value, the more precise the sampling will be – and the longer it will take to render. When in **Inverse Volumetric** mode, the calculation will simply be inverted. Previously shadowed regions will be depicted in fog. This can, for example, be an interesting effect when rendering logos because it looks like the light is being emitted from the logo itself.



7.2.10.5 Lighting Options

At the bottom of the General menu there are several options that can be enabled or disabled to affect the selected light.

- **No Illumination** Enables or disables highlights and shadows. Only the visible effects of the light will be rendered, which can help reduce render time.
- **Show Illumination** Displays additional lines and handles in the Viewport, which can be used to modify the falloff radii or the cone angle of a **Spot** light for example.
- **Ambient Illumination** An alternative shadow algorithm will be enabled. The surfaces that lie within the light's area of influence will no longer be shaded in relation to the angle in which the light falls but will only receive a general lighting. This can, for example, be used to generally brighten or darken parts of the scene.
- **Show Visible Light** Affects the lines and handles that can be used to modify the radii for visible light in the Viewport.
- **Show Clipping** Displays additional guides to better show the intervals for clipping in the Viewport. Clipping can be used to restrict the effect of light and visible light. This setting will be discussed when the Details menu's settings are described.
- **Separate Pass** Enable this option if this light's effects (shadows, highlights, etc.) should be rendered to a separate layer. **Multi-Pass** rendering must also be enabled in the **Render Settings** menu.
- **Export to AFX** When compositing files are saved, the light's position in 3D space will be exported for use in After Effects.

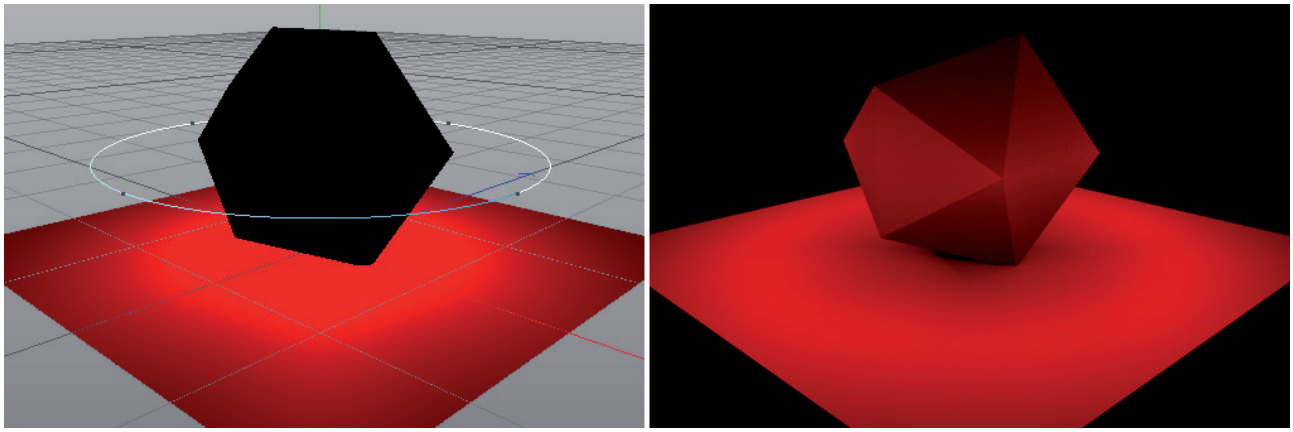
The remaining options affect the three lighting characteristics. We already mentioned that all real lights have a physical size and can therefore be reflected in a reflective surface. This is known as 'specular highlight'. However, most of the Cinema 4D lights do not have a physical size. A trick is used to create highlights. Highlights are simulated using a surface material property and can be defined independently of the light's intensity or color. We will discuss this further in the material system section. If you do not want highlights to be created, simply disable the **Specular** option.

The actual light effect is made up of shading on the surfaces on which the light falls. Various levels of brightness are generated as well as colors, depending on the angle of the light and the surface type and color. For example, if you only want to add a highlight to a surface without adding brightness from the light, disable the material color option.

We already mentioned that real light is reflected throughout a room and among objects from objects' surfaces and via refraction. This is why it's nearly impossible to find a completely black region in any illuminated room. However, this is not the case with Cinema 4D lights – they only illuminate the surfaces on which they shine. **Global Illumination**, which can be enabled in the Render Settings menu, can simulate the real-world lighting effect. The light can be dispersed across surfaces and onto other objects as well. **Global Illumination** (**GI**) can be enabled or disabled for individual lights. Hence, if you only want to see the GI effects of a single light, disable **GI** for all other lights.

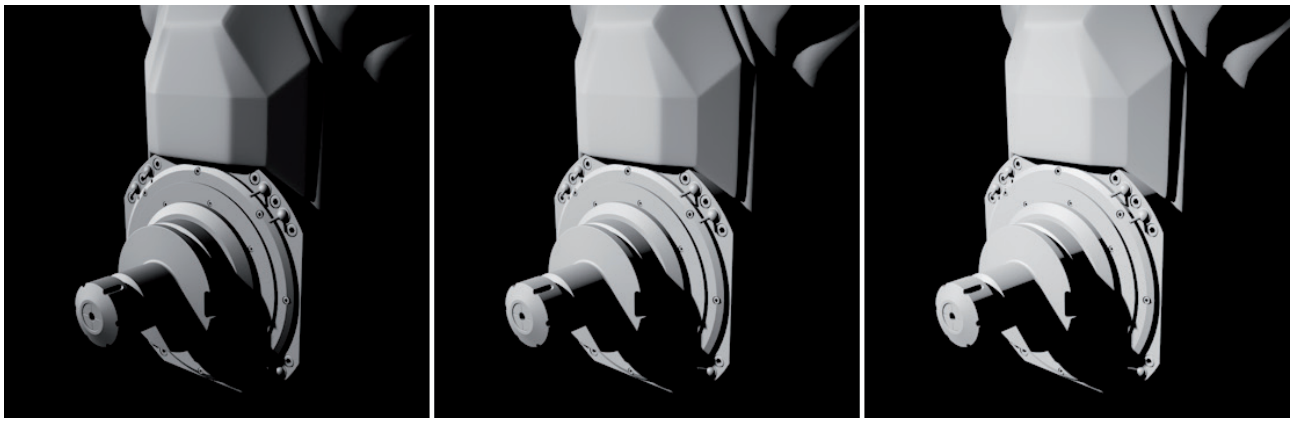
7.2.11 The Details Settings

Most of the **Details** menu's settings affect the light's shape or the shape of the emitted light. We already discussed the Size values, on which the shadow's calculation is based. For example, an **Area** light can have the shape of a rectangle, a disc or a hemisphere. You can even assign a polygon or spline object using the **Area Shape/Object/Spline** option. Parametric object must first be made editable (converted) before they can be used.



When using **Spot** lights, various angle settings will be made available. The **Outer Angle** value defines the maximum peripheral size of the light cone, or the light pyramid when using square spotlights. If the **Color** setting is enabled, the light's intensity will have a inner to outer falloff within the light cone. This falloff can be defined using the **Inner Angle** and **Outer Angle** values. The similar these values are, the harder the light will be. The **Aspect Ratio** setting can be modified if you do not want a perfectly spherical or square cross-section.

The **Contrast** value defines the degree of surface shading in relation to the light's angle of incidence. Values greater than 0% will also brighten regions not illuminated at a perpendicular angle by the light. Values of less then 0% can also be used. This will result in the illumination being withdrawn until only those regions being illuminated that lie almost perpendicular to the light.



If the light casts a shadow, its illumination properties can be omitted entirely. If Shadow Caster is enabled, only the light's cast shadow will be rendered, without any illumination. Of course this only makes sense if the scene is illuminated by at least one other light.

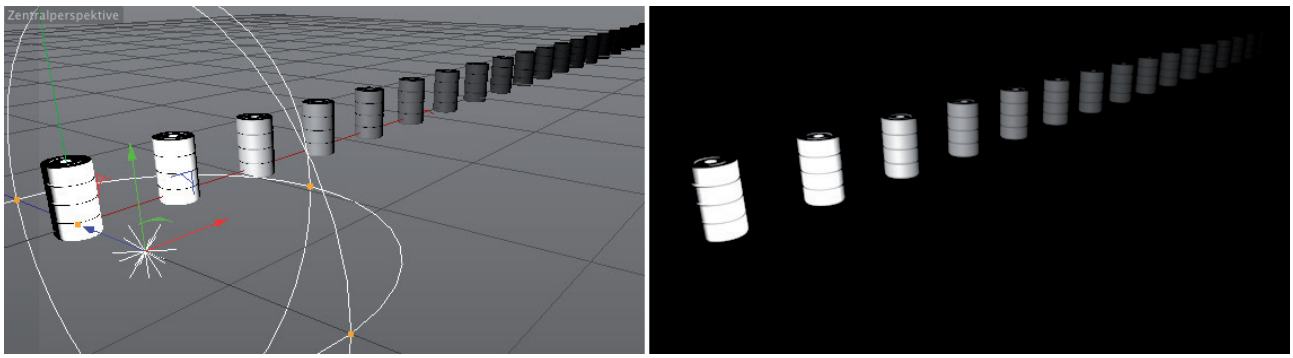
7.2.11.1 Falloff and Light Intensity

If you examine real-world lights a little closer, you will see that a light's intensity depends on the distance between objects and the source of light. The light emitted by a flashlight will diminish greatly after just a few meters. This also has to do with the fact that light waves are dispersed by the atmosphere surrounding the flashlight. In addition, the light is not bundled but spreads as soon as it is emitted by the source. The farther the light is from the source, the more the light spreads and weakens.

The light's intensity is disabled by default in the light's **Falloff** setting, which means that our flashlight could illuminate the surface of the moon because its intensity would never diminish. Of course this is not realistic at all. Therefore, when simulating artificial sources of light, you should always select a corresponding **Falloff** option. Sunlight, however, does not need a falloff.

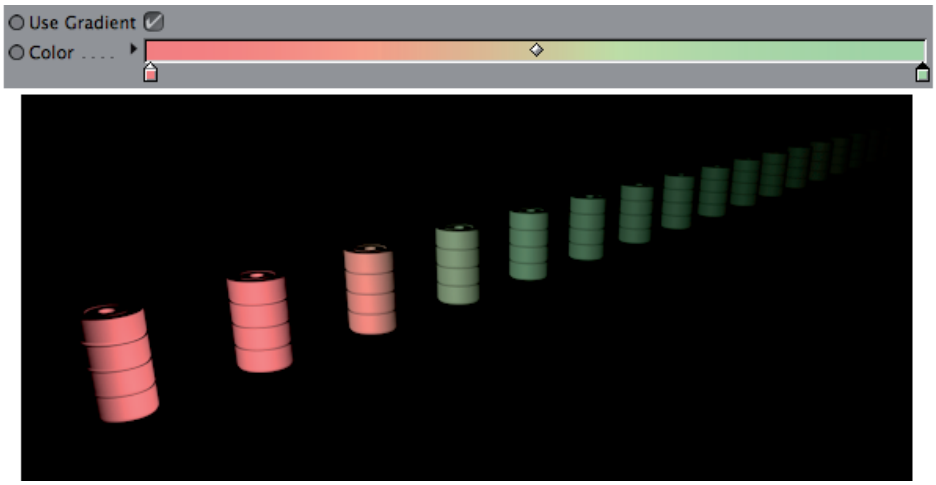
Falloff can be calculated using various algorithms:

- **Inverse Square (Physically Accurate)** Ensures that the light's intensity always diminishes at a square of the distance from the light source. This effect is defined using the **Outer Distance** value. This value can be equated to the size of the light source, similar to a glass body for your light. A surface that lies exactly this distance from the light will be illuminated with the exact same intensity as is defined by the **General** menu's **Intensity** value. Avoid placing objects closer than the defined distance. The intensity increases dramatically within this distance and will far exceed the light's actual intensity. This setting simulates natural light falloff very closely and is therefore recommended for use, keeping the **Outer Distance** value in mind.



- **Inverse Square Clamped** This method also diminishes the light's intensity at a square of the distance from the light source. However, the light's brightness within the defined Outer Distance is restricted to the light source's intensity. This prevents objects that lie very close to the light source from being overexposed.
- You can also use the **Linear** option if you need more control over the falloff. This will give you access to the **Inner Distance** and **Outer Distance** options. The **Outer Distance** value defines the point at which no more light can be seen. The **Inner Distance** value defines the radius around the light source within which the light's intensity will be exactly as defined in the **General** menu's settings. The more similar these values are, the more bundled the light will be – until it ends abruptly.
- If the **Step** option is selected, an even more radical behavior will result. The light's intensity remains constant until the **Outer Distance** value is reached, at which point it will end abruptly.

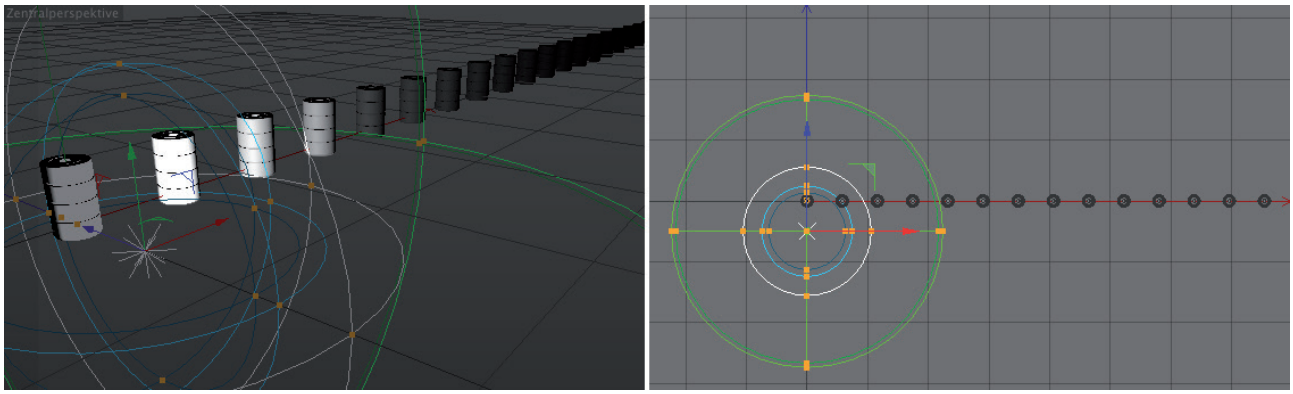
If the **Use Gradient** option is enabled, a gradient color can be defined, which will flow in the direction of the falloff. The emitted light will assume the gradient's left-most color and the gradient's right-most color will appear at the end of the falloff.



This replaces the color settings in the General menu. When using a **Spot** light, a **Colored Edge Falloff** can also be enabled. The color gradient will also be applied to the intensity range between the **Inner Angle** and **Outer Angle** values. You should already be familiar with this principle from the **Visibility** section. If the Z Direction Only option is enabled, the light's emission can be restricted to the light's Z axis.

Clipping can be used to basically cut away lighting. A car's headlight in fog, for example: The light source lies in the light's housing and should only be visible outside of the housing's glass.

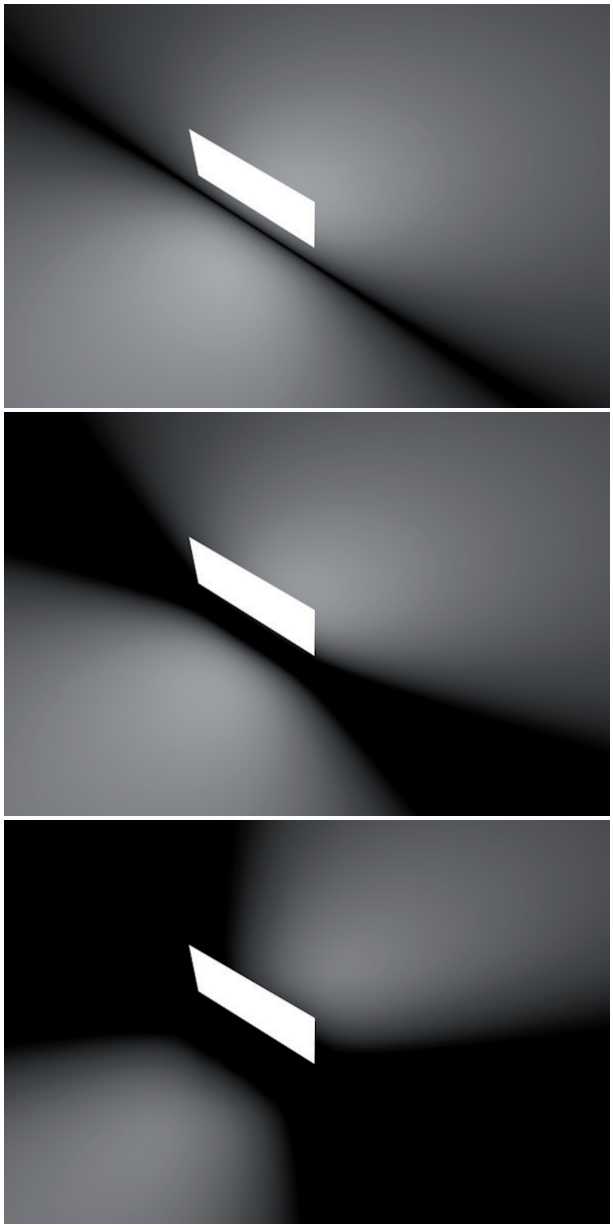
The **Near Clip** setting can be used to define a range that the light will skip before it becomes visible. The **Far Clip** setting lets you define such a range for the light in the distance. These settings can also be combined.



If clipping should also affect shadows, enable the **Clipping Influence** option in the **Shadow** menu. The clipping feature is not available for **Area** or **Infinite** lights.

7.2.11.2 Area Light Characteristics

We already mentioned that **Area** lights produce the most natural-looking light because of their physical size. When combined with **Area** shadows, they offer the highest quality lighting for direct lighting situations. The **Falloff Angle** value can be used to restrict the angle of emitted light. This is particularly important for two-dimensional surface shapes, e.g., **Rectangle** or **Disc**. Similar to a **Spot** light, smaller **Falloff Angle** values will bundle and focus the light correspondingly more until it is emitted almost perpendicularly and parallel from the surface if very small values are used.



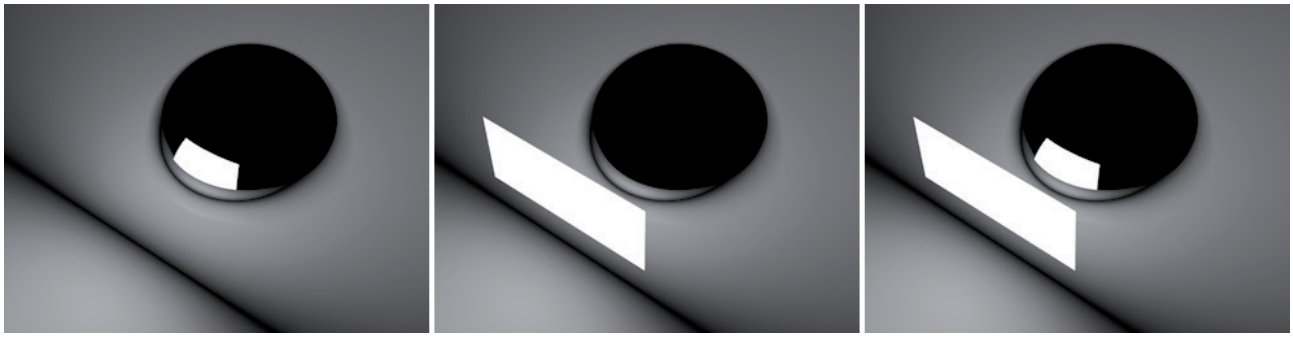
However, the **Area** light is not designed for such extreme bundling of light and therefore does not produce a useful result if very small values are used. If very large values are used, e.g., 180°, the light will even be emitted from the sides of the shape. If you want to restrict the emission angle, you can also position a light housing, or other shapes that direct the light, around the **Area** light. If shadows are cast, they will be rendered like real-world shadows that are produced by a light shining through a housing.

An **Area** light illuminates a scene by simulating numerous individual light sources that are placed on the shape's surface. The more of these helper lights that are used, the more homogenous the **Area** light's illumination will be, especially on objects that lie parallel to the direction of light or perpendicular to the light's shape. The **Samples** value can be used to fine-tune the quality of light and the specular highlights of the **Area** light. Values between 16 and 1,000 can be defined, which will not drastically increase render time.

Enabling the **Add Grain (Slow)** option will add random noise to the light. If the **Samples** value is too low, these will appear as individual high-contrast points. Increasing the **Samples** value will produce a correspondingly finer grain and blurring of the effect, which will also increase render times quite a lot. One way in which adding grain can be used is to reduce specular highlight issues with **Area** lights.

If the **Show in Reflection** option is enabled, the area shape will actually reflect in surfaces. This is the most natural type of interaction with materials. A simplification or supplementation of this would be to enable the **Show in Specular** option, which will let the Area light also affect the specular properties of the **Reflectance** channel.

In this case, enable the **Show in Reflection** option, which will make the light's shape visible in reflections. If you also want the light's shape to be rendered, enable the **Show in Render option** as well. Normally, all light sources are not visible for the camera as long as the lights have no visibility parameters.



The advantage is, of course, that light sources can be places freely without obstructing the view. As a rule, it doesn't make sense for an **Area** light to be rendered as bright, illuminating geometry.

The intensity or brightness of an **Area** light's reflection or specularity in a rendered image depends on the intensity of the light itself. You can also make its brightness in reflections or in the image independent of this factor by using the **Visibility Multiplier** setting. This way, less intense lights can be reflected strongly.

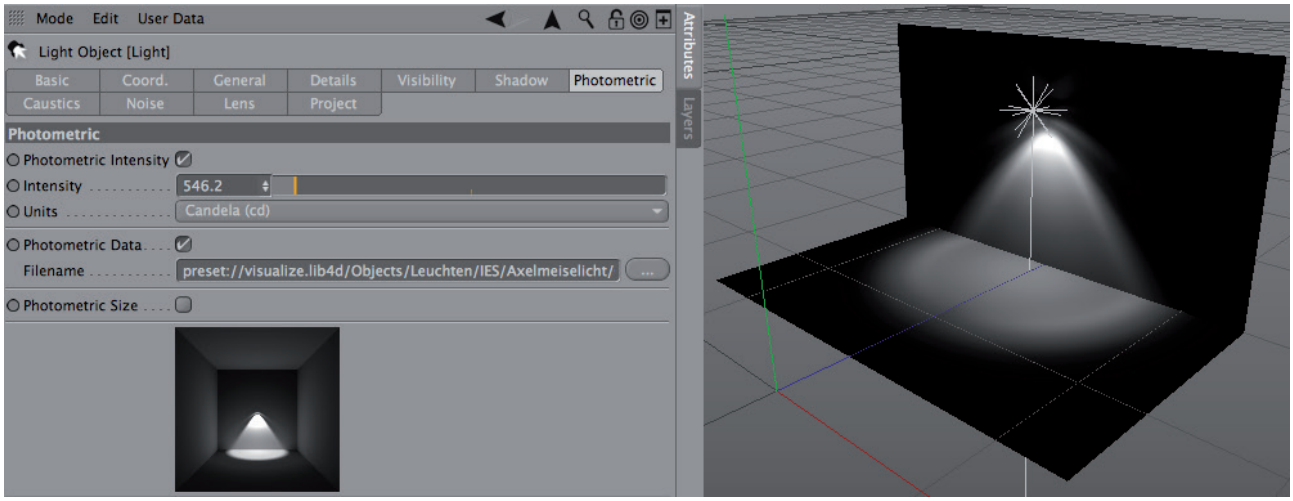
7.2.12 The Photometric Settings

We have already discussed how a light's intensity can be defined. The concept is based on common image editing techniques and works by modifying the brightness of image pixels. Real light sources can also be measured but they use different units. Cinema 4D uses Candela, Lumen or Lux.

- **Candela** is the intensity emitted by a light in a specific spatial angle. The type of light emitted and the size of the light source are irrelevant.
- **Lumen** is the strength of the total amount of emitted light from a light source. The amount of light for a spotlight, for example, will be measured within its light cone. If the light cone's size is reduced and the Lumen value remains constant, the light's brightness will increase accordingly.
- **Lux** encompasses the distance from a light source and the size of the light source. A light source double the size will therefore emit twice as much light if the **Lux** value remains constant.

Of course, these units can only be defined accurately if, for example, data from actual light sources is used. If the **Photometric Intensity** value is enabled, the required **Intensity** and **Units** values can be defined. An even more accurate method is to import data from a file, which Cinema 4D can do using the **IES** file format.

In order to use these files, the light **Type** must be set to **IES** in the **General** menu. The **Photometric Data** option will automatically be enabled in the **Photometric** menu. Click on the button with the three dots to the right of the Filename field to select the desired **IES** file. Several test files are included with Cinema 4D and can be loaded from the *Content Browser's* **Presets / Presets / Lighting / IES Lights**.



The same preview image as the file you select will be displayed in the **Photometric** menu.

IES files contain light information from real-world lights, which make them ideal for use in architectural projects or for designing trade show booths, for example, to accurately simulate real lights. All settings with regard to light falloff and intensity are derived directly from the **IES** files and do not have to be adjusted manually. **IES** lights also contain additional meta information, e.g., regarding wattage of the illuminant, light designation or manufacturer information. This information can be found in the **Information** section of the **Photometric** menu.

Numerous **IES** files also have information regarding a light's physical scale. For these files you can enable the **Photometric Size** option. The **IES** light will behave like an **Area** light, which means it can also be mirrored. In addition, a more natural light will be produced.

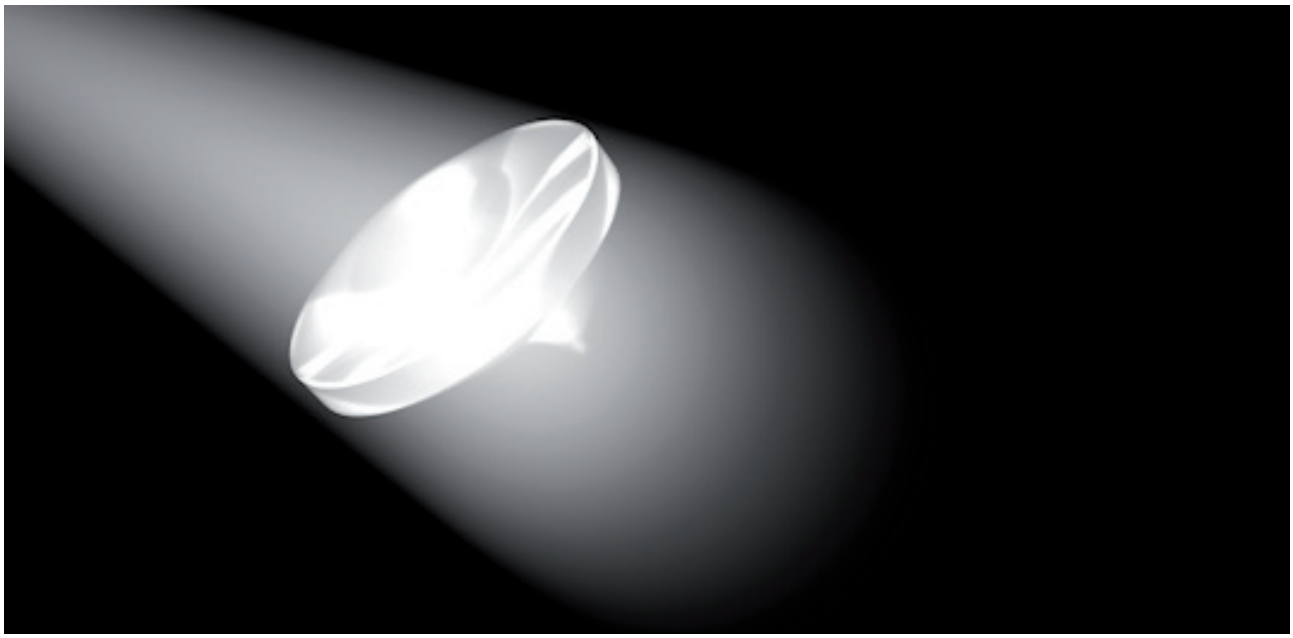
When using **IES** lights, note that the primary direction in which the light is emitted is often the same as the light source's Z axis. Light sources that emit light in 360° are quite rare, so spotlights and standing lights are, as a rule, oriented along the Z axis defined in the IES file. So rotate the light source accordingly to adjust its Z axis.

7.2.13 The Caustics Settings

Caustics is the refraction or bundling of light that produces bright regions that exceed the light's own brightness settings. We are surrounded by exactly these effects in the real world. Caustic rays can occur anywhere light is reflected or refracted when passed through corresponding materials or surfaces. A very common example are the wavy lines at the bottom of a filled swimming pool or the focal point of a magnifying glass.

CIEMA 4D lights can also be used to simulate this effect, which can be very interesting on reflective surfaces or glass in particular. Two main options are available in the **Caustics** menu with which this effect can be simulated. Enabling **Surface Caustics** will create the effect caused when light is bundled on mirrored surfaces or when it passes through a transparent object that refracts the light. This is also the option that you will use in most cases when generating caustic effects (Project **"CausticsExample"**).

Volume Caustics is used only in special cases when creating caustics within a volumetric visible light (Project **"CausticsExample"**).



You can probably imagine how long such an effect will take to render. This effect is also much more rare in the real world, with the exception of a mirrored disco ball rotating in a smoke-filled disco. Both Caustics effects can be combined as well.

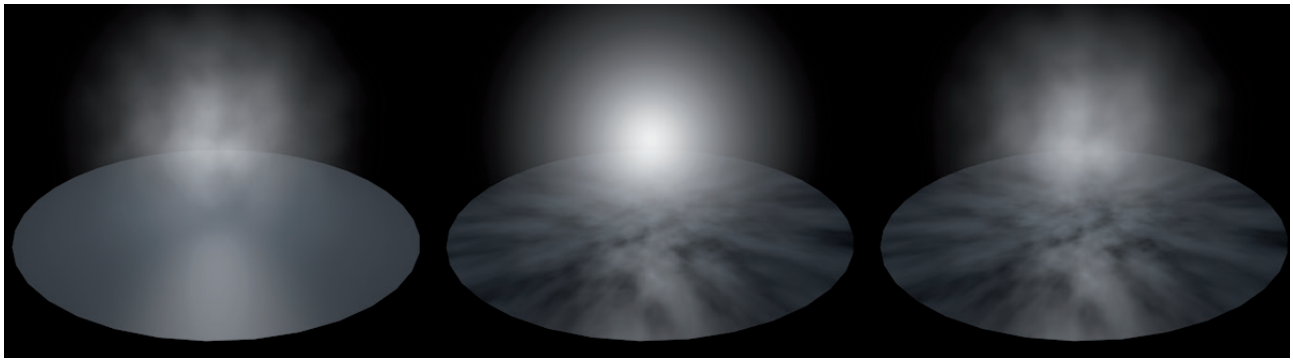
In both cases, simulated light particles – **Photons** – are created in the light source and emitted in the direction of the rays of light. Each photon has energy, which is defined as a percentage of the light's strength or intensity. Whenever multiple photons pass through refractive surfaces, or are within a simulated fog using Volume Caustics, and collide, a bright point will be created. If too few photons are used, the probability that multiple photons will be bundled will be reduced accordingly. Since the calculation of the photons has no major effect on the render time or on the amount of memory required during rendering, the direction and number of photons should be optimized to achieve the best result. It's best to use Spot lights in conjunction with the **Caustics** feature. The lights should be pointed at the object used to create the caustics.

The photons will also be interpolated using radii and sampling settings in order to achieve soft caustic structures using as few photons as possible. These settings can be found in the Cinema 4D material system – not in the light's settings. You can also define which surfaces – if any at all – should be used to generate and receive caustics and how these should be interpolated. We will discuss this in the material system section. The **Caustics** effect must also be enabled in the **Render** Settings menu. Otherwise no caustics will be calculated, even if this option is enabled in the light's menu.

If **Volume Caustics** is enabled for a given light, a **Falloff** option can also be selected, whose function is the same as the Falloff function in the **Details** menu. Caustics will then be calculated according to the **Inner Distance** and **Outer Distance** values. An unrestricted calculation will not take place, even if **Falloff** is set to **None**. Otherwise, the calculation of rays can, theoretically, be endless. As previously explained, it's better to restrict caustics to those regions of the image in which they are needed.

7.2.14 The Noise Settings

Normal lights as well as visible lights are both relatively homogenous despite their various Falloff settings. For example, if you want to vary the lighting or the structure of visible light, you will need to use the settings in the **Noise** menu. The Noise menu's settings are used to define whether the light's **Illumination**, **Visibility** or **Both** should be affected.



The Type setting offers various mathematically generated structures that can be applied, whose effect is displayed below the dropdown menu as a preview. The **Octaves** value defines the depth/precision of the effect. Smaller **Octaves** values will produce correspondingly softer, less detailed structures.

Because these structures have a spatial mass they can also be scaled three-dimensionally. This can be done using the three **Visibility Scale** values. The direction of scale will be calculated parallel to the world axis. The **Noise** structure is always endlessly large and will fill the light's area of influence completely.

The **Illumination Scale** value is an additional multiplier for the effect's scale and is only used to affect the size of the structure with which the light is superimposed. This makes it possible to use noise structures of different sizes for **Illumination** and **Visibility** if **Type** is set to **Both**. The larger the **Illumination Scale** value, the finer and more detailed the structure will be. This effect for the **Visibility Scale** values is exactly opposite – the smaller the value, the finer the structure.

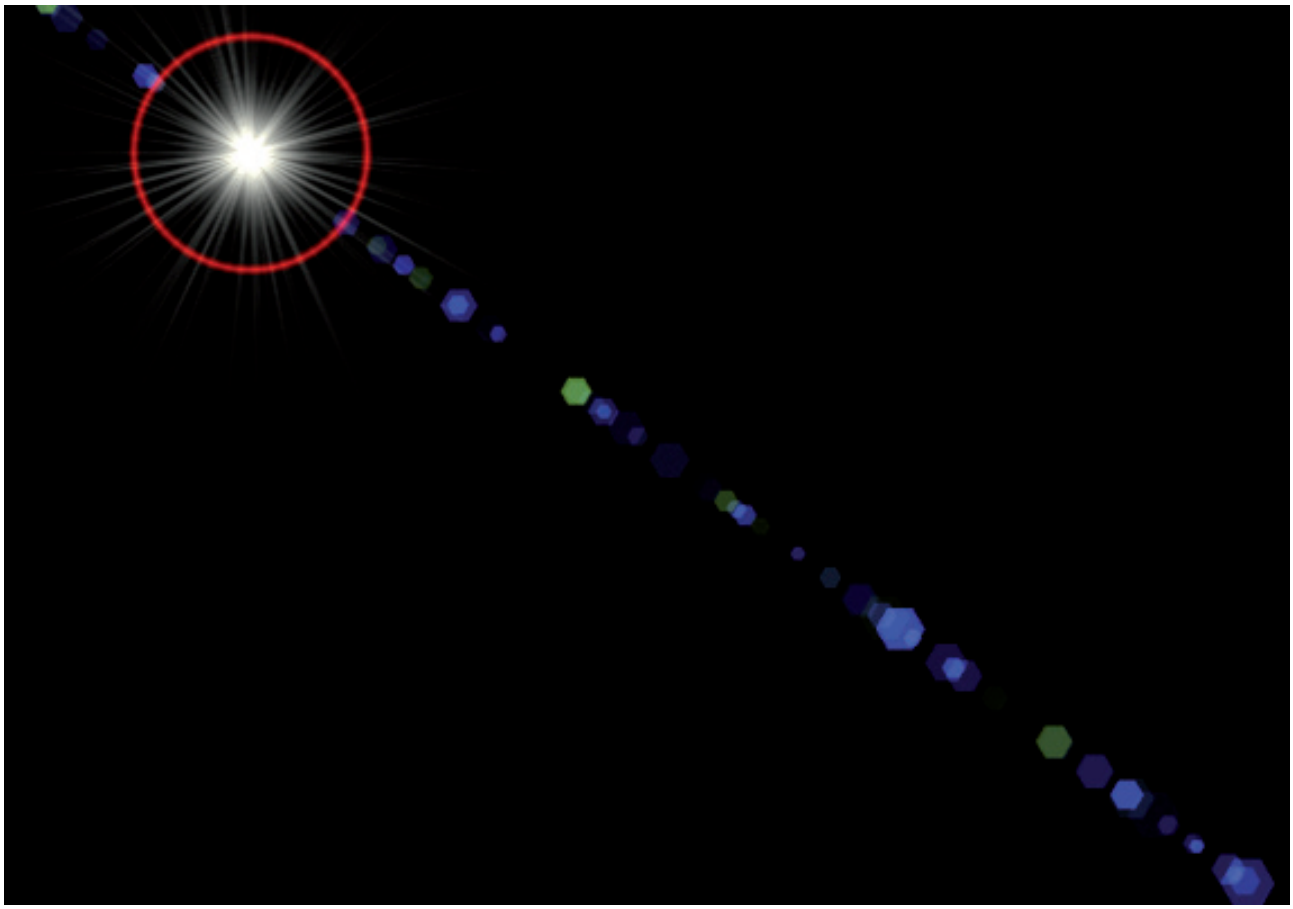
The **Brightness** and **Contrast** values define just that – the structure's brightness and contrast, respectively.

If the **Local** option is enabled, the noise pattern will be bound to the position of the light source. If the light source is moved, the pattern will move with it. If this option is disabled, the noise pattern will be bound to the world coordinate system, which means that it will act autonomously. As a rule, this option should be disabled.

The noise structure can also be animated, which is, of course, only relevant for animations. The **Velocity** value defines the structure's overall fluctuation. However, the visible noise is animated using the **Wind** and **Wind Velocity** settings. The Wind vector defines the direction of the wind. Use the **Local** option to define which coordinate system will be used. The **Wind Velocity** defines the strength with which the simulated wind will blow.

7.2.15 The Lens Settings

The **Lens** menu's settings are used to simulate visual effects that can occur when light reflects or refracts in a camera lens or on a film negative or CCD chip. These can, for example, be lens flares or glowing effects around very intense sources of light.



Note that this is a post effect that is only available in conjunction with the Standard renderer or the Physical renderer. In addition, this effect will also be visible in reflections and therefore has corresponding restrictions such as with the **Glow** effect when using a standard material.

The **Glow** and **Reflexes** menus contain numerous default settings for glow and reflexive effects, which can be used to make the otherwise invisible light sources visible in the rendered image(s). You should note that the light defined in the **Type** setting can have an effect on the lens effect(s). The glow and reflexive effects can only be seen in their full intensity when using the default settings when the light source and the light it emits are viewed directly. If the effect should be generated independent of the light type and angle of view towards the light, disable the **Use Light Parameters** option.

If **Use Light Parameters** is enabled, the light's color and intensity will also be included in the creation of the lens effect. Otherwise, the **Brightness** and **Scale** values can be used to define the intensity of the effects. Test renders should definitely be made. We will discuss these in detail in the materials section. Glow effects that generate rays of light can be rotated in the right direction using the **Rotation** value.

The **Aspect Ratio** value can be used to create elliptical effects if set to a value not equal to 1.

The **Fade if Behind Object** option defines whether or not the glow and reflexive effects should be seen if the light source lies behind an object from the observer's point of view.

A similar effect is produced if the **Fade if Near Border** option is enabled and is particularly well suited for use with the Reflexes effect. In the real world, this effect's intensity increases the closer it lies at the center of the image. If this option is enabled, the brightness of the **Glow** and **Reflexes** effects will be reduced accordingly as they approach the image edge.

If **Fade if Approaching Object** is enabled, glow and reflexive effects will not be disabled abruptly if the light source is obscured by an object. The effect's brightness will start being reduced just as the light source starts to be obscured by the object. If these effects should also be influenced by the distance between the light source and the observer, the **Glow Distance Scale** and **Reflex Distance Scale** options should be enabled.

To calculate this effect, a **Reference Size** must be defined. The effects will be rendered at their normal size at the defined **Reference Size** distance. The size of the effects will diminish the farther away they are. If the light source approaches the observer, the glow and reflexive effects will become correspondingly larger (visually).

In addition to the numerous **Glow** and **Reflexes** settings, individual effects can also be created. To do so, select one of the default options and click on the respective **Edit** button.

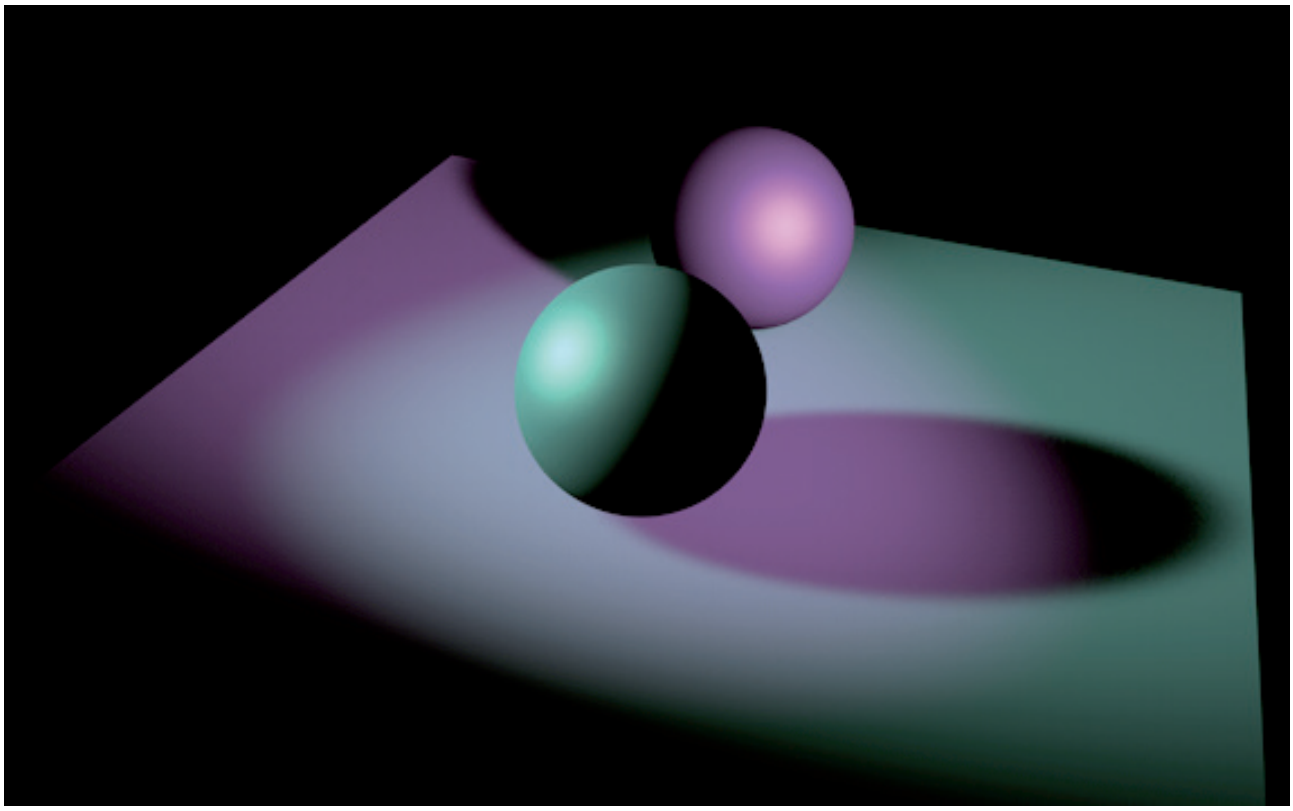
For the glow effect, the **Glow** itself as well as the **Ring** and **Beams** it emits can be modified. The **Glow** and **Beams** features both have several options from which to choose to define their look. The **Size**, **Color** and **Aspect Ratio** (abbreviated with **R**) can be modified for each feature individually. The **Beams** menu offers additional settings: **Angle**, **Thickness**, **Beams** (number of), **Breaks** and **Width**.

Enabling the **Random Dispersion** option causes the beams to be generated in a random sequence, i.e., the ring will be irregular. The length can also be generated randomly by enabling the **Random Beam Length** option. Enabling the **Star-Like** option will vary the beam density towards the center of the effect. The beams will look more like jagged edges, which can especially be observed if the number of beams is reduced.

The **Reflexes** settings have a similar structure. Here, the reflex element you want to modify must be selected by either clicking on the **Add** or **Rem.** option or by entering the corresponding **Element** value. **Position**, **Size** and **Color** can be modified individually for each element. The **Lens Type** setting contains numerous options from which you can choose. What the modifications look like can be seen in the preview image at the right.

7.2.16 The Project Settings

In real life, it's practically impossible to make a light affect only specific objects and exclude all the rest. However, this can be done easily in Cinema 4D. The **Project** settings menu has a **Mode** setting that offers an **Exclude** and an **Include** option – depending on whether you want to exclude or include specific objects for lighting.



Elements can be added to the list by simply dragging and dropping them from the ***Object Manager*** into the list. You can also click on the cursor symbol at the right of the list and then click on the objects you want to add in the ***Object Manager***. Click on the cursor symbol again do deactivate this selection mode.

You don't have to add each object to the list individually. You can simply select the top-most object in a given hierarchy and add it to the list.

The far-right icon in the Objects list for this object is the **Hierarchy** icon. If this icon is enabled, all of this object's sub-objects in the ***Object Manager*** will be affected by the inclusion/exclusion. Disable this icon if you do not want the sub-objects to be included in the inclusion/exclusion.

The other three icons from left to right are: **Material Color**, **Specular Highlight** and **Shadow**, respectively. These icons can also be enabled or disabled individually. The effect depends on the **Mode** defined in the **Project Settings** menu. If **Mode** is set to **Exclude**, the objects in the list (including their sub-objects, if applicable) will be excluded entirely from this light's illumination. If the first **Material Color** icon is disabled, the object will receive shadow. Specular highlights and shadow casting will, however, still be disabled. These icons can be enabled and disabled to create different combinations.

If Mode is set to Include, only the objects in the list (including their sub-objects, if applicable) will be illuminated by this light. The icons can also be enabled or disabled to create the desired effect. Right-clicking on an item in the list will open an context menu, which, among others, lets you remove one or all elements from the list.

The **PyroCluster** options at the bottom of the Project menu are only relevant when you combine a particle system with PyroCluster materials, e.g., to simulate fire, clouds or smoke. If both options are enabled, the light will illuminate these PyroCluster clouds and generate shadows for them as well.

7.2.17 How to Create Targeted Lighting

We have already discussed several types of lights whose implementation can be made more effective by rotating their Z axis towards the objects to be lit. This can, however, be somewhat arduous. When working with an animated character, for example, that must always be kept in the spotlight while moving. This is why there are features available that can help align light sources to objects automatically. These are called Expressions in Cinema 4D, which can be added via the ***Object Manager's*** **Tags**/**Cinema 4D Tags** menu.

Expressions are small 'scripts' that can be used to control the behavior of objects during an animation. For example, the movement of an object along a spline, a random vibration or even automatically aligning one object to another can be done using Expressions.

We will use the **Target** Expression, which we can assign directly to a light source. The **Target** Expression automatically uses the object's Z axis to align itself to an object. All that needs to be done is to define the object to which it should be aligned. To do so, simply drag the object to which the light should be aligned into the Expression's **Target Object** field. In most cases, this is all that's required. No matter where the Target Object goes, the light (or object) will follow. This is very useful when working with **Spot** lights, which is why Cinema 4D also offers a default **Target Light**, which contains a **Target** Expression by default. A **Null** object will also be created automatically, which is already linked in the light's **Target Object** field. Because the **Null** object is not visible for rendering, it can be easily used as a target for the **Target Light**.

▶ *See: Exercises for light sources*

SUMMARY: LIGHT

- All light sources always create direct light by default.
- Additional lights are required to create indirect and diffused light, or Global Illumination can be enabled.
- **Omni** lights emit light in all directions.
- **Spot** lights emit light in a specific direction in the shape of a cone, cylinder or pyramid.
- **Parallel** and **Infinite** lights simulate bundled light or light coming from a distance, e.g., sunlight.
- **Area** lights simulate a three-dimensional shape that is used as a body of light; they produce the most natural-looking light.
- Only **Area** lights can be reflected in surfaces to realistically simulate specular highlights.
- The remaining light sources use material properties to simulate specular highlights on the surfaces they illuminate.
- **Shadows** are a separate light property and must be activated separately.
- **Soft Shadows** are based on a depth mask that is saved as part of a bitmap. Quality, render time and required memory is determined by the bitmap's resolution.
- **Soft Shadows** can be optimized by pointing the light directly at the object it illuminates.
- **Hard Shadows** are always calculated mathematically correct and are therefore always rendered with perfectly sharp edges.
- **Area Shadows,** like **Area** lights, use a shape from which the shadow is rendered, which creates very realistic-looking shadows but also require the most render time of all shadow types.
- The accuracy of **Area** shadows depends on the number of **Samples** used to make up the effect.
- **Area** shadows become softer when the light source's shape increases and vice versa.
- Lights are generally not visible for rendering. Only the illumination can be seen by default.
- The falloff of the light's intensity in relation to the distance between the light and the objects can be defined using the Falloff menu.
- Atmospheric impurities can be added using visible light.
- Objects can only cast shadows into visible fog if volumetric visible light is used.
- Additional patterns can be superimposed using Noise, which can be used to make the visible light look like smoke or clouds.
- Basic light properties such as shadow casting, specular highlights and Global Illumination can be configured individually.
- **Clipping** can be used to restrict a light's reach to a range near the light.
- **Caustics** can also be used to simulate bundled light. This feature must be enabled for the light itself and in the Render Settings menu. An object's material properties can also be used to create caustics.
- Glow effects and lens flares can be created separately and independent of other lighting properties.
- Real-world light sources can be accurately simulated using **IES** files, which contain photometric data regarding emission and intensity of a given light.
- Individual objects can be excluded from being illuminated by a specific light or lights by adding them to the corresponding list in the light's **Project** menu.

8 Environment Objects

All 3D scenes are empty to begin with. Nothing but pitch black 3D space. **Environment** objects are designed to help fill this space with useful objects such as a **Floor**, **Sky**, fog or a **Background**. These objects are located in the main **Create/Environment** and in the **Create/Physical Sky** menus, or they can be accessed via the top icon palette.

8.1 Floor Object

As a rule, 3D objects will not simply float in mid air. They will generally lie on a floor. This is especially true for outdoor scenes that typically have a floor and sky, which are ultimately separated by a horizon. The Cinema 4D **Floor** object will automatically extend indefinitely for rendering – so don't be fooled by its relatively small size in the Viewport.

Otherwise, this object can be moved and rotated just like any other shape. Hence, a **Floor** object can also be used to create an infinitely high wall, for example, if positioned vertically. It can also be positioned over the observer's view and have a photo or material assigned to it to simulate a simple sky or cloud cover.

8.2 The Sky Object

This object is not visible in the Viewport but is visible for rendering. The **Sky** object is a sphere with an infinitely large radius. This ensures that it encompasses all objects within the scene. The **Sky** object itself does not represent a real sky, which is why a fitting material or photo must be assigned to it. Fully spherical HDR images or panorama images are well suited for this purpose. The Cinema 4D *Content Browser* contains several such images that you can use. They can be found in the *Content Browser's* **Presets/Presets/Lighting/HDRI** directories. Of course you can also use your own panorama images that don't have a 32-bit color depth.

The advantage of using the **Sky** object is that it encompasses the object completely. This makes it possible to simulate reflections and reflective objects that look very natural. The **Sky** object can also used to illuminate the scene in conjunction with HDR images if **Global Illumination** is enabled in the **Render Settings** menu. The brightness values of the image assigned to the **Sky** object can be interpreted as light emitted from the sky. We will discuss this in more detail in the materials section.

8.3 The Environment Object

This object will initially not be visible. This object creates a type of atmosphere that fills the entire 3D space. This atmosphere can be filled with fog if the **Enable Fog** option is enabled. The fog will spread spherically in all directions starting from the camera's location, which means it will cover any existing **Sky** or **Background** objects. To prevent this from happening you can disable the **Affect Background** option. The fog's color can be set to any color – even black. The **Strength** value is a brightness multiplier for the fog's color. The **Distance** value defines the fog's density and uses the distance from the camera as a reference. For example, if Distance is set to 10,000 cm, all surfaces that lie at least 10,000 cm from the camera will be completely covered by fog. The closer these objects come to the camera, the less they will be covered by fog and the better their shapes can be recognized.

A second function of the **Environment** object affects the creation of additional lights. The Environment light has the same effect as a light that is configured for **Ambient Illumination**. This light's **Strength** and **Color** can be defined using the Environment object. This type of light should be used subtly, if at all, because it brightens all surfaces with an overall brightness and does not restrict its influence using falloff like normal lights do. Too much environmental/ambient light will make objects look flat and two-dimensional.

8.4 Foreground and Background Objects

The **Background** object can be used to display images or colors in a scene's background. The **Background** object is "unattainable" for the observer because it will always remain the farthest element from the camera no matter how much you zoom in. Hence, this object cannot be used to cover up 3D objects that lie "in front" of it. This object is like a flat movie screen that does not enclose the entire scene. This is also why this object will not reflect onto surfaces in the scene. If a **Sky** and **Background** object are both in the scene, the **Sky** object will "have the upper hand" and the **Background** object will not be visible in the scene. It will be obscured by the **Sky** object. These types of conflicts between objects can be resolved using the **Compositing** tag, which can, for example, be used to define visibility per object. This will be discussed in detail in the rendering section.

The **Foreground** object has a similar function and instead always lies in the foreground, as the front-most object in front of the camera. This object can, for example, be used in conjunction with masked images to make sure that logos or copyright information is always rendered at the same location. If a material with an Alpha mask is used, the **Foreground** object will cover the entire scene like a camera lens cap.

8.5 The Stage Object

This object is only relevant for animations. It offers various link fields for Sky, Foreground, Background, Environment and Camera objects. Links can be switched during the animation to create hard camera cuts to simulate cutting from one camera to another.

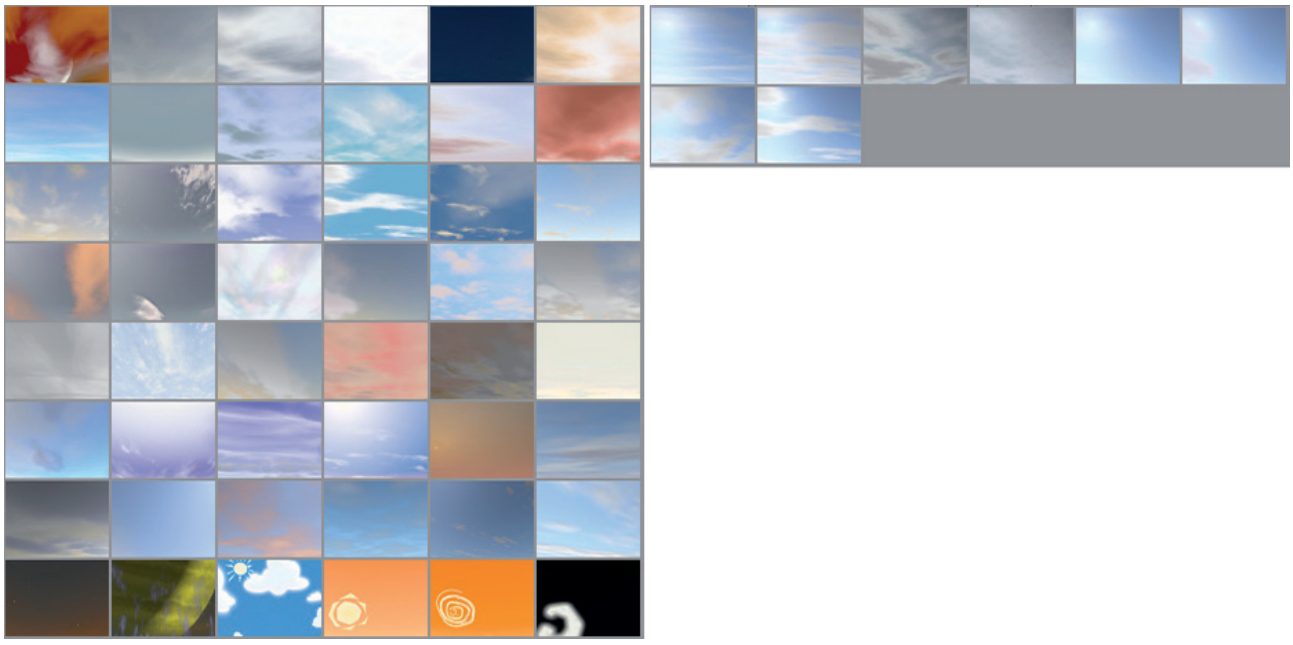
8.6 The Physical Sky Object

This object goes far beyond the features of the normal **Sky** object. The **Physical Sky** object not only simulates skies based on the time of day or season but also generates the lighting for the time of day, etc. This, in conjunction with **Global Illumination** for an even more natural dispersion of light, is basically the only type of lighting needed for outdoor scenes such as the rendering of houses.

The **Physical Sky** offers numerous atmospheric effects that can be enabled individually. These settings can be found in the **Sky** object's **Basic** tab in the ***Attribute Manager***.

8.6.1 The Basic Tab's Settings

The Load Sky Preset button can be used to load pre-defined skies, which can also be subsequently modified and fine-tuned. The Load Weather Preset button can be used to primarily load pre-defined cloud scenarios, which can also modified and fine-tuned.



The following options are used to enable various Physical Sky effects and can be used in any combination:

8.6.1.1 The Sky Option

The Sky option refers to the color in which the sky is rendered. The sky can be given a custom color or you can let Cinema 4D automatically define the sky's color according to the position of the sun. A light will also be generated that renders dispersed sunlight to complete the makeup of the sunlight.

8.6.1.2 The Sun Option

Enabling this option will position a sun according to the season and time of day defined. A light source will also be created that produces shadows. This and the **Sky** option are the two most important lighting factors for outdoor scenes.

8.6.1.3 The Atmosphere Option

This option can be used to simulate the dispersion of light in the atmosphere. Objects in the distance will be given a blue hue to simulate the effect of atmospheric perspective.

8.6.1.4 The Clouds Option

Enable this option to create two-dimensional layers over the sky that automatically have cloud patterns. **Density**, **Color**, **Scale** and **Shape** can be defined individually.

8.6.1.5 The Volumetric Clouds Option

If you want more than just simple clouds, volumetric, i.e., three-dimensional clouds, can be created by enabling this option. These must be painted manually in the sky and offer the advantage that you can physically move around or through these clouds, e.g., when animating a plane or balloons, etc.

8.6.1.6 The Fog Option

Similar to the Environment object, the atmosphere can also be filled with fog. However, this option offers numerous additional options, e.g., for defining the fog's height off the ground.

8.6.1.7 The Rainbow Option

A rainbow always lies opposite the sun and can be added to your scene by enabling this option.

8.6.1.8 The Sunbeams Option

Enable this option to create visible sunbeams when the sun is partially obstructed by clouds. Beams will be created that resemble the volumetric light effect.

8.6.1.9 The Sky Objects Option

Enable this option to add images to your sky. You can, for example, have a death star orbit the Earth if you like.

8.6.2 Time and Location Settings

These are among the most important settings for creating a **Physical Sky** and accurately calculating sunlight.

Of course the position of the sun depends on the time of day but also on the time of year and your own position on the Earth.

The **Time** menu's settings let you define the year, month and day for your scene. The time of day can be defined manually or by clicking and dragging on the clock face. Clicking on the **Today** or **Now** buttons will automatically define today's date and the current time, respectively (as it's set for the computer on which you are working!). If you click on the small arrow to the left of the **Time** setting, additional options will be made available. If **Current Time** and **Today** are enabled, the current time and date will always be used. This also means that the current time and date will automatically be used the next time the Project is opened, which means that the sky can look very different from when you originally saved the file.

The other two options are only relevant when rendering an animation in time lapse. Time and date can also be animated using keyframes. Enabling these two options means that keyframes will be ignored with regard to these parameters.

Keyframes can be seen as data packs of sorts to which simple parameters such as an object's position – or the time and date for a **Physical Sky** – can be saved. A **keyframe** also saves information regarding the location within the animation's timeline at which this information needs to be called up. If at least two **keyframes** exist with differing information at different points along the timeline, the values will automatically be interpolated. For example, if the first **keyframe** has a time of 12:00 noon saved and the second **keyframe** has a time of 13:00 saved, the sun in the animation will move a total of 1 hour between these two frames. This method can be used to create day-to-night animations, for example.

Another factor for determining the sun's position is the location of the observer on the Earth's surface. The easiest way to define this is to use the **City** options, which contains a long list of major cities from which you can choose. As a rule, all you have to do is select a city near to the actual location at which you want to be.

You can also add locations if you know their longitude and latitude degrees.

The **Time Zone** setting can use either that of your computer's operating system or you can define it manually. The **Custom** option is recommended if a file is passed to a colleague or client in another time zone to prevent a change in the sun's position when the file is opened at the new location.

If daylight savings time is used in the selected region, its length can be selected from the **Daylight Saving** setting. The **Time Zone Diff.** value defines the difference of the local time to the GMT (Greenwich Mean Time). The **DS Time Zone** is the daylight saving time's difference to GMT. If a city is selected form the list, these settings will all be defined automatically.

8.6.3 The Sky Settings

These settings affect the sky's color and brightness as well as the superimposed rendering of the starry sky. You can basically select one of two methods of calculation. If **Physical Sky** is enabled, realistic renderings will be made, based on such factors as humidity and the sun's current position. You can also define all of these factors manually, which is a good way of creating abstract environments on alien planets, for example.

Enable the **Horizon Line** option if you only want to render the top of the hemisphere.

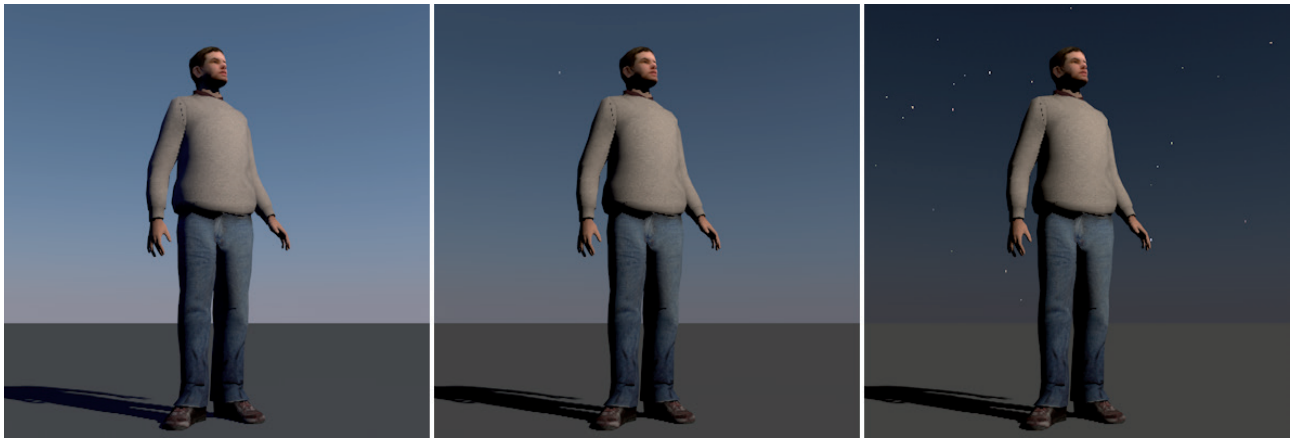
If the **Physical Sky** option is enabled, the **Physical Sky** object will work with a much larger color spectrum than the standard RGB color spectrum, which means that you can fine-tune the rendering of the color ranges, if desired. Higher **Color Warmth** values, for example, will render color ranges with a yellow or even red touch.

How the horizon is rendered also plays a large role for the rendering of the sky. The horizon's position is defined by the planet's radius. This radius can be defined manually using the **Earth Radius (km)** value in case you want to create a scene on a planet other than the Earth. The default value of 6370 km represents the Earth's radius. The horizon can also be lowered by entering negative **Horizon Start** values. This is often useful, for example, when using a **Landscape** object that does not reach all the way to the horizon. The horizon can be lowered slightly to make it disappear behind the **Landscape** object's hills.

The sky's color range can also be defined manually if **Physical Sky** is disabled and **Custom Horizon** is enabled. Use the Color settings to define a custom color range.

The color at the left of the gradient is the horizon color and the color at the right is that of the zenith. Use the **Max. Altitude** value to define the angle at which the first half of the gradient should be used. The default value of 20° means that the color at the left end of the gradient will be applied to the first 20° of the sky – starting at the horizon. This will only be applied if the sun lies lower than the defined **Max. Altitude** value, in which case the sun's angle will be used as the **Max. Altitude**.

The **Atmosphere Strength** value defines the thickness of the density of the visible atmosphere. The lower the value, the more transparent the atmosphere will be.



Larger values will increase the sky's brightness and intensity. In extreme cases, this can result in a glisteningly white sky.

The **Effects Turbidity** value defines the atmosphere's haziness. The sky's overall brightness will increase somewhat and all colors will appear less saturated.

If **Physical Sky** is enabled, numerous additional options will be made available. **Intensity** is a multiplier for the sky's brightness. This applies to the light emitted from the sky onto the scene and not the brightness of the visible color gradient. The sun is the primary light source and its light is diffused by the atmosphere, which means it is also affected by the light emitted by the sky. The **As Seen Intensity** value can be used if you want to increase the brightness of the visible sky.

Because even the night sky can emit light, there is a special setting with which this effect can be fine-tuned. The **Night Intensity Ratio** serves as a multiplier for the **Intensity** value. This makes it possible for you to have separate settings for day and night. This is, of course, only relevant if you're creating nighttime images or if you strongly reduce the **Atmosphere Strength** setting.

As we already mentioned, the sky also affects the scene's lighting. Since this effect often has a blue hue, the scene will be affected accordingly. The **Saturation Correction** value can be used to fine-tune this hue individually. Low values will de-saturate the sky's light and a value of 0% will produce gray tones only. If you do not want to create a natural color or if you want to correct the sky's color in a specific way you can modify the **Hue Correction** value. You can use any color in the color spectrum and give the sky a completely new color. A value of 0% will produce the physically correct hue for the sky.

As you know, the depiction of the sky is done using a much larger color spectrum than what can be displayed by your monitor. The **Gamma Correction** value affects the contrast between the white point and the black point of an image. High values will brighten the sky and low values will darken the sky.

Dithering adds a slight noise to the brightness and color gradients. This helps avoid jumps and banding when color spaces are converted. A value of 0% will disable this function.

The **Turbidity** value defines the amount of humidity in the air. The higher the humidity, the more light that will be diffused. This will result in the sun having a correspondingly stronger effect on the sky's coloring.

The **Ozone (cm)** value blocks UV rays from the sun. The sky's look will not be affected but the sunlight will have an increased blue hue with higher **Ozone (cm)** values.

8.6.4 The Sun Settings

If the **Sun** option is enabled in the Basic settings, this menu can be used to define the lighting and coloring of the scene by the sun. Since the sun's position is automatically defined by the date, time and location selected, fine-tuning is limited to the sunlight's visible attributes. The sun's path and position is also displayed using a wind rose in the Viewport. Make sure the **HUD** option is enabled in the **Details** menu. Note that the lighting of the night scene by the moon also indirectly depends on the sun parameters. The sun should be left enabled for night scenes as well.

A main criteria for the sun is its color – or the color of the light it emits. If **Custom Color** is enabled, you can define your own color. Otherwise the sun's color will be defined by the physically correct simulation. The color of the sun as defined by the simulation is shown in the **Preview Color** setting. **Custom Color** must be disabled for this preview to be shown.

The **Intensity** value defines the brightness of the sunlight. The **As Seen Intensity** value works in conjunction with the **Intensity** value and defines the brightness with which the sun is displayed in the sky, i.e., how bright the sun is in the rendered image. The **As Seen Intensity** value is multiplied with the **Intensity** value to define the sun's rendered brightness.

The next settings are the same as those in the **Sky** menu but they only affect the illumination by the sun. The **Saturation Correction** value defines the coloring of the sunlight. If set to 0%, the sunlight will be completely de-saturated and left with no color. The **Hue Correction** value can be used to define the color of the sunlight. You can define a custom color without having to enable the **Custom Color** option to do so. The **Gamma Correction** value controls the conversion of the sun's brightness. Large values will artificially darken the shadows in the scene that are cast by the sunlight. This can be used to reduce over-exposure, for example. The **Size Ratio** value defines the size of the visible sun in the sky. Because this also changes the size of the sun light source, shadow casting might be affected as well. We will discuss this in detail later.

You already know that the ring-shaped reflections created by a camera lens are called lens flares. They are created when light is reflected and refracted in a lens. To create this effect, enable the **Lensflares** option. The halo around the sun is referred to as a lens flare. This effect is controlled by the **Glow Intensity** value, which can be accessed by clicking on the small arrow next to the **Lensflares** option. You will also see the **Flare Intensity** value, which is used to define the visibility of the glow effect.

The distance of the sun light source from the wind rose can be adjusted using the **Distance Scale** option. However, this is only necessary if you use a custom polygon object in the **Custom Sun Object** field. This will not affect the lighting by the **Physical Sky**. A light source can even be placed into this field, whose coloring, brightness and position will then be automatically assumed by the **Physical Sky**. This bears the advantage that other light sources can also be used and the additional light object's settings can be used as well. Either way, to add an object to the **Custom Sun Object** field, simply drag it from the ***Object Manager*** into this field.

Below this setting is the **Shadow** menu. You can select between **None**, **Hard** and **Area** in the **Shadow** setting. If **None** is selected, no shadows will be cast. If **Hard** is selected, shadows with hard edges will be cast. This shadow type is a good compromise between achieving a realistic-looking result and an acceptable render time. If **Area** is selected, additional samples will be emitted into the scene from the sun's surface. An interpolation of many of these samples can result in extremely natural looking shadows that look hard at the object base but get softer farther away from the object. The softness of the **Area** shadow can be adjusted by the size of the sun and using the **Size Ratio** value. A larger sun will automatically create softer shadows.

The number of samples is defined by the **Minimum Samples** and **Maximum Samples** values. Larger values will produce more precise results but the render times will increase accordingly.

However, if the values are too low, visible noise will be produced in the shadows.

We already discussed how these values can be optimized when we discussed **Area** shadows in the light sources section.

The **Accuracy** value determines whether Cinema 4D will use the maximum or the minimum number of samples. Reducing this value can help reduce render times. The best ratio between the **Minimum Samples** and **Maximum Samples** values has to be found in order to avoid poor render results. You can also define the shadow's **Color**. Normally, black will be the right color but the color can be modified manually if the shadows need to be brightened or if they should have a different color. Remember that even black shadows can be brightened or given a slight hue by the sky or other light sources within the scene.

The Transparency option should remain enabled as a rule. This ensures that shadows will be colored and their density reduced automatically when sunlight passes through transparent objects.

8.6.5 The Atmosphere Settings

This effect simulates the haziness in the atmosphere that diffuses the sunlight. This causes the color of objects that lie near the horizon to change – they will have a blue hue. The intensity of this effect depends on the objects' distance from the observer. Cinema 4D uses real-world units of measure for this effect. 1000 m in Cinema 4D are 1000 m in the real world if the project scale ratio is set to 100%. If set to 50%, 1000 m in Cinema 4D would be 2000 m in the real world.

The **Intensity** value defines the amount of light the atmosphere will diffuse. Normally, the sky's color will be used. However, if you increase the amount of **Horizon Fade**, the horizon's color will be used. This will produce an even softer transition between ground and sky. The saturation and coloring of the haze can also be affected by the **Saturation Correction** and **Hue Correction** values. You should already be familiar with these settings from the previous section.

The **Gamma Correction** value can be used to adjust the atmosphere's brightness. Larger values will brighten the atmosphere. The **Dithering** value can be used to add a slight noise to the lighting effect. This can help avoid visible brightness or color banding.

8.6.6 The Clouds Settings

A sky wouldn't be complete without clouds. The **Physical Sky** object lets you add simple cloud formations by superimposing noise patterns. This is similar to the **Noise** effect available for the **Light** objects.

At the top of the **Cloud** menu you will find the **Rolloff** gradient. This horizontal gradient is responsible for automatically fading the clouds out at a defined distance. The clouds would otherwise not look realistic if they meet the horizon in the distance.

The color handles can use the **Intensity** value, which is made available when a handle is clicked on. Click on the small arrow next to the **Rolloff** setting to reveal the **Intensity** and other options. Custom colors cannot be defined. An **Intensity** value of 0% will make the clouds completely visible and a value of 100% will fade them out entirely. The right end of the gradient represents the horizon and the left end represents the clouds' zenith. If you only want clouds to appear near the horizon you can, for example, set the left **Intensity** value to 100%.

The **Cast Shadows** option will cause clouds to actually block and filter sunlight. This can cause corresponding shadows to be cast onto the ground. This option must be enabled if you want to use the **Sunbeams** option in the **Basic** menu.

Just below the **Cast Shadows** option are options that let you combine up to six cloud layers. Each active cloud layer makes its own settings available in the corresponding menu below. The menus for each layer are identical so we will only use the first one for explanatory purposes.

The first setting is the **Noise** setting, from which you can choose the type of cloud formation you want. You can click on the small gray arrow next to the drop-down menu to display a small preview gallery of the different cloud types, from which you can also make your selection.

Color and **Height** can be defined using the respective settings. These can, for example, be used to arrange darker clouds at a lower altitude than brighter clouds. This amplifies the three-dimensional effect of the clouds, even though it's only a two-dimensional effect. If you want to save the cloud formation you created, simply click on the **Save as Preset** button to save it to the ***Content Browser***. This preset can later be loaded into any scene by simply dragging it from the ***Content Browser*** into your scene. The preview images can also be used to transfer one layer's settings to another by dragging one layer's preview image onto that of another layer.

The **Noise** option selected defines the cloud distribution and their shape. Their look can, however, be modified using the layer's settings. The **Density** value defines the clouds' visibility and contrast. The higher the value, the more massive the clouds will be. **Coverage** defines the clouds' size and subsequently the number of clouds in the sky. The **Thickness** value defines the clouds' visibility for illumination by the sun. For example, a low **Thickness** value will produce correspondingly brighter clouds.

Transparency does not affect how the clouds look. It affects the intensity of the shadows they create. This is only relevant if **Cast Shadows** is enabled. If you want to add more lights with active shadow casting, these can also use the clouds to cast shadows. These lights must, however, be placed higher than the 10,000 meters at which the clouds lie when rendered. The **Height** value, therefore, does not represent a real distance. However, as a rule you will want to avoid creating additional shadows because the sun already casts shadows using the clouds.

The following settings are used to scale the Noise pattern and help when positioning and animating the clouds. The **Scale N-S** and **Scale W-E** values can be used to scale the clouds in North/South or West/East directions, respectively. These directions are defined by the **Physical Sky's** wind rose.

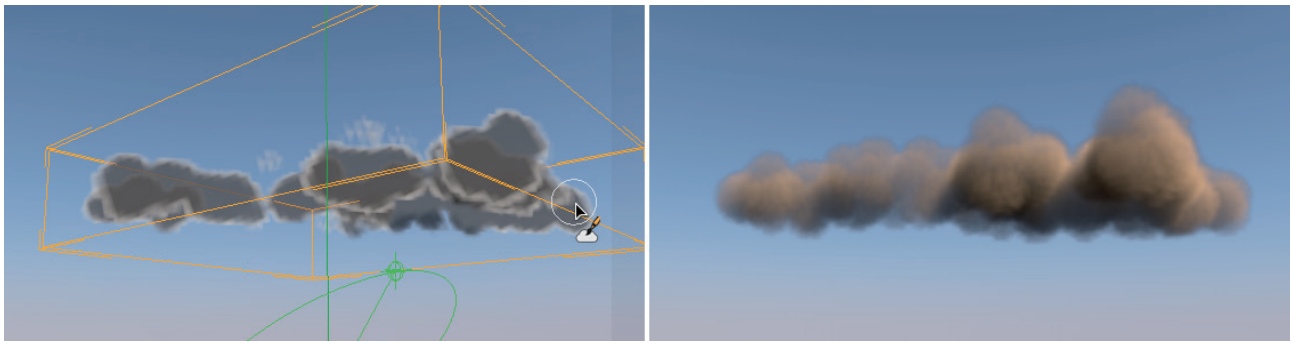
Pos. N-S and **Pos. W-E** work the same way but they are used to move the pattern in the respective directions. These values can be animated to simulate passing clouds. The shapes of the clouds can also be animated. This is done using the **Anim. Speed** value. The higher the value, the faster the cloud pattern will fluctuate. These random fluctuations can be varied by modifying the **Offset** value to make them look more interesting.

8.6.7 Volumetric Clouds

The **Physical Sky** object also supports volumetric structures once the **Volumetric Clouds** option has been enabled in the **Basic** menu. This type of cloud can, for example, be flown around or through. To create a volumetric cloud, use the **Create/Physical Sky/Cloud Tool** command.

8.6.7.1 The Cloud Tool

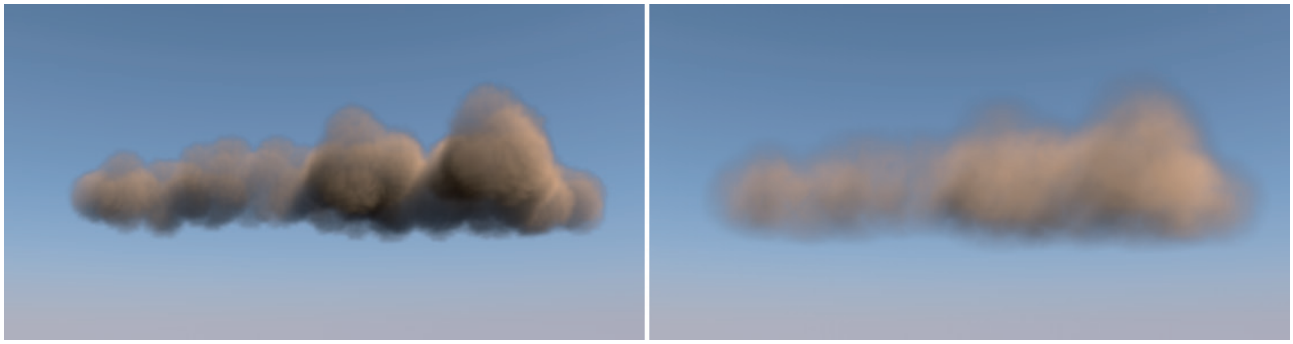
Select the **Physical Sky** object and use the **Cloud** tool to simply draw clouds in the sky in the Perspective view. After releasing the left mouse button, i.e., after painting the clouds, a new **Cloud** object will appear under the **Physical Sky** object in the ***Object Manager***. This object contains all information about the clouds you just drew.



Select this object and **Shift** + drag in the Viewport to position a transparent layer within the cloud's bounding box. This layer can run vertically or horizontally through the bounding box, depending on which edge of the bounding box you drag the cursor.

This transparent layer basically defines a painting plane. Click again with the LMB and release the **Shift** key to set the layer to its new position. If you select the **Cloud** tool again, the next cloud you create will be created on this plane only. The Cloud tool also offers numerous additional functions and settings. Because the tool resembles a brush, the **Radius** value can be used to define the size of its tip. The **Density** value defines the number of clouds that will be drawn. The **Maximum Size** values are used to define the maximum spread of the clouds, and **Threshold** defines the depth within the cloud in percent at which the clouds can be painted. If set to 100% you will be painting at the center of the clouds; at 50% you will be painting at their edge. This keeps you from always having to take the transparent reference plane into consideration. Additional buttons let you **Clear** (delete) the cloud, automatically fill a defined reference plane with clouds (**Fill Plane**), or create a spherical cloud (**Fill Sphere**) that will be created within the area of the bounding box. The **Fill Sphere** command can also be used to create lenticular clouds when the bounding box is very flat. The volume of the clouds will orient themselves to the **Density** value.

The **Cloud** tool has even more to offer. In the **Smoothing** menu you will find commands (**Smooth Borders**) with which the edges of drawn clouds can be made to fade out naturally. This function can only be executed using this button and cannot be done interactively when painting the clouds. If you want to smooth the edges of the clouds, first define the **Distance** from the edge over which the density should stretch. Use the **Variance** value to create random variations in the smoothing's **Distance**. The smoothing's effect and its intensity can be defined manually using the Shape curve. Without this curve, a linear smoothing would be used. Clicking on the **Smooth Borders** button will apply the smoothing to the clouds. This can be done multiple times until the desired look has been achieved.



If you want to smooth the entire cloud, use the **Smooth All** button. The **Strength** value defines the intensity of smoothing and the **Variation** value defines the randomness of smoothing.

The lower group of settings can be used to completely redesign the density of the cloud. As a rule, the center of the cloud has the highest density. However, this doesn't always look the best or the most realistic. This is where the density curve must be used to fine-tune the cloud. Simply **Cmd/Ctrl** + click on the Remap function graph to create a curve. The X axis represents the cloud's old density. The left edge represents a density of 0% and the right edge 100%.

The curve's height defines the cloud's new density. The bottom of the graph represents the new density at 0%, which means that the top of the graph represents the new density at 100%. Click on the **Remap** button to apply the new density to the cloud.

The **Display** menu is used to define the preview quality of the clouds. Use the **Quality** slider to define the number of cloud points that should be used for the preview in the Viewport. Click on **Render Preview** to render the clouds using the defined level of quality, which will also be used for final rendering, which can take longer. An alternative is to use Enhanced OpenGL in the Viewports' **Configure/View** menu. This also improves the display quality enough so the cloud formations can be checked for accuracy.

A somewhat hidden function is triggered when a polygon object, spline object or particle emitter is made a sub-object of a **Cloud** object. This will delete the existing cloud and replace it with a new cloud in the shape of the subordinate object. The object used to define the shape can then be deleted. The cloud will remain.

8.6.7.2 Grouping Clouds

Because clouds generally appear in groups, volumetric clouds are considered a **Cloud Group**. This command can be found in the **Create/Physical Sky** menu. A **Cloud Group** must be made a sub-object of the **Physical Sky** in the *Object Manager*. The individual **Cloud** objects will in turn be sub-objects of the **Cloud Group**.

The **Cloud Group** serves not only to group the clouds but also for animating clouds. Multiple clouds can be moved or rotated together. Clouds or **Cloud Groups** can even be scaled when in **Use Object** mode. The **Cloud Group** offers numerous settings for positioning and defining the look of clouds. The **Min. Altitude** and **Max. Altitude** values define the altitude range for the subordinate clouds. **Min. Falloff** and **Max. Falloff** work similarly to the distance values for the fog effect. The smaller the values, the more dense the clouds will appear. Larger values will make the clouds look correspondingly thinner and more transparent.

Min. Lum. Falloff and **Max. Lum. Falloff** define the absorption of sunlight in the subordinate clouds. The smaller the values, the more sunlight that will be absorbed, which will make the clouds correspondingly darker. Larger values result in correspondingly brighter clouds. The **Min. Trans. Falloff** and **Max. Trans. Falloff** values define the visibility limits for cloud shadows. The smaller the transparency, the opaque the clouds' shadows will be. The last settings are the **Color** settings, which can be used to define the color of the clouds. Normally, white and gray tones should be o.k., unless you want to simulate smoke or exhaust. If you're wondering why a minimum and maximum value is required for these options, we will explain why below.

If you want to link existing Cloud objects to a new **Cloud** object, select the **Cloud** object in the *Object Manager* and then select the **Connect Clouds** command from the **Create**/**Physical Sky** menu. Existing **Cloud** objects will remain but will be deactivated.

8.6.7.3 Cloud Settings

You already know that a **Cloud** object is created when a cloud is drawn using the **Cloud Tool**. You can also create a cloud by selecting **Create**/**Physical Sky** and then drawing or making a polygon, spline or emitter object a sub-object and filling this object.

The clouds can then be scaled using the handles. If the **Cloud** object already has a cloud structure, it will automatically be scaled as well. By making the cloud a sub-object of a Cloud Group, the cloud will assume the group's settings. Enable the **Override Group** option if the cloud should use its own settings, even when it's a sub-object of a **Cloud Group**.

This will make the Cloud object's **Altitude**, **Falloff**, **Luminance Falloff**, **Transparency Falloff** and **Color** settings available. You should already be familiar with these settings from the Cloud Group. If the **Override Group** option is disabled, the **Cloud Group** settings will be used. Because these settings always define minimum and maximum values, the **Mix** value can be used to define each **Cloud** object in a **Cloud Group** individually, e.g., with regard to their **Altitude** or **Falloff**. A **Mix** value of 100% will use the maximum values defined in the **Cloud Group**. A value of 0% will use the minimum values.

If **High-Quality Lighting** is enabled, clouds can be illuminated using **Light** objects, e.g., positioned at the center of the cloud. This can be used when creating storm clouds that are illuminated by lightning bursts. This mode will, however, increase render times. If this option is disabled, lighting will be optimized to sunlight as the single source of light. If the cloud's shape was created using a polygon, spline or emitter object, the **Falloff** function graph and the **Distance** value can be applied. The **Falloff** graph defines the density range at the cloud's edge. The **Distance** value defines the distance from the edge of the cloud to the falloff curve. When using a sub-object for defining the cloud's shape, the Keep Shape option should be enabled. This speeds up the cloud's display in the Viewport. This option should only be disabled if the element used to define the cloud's shape will be animated. The Cloud Type setting basically defines how the existing cloud points should be arranged, regardless of whether they are painted or created using a subordinate object. The following options are available:

- **Standard**: The cloud will be calculated using the standard parameters. The cloud shape will roughly reflect the painted outline.
- **Ac perlucidus**: Jagged clouds will be created through which the sky can be seen in several places. The **Coverage** value defines the number and size of gaps within the cloud. Larger values produce more dense and larger clouds. The **Contrast** value affects the clouds' edge. A low **Contrast** value will produce soft, fading cloud edges; a high contrast value will create correspondingly hard edges.
- **Ac lenticularis**: These clouds appear uniform and look the most natural when in a flattened formation. This type is very similar to the **Standard** type.
- **Cb capillatus incus**: These are typical stacked storm clouds. The Ratio value defines the difference between the width at the bottom and the width at the top of the cloud formation. The Groove Depth value defines the formation's irregularity and the Shape Strength value defines the length of the transition between the top and bottom parts of the formation.

The three **Scale** values can be used to scale the **Noise** pattern that is used to create the irregular outer shape of the cloud formation. You can even create stone-shaped clouds, e.g., by defining a much larger X scale than Y and Z scales. The three **Grid Points** values display the actual size of the cloud in the three spatial directions.

8.6.7.4 Volumetric Cloud Settings

In addition to the various **Cloud** objects, the **Physical Sky** also offers numerous settings for creating volumetric clouds.

The box that is automatically created around volumetric clouds is called a bounding box. This bounding box can be scaled along any of the three axes using the handles. To do so, the **Move** tool must be active and the corresponding **Cloud** object has to be selected. The bounding box has handles like a Cube object, for example. Scaling the bounding box will automatically scale the cloud within it accordingly. If you want to hide the bounding box from view in the Viewport, disable the **Show Bounding Box** option in the ***Attribute Manager***. Either way, the bounding box will not appear in any rendered image.

The **Editor Color** setting defines the color of the bounding box that is displayed when a **Cloud** object is not selected. The cloud itself will, as a rule, only be represented visually by an arrangement of points in the Viewport. The density of these points can be defined using the Editor Quality value. If the **Cloud Tool** is active, the quality value defined in its **Display** menu will be used.

If the **Adjust Altitude** option is enabled, the volumetric clouds will automatically be positioned in relation to the defined **Earth Radius (km)**. The altitude consists of the **Earth Radius** plus the cloud **Height**.

Enabling the Receive Shadows option will cause volumetric clouds to cast shadows on themselves as well as on other clouds. All 2D clouds and all other objects in the scene will also be able to cast shadows onto the volumetric clouds. The Sample Size value is another important setting for determining cloud quality. The lower this value is, the longer the rendering will be – and the more precise the cloud formation will be rendered.

The edges of the clouds are faded out to prevent them clouds from looking like massive bodies. This transition can be broken up and made more random using additional structures. The **Edge Noise** setting offers several types of noise patterns with which this can be done. Click on the small gray arrow at the right of the drop-down menu to open a preview window that contains a small preview of each pattern. Click on a pattern to select it. The larger the **Contrast** value, the more clearly the selected pattern will appear at the cloud's edge. The **Scale** value defines the pattern's size. Low values will create correspondingly smaller and finer structures.

The **Noise Speed** setting is for animation. It varies the noise pattern over time. The larger the value, the faster the variations will take place.

The **Lights** list works similarly to the **Light** object's **Object** list. If the list is empty, the volumetric clouds will only be illuminated by the sun and the moon if no other measures have been taken with regard to lighting.

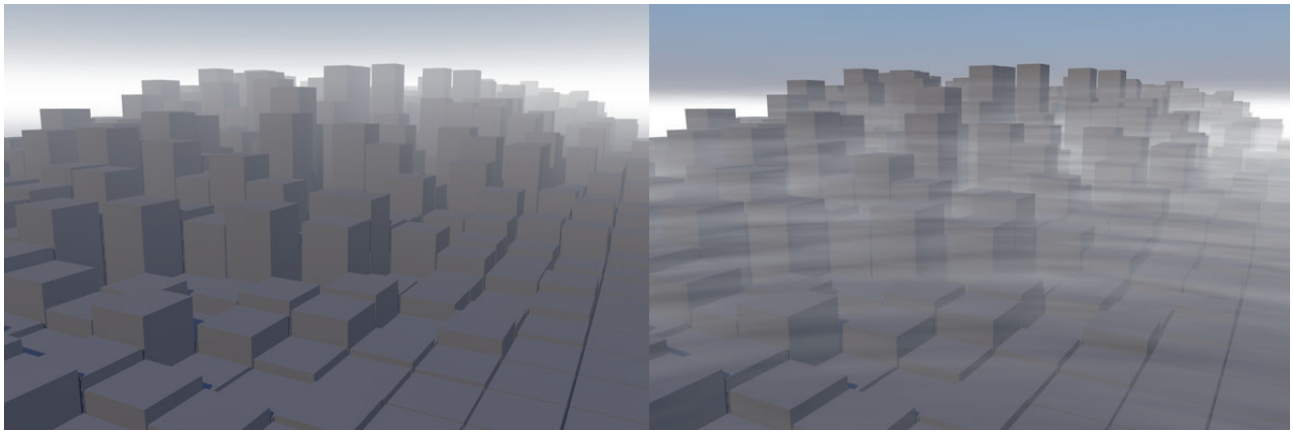
If additional light sources should be used to illuminate the clouds, drag them from the *Object Manager* into the **Lights** list. Alternatively you can use the special selection mode, which is activated by clicking on the cursor icon at the right of the list. How this mode works was explained in the lights section. Right-clicking on an item in the lest will open a context menu with which you can, for example, remove this or all items from the list. You can also enter or exit the special selection mode using this menu.

You can also select objects in this mode, which makes them easier to locate in the *Object Manager*.

8.6.8 The Fog Settings

The **Start Height** and **End Height** settings let you precisely define the fog's height above the ground. The **Max. Distance** value affects the spatial expansion of the fog. The distance is calculated starting from the camera's current position. The fog will fill only the space between the camera and this defined distance. The **Density** value defines the fog's actual density. Larger values will make the fog correspondingly more dense, smaller values will make it increasingly more transparent. The density's spread can also be defined individually using the **Density Distribution** function graph. The left end of the graph represents the observer's location and the right end represents the maximum distance. The height of the curve defines a multiplier for the **Density** value. If the curve reaches the top edge of the graph, the defined **Density** value will also be reached at this distance. The fog can also be assigned a custom **Color**.

In the following we will run across known **Noise** types that can, in conjunction with **Noise Strength**, be used to add variation to fog. **Noise Strength** works similarly to a contrast slider that increases the visibility of the noise pattern in accordance with its own increased intensity (**"Sky_Frog"** Project).

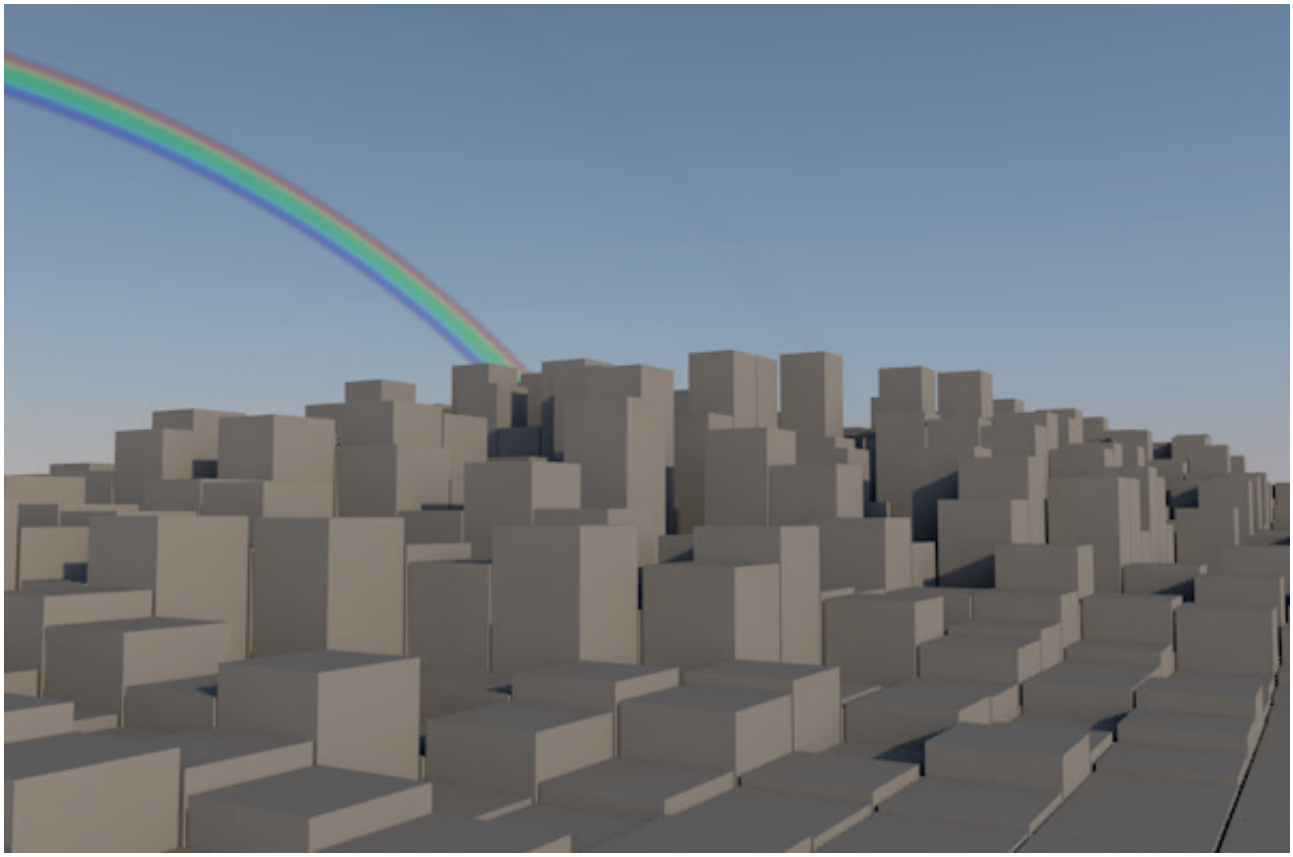


The **Noise** pattern can be scaled along all three axes using the **Scale** values. The **Noise** pattern can also be moved using the **Movement** values. Hence, these values can be animated to simulate passing fog. The **Noise** pattern will, however, not change. This can only be done using the **Animation Speed** value, which varies the **Noise** pattern. **Movement** and **Animation Speed** can be used independently of one another but both are relevant for animation.

The **Sample Size** value is a unit of measure for the density of the render samples used to render the fog. Smaller values will produce more precise results, which will also take longer to render. The principle is the same as for rendering volumetric clouds. Increasing the **Shadow Intensity** value makes it possible for objects in the fog to cast shadows. This will also increase render times accordingly. The **Illumination Intensity** value colors the fog using the sunlight emitted onto it.

8.6.9 The Rainbow Settings

Whenever light is passes through a transparent medium it is refracted. The light's wavelengths are diverted to differing degrees. The light's color spectrum becomes visible. When this effect occurs in the atmosphere it manifests itself in the form of a rainbow. If all conditions are met for the creation of a rainbow, the rainbow will appear opposite of the sun's position. The lower the sun lies, the higher the rainbow will appear. To be exact, a rainbow is made up of two different bows. This second bow can also be simulated using the **Physical Sky**.



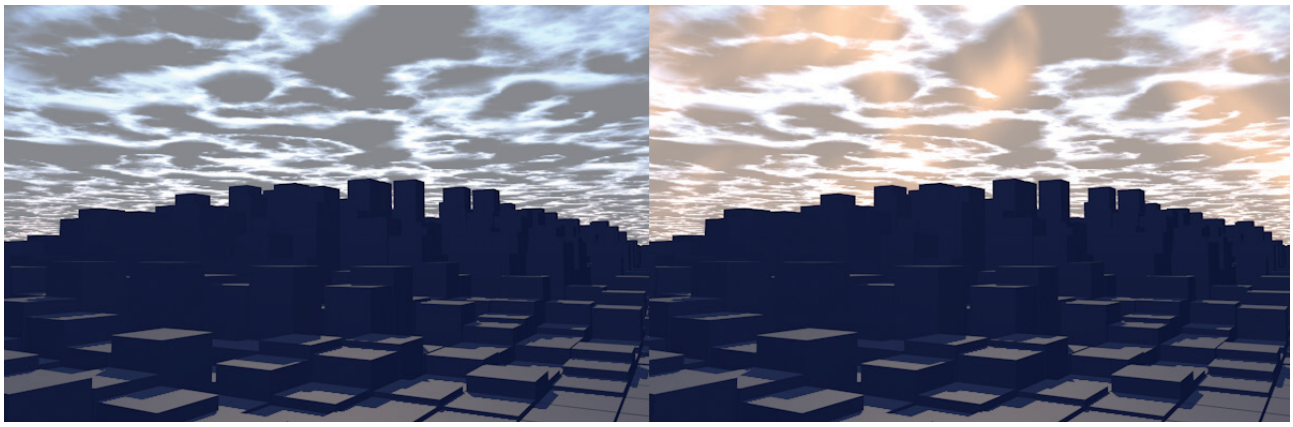
The **Max. Strength** value is a multiplier for the visibility of the rainbow effect. Larger values will reduce the rainbow's transparency accordingly and will eventually result in the rainbow having a comic look. As you know, rainbows are created by sunlight refracting in drops of water in the atmosphere. We have already discussed humidity in the **Sky** menu's **Turbidity** description. If this value should also be used for the rainbow, enable the **Turbidity Dependent** option. The **Min. Turbidity Threshold** and **Max. Turbidity Threshold** values can be used to define a percent range for the **Turbidity** within which the rainbow will be visible. This lets you influence the rainbow's visibility using the **Turbidity** effect for animations.

The following angle values are used to define the width of both rainbows. The **Inner Angle** and **Outer Angle** settings can be defined separately for each rainbow. Using these settings to correctly define a specific position and width requires practice and several test renderings.

The Start Clipping and End Clipping values are used to define the rainbow's distance from the observer. Each distance is measured from the camera outward. Objects that lie nearer than the Start Clipping distance will lie in front of the rainbow; objects that lie beyond the End Clipping distance will lie behind the rainbow. Objects that lie between these two values will be partially overlapped by the rainbow.

8.6.10 The Sunbeams Settings

If the sun lies partially behind dense clouds, visible light beams can poke through gaps in the clouds or at their edges.



The **Sunbeams** menu lets us simulate this effect in Cinema 4D. To simulate this effect, the sun must at least be partially obstructed by clouds. Here you can also use the **Turbidity Dependent** option for defining the intensity of the visible beams by making them dependent on the humidity. The **Turbidity** value defined in the **Physical Sky's Sky** menu will be applied. Otherwise, the **Intensity** value in the **Sunbeam** menu will be used. Adjust **the Min. Brightness** value to save render time and only render the more intense beams. Only beams that are brighter than the value defined here will be rendered. Since the beams of light spread through 3D space, you can use the **Start Distance** and **End Distance** values to define a depth at which the beams should be rendered. No beams will be rendered at a depth between the observer (or the camera) and the **Start Distance** value. The same goes for any depth beyond the **End Distance** value.

This menu also contains a Sample Distance setting. Smaller values will result in more calculation steps, better beam render quality and longer render times.

8.6.11 Sky Objects Settings

The **Physical Sky** includes numerous effects and object types. For example, sun, moon and many stars visible from earth will automatically be made visible for rendering. Simply define a time of night to see the stars.

The **Sky Objects** menu lets you load bitmaps that you can use to render celestial bodies. Click on the **Place Object** button at the bottom of the menu to open a dialog window in which you can select your bitmap. If these bitmaps contain alpha channels, these will automatically be used and will mask corresponding areas of the image. If you load the wrong image, simply right-click on it in the list and delete it.

The loaded image's size and position can be adjusted interactively in the perspective Viewport. Click on the image and drag the mouse to the side. A circle will be created that shows the image's size. After the mouse button in released, the image will be positioned at that location and scaled accordingly. When doing so, note that a disc with the image will be displayed even if the image does not contain an alpha channel. The image's corners will be cropped.

The position and scale of the image can, however, still be modified. Select the **Physical Sky** and switch to its **Sky Objects** menu. Next to each loaded image's name you will see various numerical values listed, which can be edited if when clicked upon. The **Azimuth** value defines the angle between the sky's south-pointing axis and the image. The measurement is made in a clockwise direction. At 90° the image will point westward, at 180° to the North and so on.

The **Altitude** value displays the angle between the horizon and the image. A value of 0° means that the image lies exactly on the horizon. Note that the image cannot be positioned at the zenith by entering a value of 90° without risking a faulty display of the image. Use a value of just under 90° instead, e.g., 89°.

The **Angle** value is a measure of the size of the image. The loaded image can also be illuminated by the sun. The image will be rendered as if it were a spherical object. If you do not want to disable this effect, click on the green check mark in the **Illum.** column to turn it into a red **X**. Finally, you can also define the image's visibility using the **Intens.** value. A value of 100% will make the image completely transparent and thus invisible.

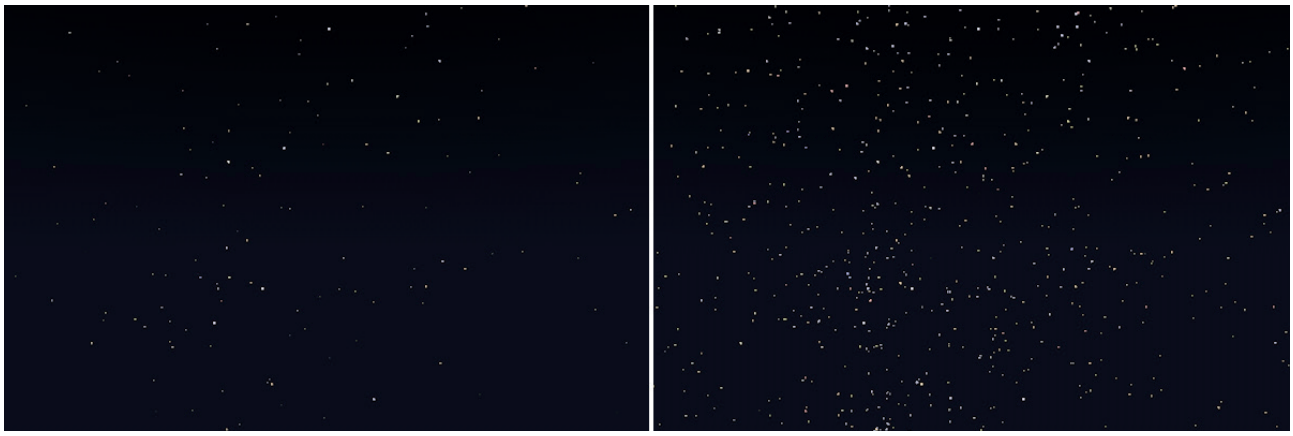
8.6.12 The Details Settings

Apart from the other options and settings, the **Details** menu contains general settings for the **Physical Sky**. The **Show Moon** and **Show Stars** options can be enabled or disabled individually. The **Show Planets** option will, of course, show the planetary bodies, if enabled.

Clicking on the small triangle next to the **Show Moon** and **Show Stars** options will make additional settings available for each option.

The **Scale** value defines the size of the moon. The **Bright Intensity** and **Dark Intensity** values are used to define the moon's brightness in accordance with the sun's illumination. You can also use a custom object as a moon by dragging it from the *Object Manager* into the **Custom Moon Object** field. You can also make use of the special selection mode by clicking on the cursor icon to the right of this field and selecting the object you want to define as the moon.

The distance of the custom moon from the Earth can be defined using the **Distance Scale** option. Numerous settings are also available to define the look of the stars. The **Min. Magnitude** value affects the stars like a filter. If low settings are used, only the brightest stars will remain visible; larger values will make correspondingly more stars visible.



In addition to the over all brightness, the size of individual stars can also be varied. If the **Resize Stars with Magnitude** option is enabled, brighter stars will be rendered somewhat larger. If this option is disabled, all stars will have the same size. The stars' brightness can also be affected using the **Brighten Stars** value. **The Star Radius** value is used to scale the stars. If the **Show Constellations** option is enabled, the star constellations will be shown, connected with lines. These connecting lines can be assigned a custom color using the **Color** setting. If the constellations' longitude and latitude should also be shown, increase the **Grid Width** value to more than 0°. These lines can also be assigned a custom color using the **Grid Color** setting.

You already know that light is not only emitted by the sun but also by the sky itself. This effect can be enabled or disabled using the **Sky Dome Light** option. This can be an advantage when Global Illumination is used to render a scene. Often, the blue hue from the sky can be bothersome. And not only lights but luminous materials also contribute to the illumination. The **Generate GI** option can be used to enable or disable this effect when using Global Illumination. The sky color will possibly also be prevented from affecting the scene's illumination.

Because the **Physical Sky**'s sun is principally a normal Cinema 4D light source it will also generate specular highlights. To be exact, this is not something sunlight should and is also not necessary since the sun appears as a bright object and can be reflected in surfaces. The **Merge Sky and Sun** option is a special mode that lets the **Physical Sky** function like an HDR image for rendering. This only makes sense if you render using Global Illumination, which lets HDR images be used as light sources. Also, all materials must have mirroring properties so the sun's shine can be displayed correctly.

If the scene should be illuminated using the sky's colors in Global Illumination, use the **Strength** and **Saturation** values to define the degree of their effect on the scene. If you are also using clouds to affect illumination, their influence with regard to GI can be defined using the **Cloud Influence** value.

As you know, the Viewport already supplies a good preview of the sky, which means that you don't necessarily have to do a test rendering. The Viewport's display quality can, however, be improved further by modifying the **Texture Preview Size** setting. Selecting a higher resolution will increase the amount of memory required but will also increase display quality dramatically.

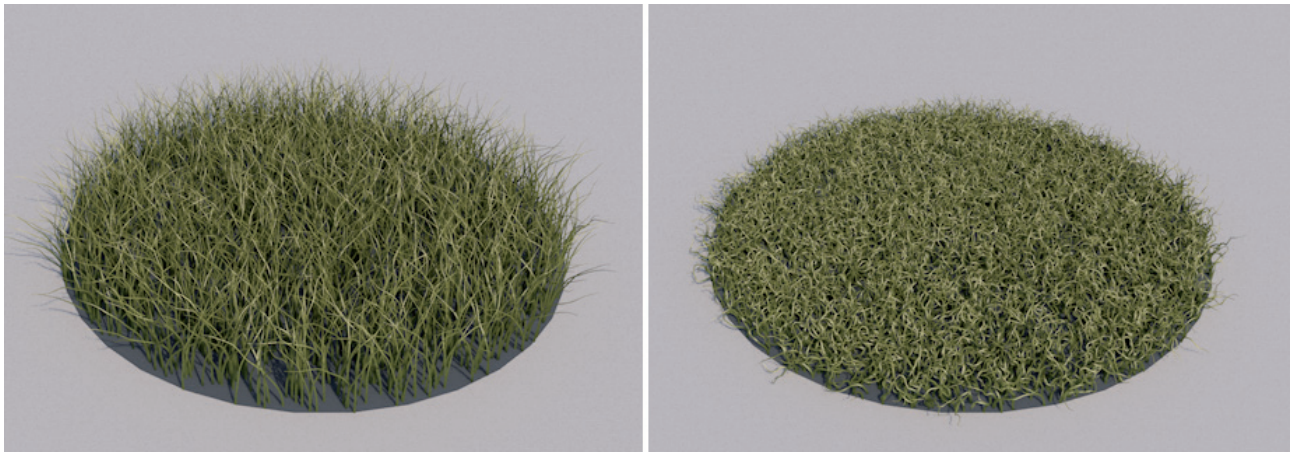
Since all light parameter calculations and the various display options for the **Physical Sky a**re always take place at a specific time, problems can occur, for example if you use a custom expression that affects the **Physical Sky's** settings. This specific time is called **Priority** and can be defined using the corresponding setting. Various options are available, each of which is rendered at a different time. **Initial** is always the first group of objects that will be rendered. **Generators** will be rendered last. They will be the last object group to be updated. The time at which a group is rendered can also be changed. For example, an object with a **Priority Expression** of -499 will be executed prior to an object with the same **Priority** type but with a **Priority** value of +499. Larger values result in correspondingly later execution. This can be important when working with **XPresso** or **Thinking Particles** expressions, for example, if an expression should react to specific values of an animated object.

The expression can only be executed after an object's animation has been rendered to ensure that the current values are always available. Enabling the **Show Location HUD** will activate a wind rose in the Viewport. North runs along the world Z axis by default. However, this can be modified manually using the **Rotate** tool, e.g., if you have an object that were oriented along a different axis. The position of the sun will adjust to the manual rotation of the wind rose. The wind rose's height defines the height of the horizon. This position can also be modified manually using the **Move** tool. But such a modification should rarely, if every, be necessary. With regard to the sky environment, most scenes will basically have the same scale.

You also know that the **Physical Sky** can display numerous effects directly in the Viewport. This includes the sky's color gradients, illumination by the sun as well as displaying clouds, if enabled. If the **Update Editor** option is enabled, the Viewport display will be updated as soon as a **Physical Sky** setting has been modified.

8.7 Grass Simulation

The final **Environment** object is the **Grass** object, which lets you grow grass on any polygon object. This effect can however, only be found in the Environment object's icon menu or in the main menu under **Create / Generators**. Of course, creating grass is perfect when creating outdoor scenes but can also be used for indoor elements such as carpets, etc.



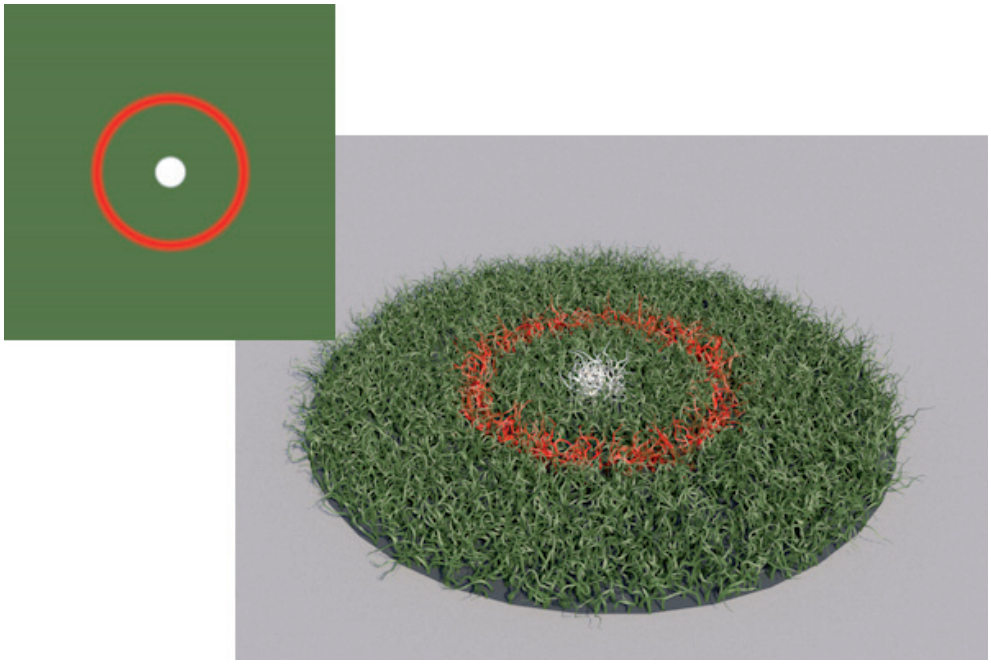
To make grass grow on an object, select it and call up the **Create / Generators / Grow Grass** command. The grass will not be displayed in the Viewport. The scene must be rendered to see the grass. Use the **Interactive Render Region** function to periodically check your grass.

This effect does not use actual blades of grass. Its look and growth are controlled primarily using a **Grass** material, which is located below the Viewport in the ***Material Manager*** window. A single click on the icon will make the material's settings available in the ***Attribute Manager*** at the right; double-click on the icon to open the material in a separate ***Material Editor*** window, which can be scaled or moved to a second monitor. We will discuss the ***Material Manager*** and Material Editor in more detail when we explain the material system. Here, we will concentrate on the **Grow Grass** effect.

The **Color** gradient is used to define the color along the blades of grass. The left end of the gradient represents the bottom of the blade and the right end the top. The colors can be modified by double-clicking on a color handle or by clicking on the small triangle to the left of the gradient bar to make the corresponding settings available.

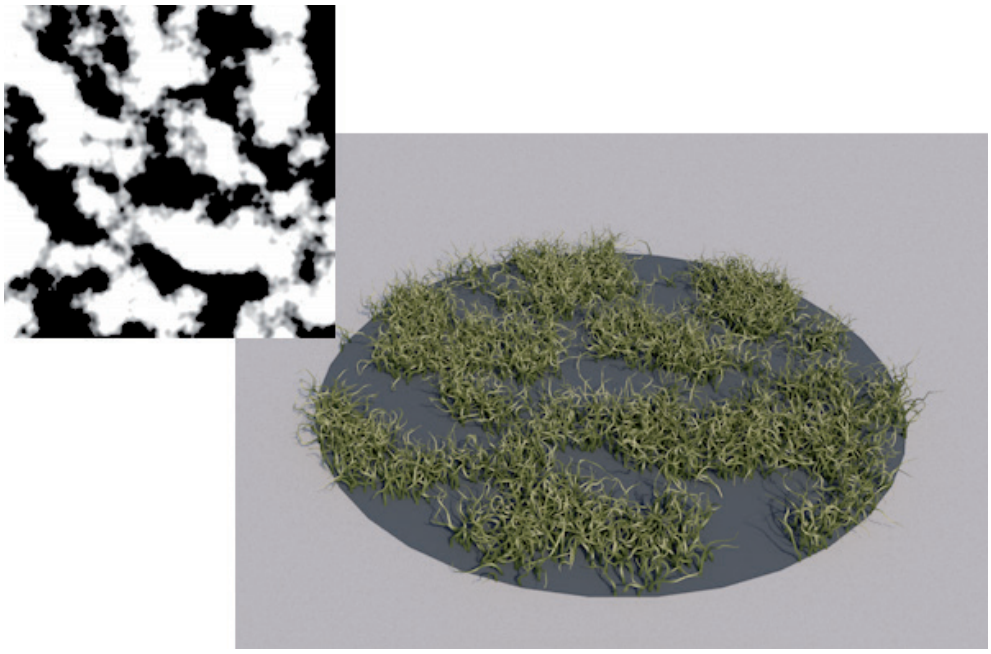
Additional colors can be added by clicking just below the gradient bar. Colors can be deleted by dragging the corresponding color handle upwards or downwards.

If, for example, the lines of a sports field or a rug's pattern should be used to color the grass, use the **Color Texture** setting to load a corresponding image.



To do so, click on the small button at the far right to open a selection window from which you can select your image. Use the **Mix Strength** value to define the strength with which this texture will be mixed with the **Color**.

The size of the grass blades can be defined using the **Blade Length** and **Blade Width** values, whereby **Blade Width** only applies to the blades' roots. Blades of grass always taper in a curved fashion towards the top. The number of blades can be defined indirectly using the **Density** percentage value. If you want to add more variation you can load in image into the **Density Texture** field. The image's dark regions will have little or no grass growth. The brighter the image region, the correspondingly more grass that will be grown in accordance with the defined **Density** value.



Both the **Color Texture** and the **Density Texture** settings use the object's UV coordinates to correctly display the respective image. These coordinates are two-dimensional texture coordinates that many objects have by default, can be added/modified in Cinema 4D or that can be constructed with splines. If an object has no UV coordinates or if the existing ones cannot be used, they can be edited in **BodyPaint 3D**, Cinema 4D's integrated UV and texture editing module. We will discuss this in detail in the material system section.

Alternatively, shaders can be loaded into the **Color Texture** and **Density Texture** fields by clicking on the white arrow next to the setting's name and making a selection from the list that appears. Hence, it's not always necessary to work with bitmaps. Shaders bear the advantage that numerous numeric values can be edited, which lets them be configured directly within Cinema 4D. We will also discuss this and the important role shaders play for texturing surfaces in more detail in the material system section.

The following **Grass** material settings are used to modify the shape of the grass blades. Adding **Crinkle** and **Bend** to the grass blades will add variation and will make the grass look less artificial. This is especially useful when creating other surfaces such as rugs, for example.

Wetness defines the shininess of the grass. The higher the value, the more specular highlights that will be produced when the grass is illuminated. These are the same specular highlights that were discussed in the lights section, i.e., a reflection of the light source in surfaces. Hence, you must have at least one light source with an active Specular option to make the **Wetness** effect visible.

8.7.1 Grass Quality Settings

As we already mentioned, this Grass is not made up of geometry if the **Standard Renderer** is used. On the one hand it's an advantage that the effect uses less memory and renders faster. On the other hand, it required the fine-tuning of additional settings in the **Render Settings** menu with which the quality and type of rendering can be defined. These settings can be found in the **Render Settings/Hair Render** menu, which is also used to edit the **Grass** effect settings. This menu also contains settings that affect hair and play no role for the **Grass** effect. Note that these settings change depending on which renderer you use – the **Standard** or the **Physical Renderer**.

In the following we will discuss the most important **Render** modes and sampling.

8.7.1.1 The Render Settings

There are different **Sampling** settings available in the Standard Renderer's **Render Settings** menu for rendering. Depending on which one is selected, the quality and render times will be affected accordingly. For example, if **Vertex** is selected, one surface sample will be made at the start and one at the end of the hair segment and these will be interpolated for the segment's length. If **Pixel** is selected, a more precise calculation will be made and the hair will be sampled at each of its image pixels. Of course this offers the best quality and will also take correspondingly longer to render. However, this method offers no discernable difference to the **Vertex** method when rendering short hair or hair with many segments.

8.7.1.2 The Objects Tab

This tab contains several general options that affect the rendering of hair, grass and other objects. Hair will only be rendered if the corresponding option is enabled in this menu. This does not apply to the **Grass** effect, and this effect cannot be disabled using the settings in this menu. If **Spline** is enabled, all normal splines with a **Hair** material assigned to them will be rendered with hair. This Hair material is a different from the **Grass** material. The **Grass** material cannot, for example, be used on splines – it can only be applied to actual polygon surfaces.

If the **Polygon** option is enabled, polygonal objects will be covered in fur if a **Hair Material** tag is assigned to them. In this case, a **Fur** object does not have to be created. The **Length** and **Count** values can be used to modify the fur's length and density. The Hair Material tag can be found in the ***Object Manager's*** **Tags**/**Hair Tags** menu. If the **Particle** option is enabled, hair can, for example, even be assigned to an **Emitter** object's particles. To do so, assign a **Hair** material to an **Emitter** object or to the **Thinking Particles Particle Geometry** object. The particles themselves will then be rendered as tiny hair points without having to attach geometry to each particle. This does not apply to the **Grass** effect.

This tab also contains an **Include**/**Exclude** list. Here you can define which objects should be included in or excluded from hair rendering. You can define whether or not **Feathers**, **Fur** or **Hair** object or splines, emitters or polygon objects are rendered with hair. This also does not apply to the **Grass** effect.

8.7.1.3 The Multi-Pass Tab

This tab is only relevant if **Multi-Pass** is enabled in the **Render Settings** menu and a **Post Effect** channel is used. Furthermore, these options will only be made available if the **Standard Renderer** is used. Multi-Passes for Hair cannot be rendered using the **Physical Renderer** – which in turn offers more realistic-looking blur effects. Cinema 4D can output images as a single file or as a multi-pass file. Individual properties will be output separately to separate layers, which makes subsequent editing much easier. Note that the **Multi-Pass** option in the **Render Settings** menu's left column must be enabled. You also have to add the **Post Effects** option from the **Multi-Pass …** selection menu at the bottom of the left column.

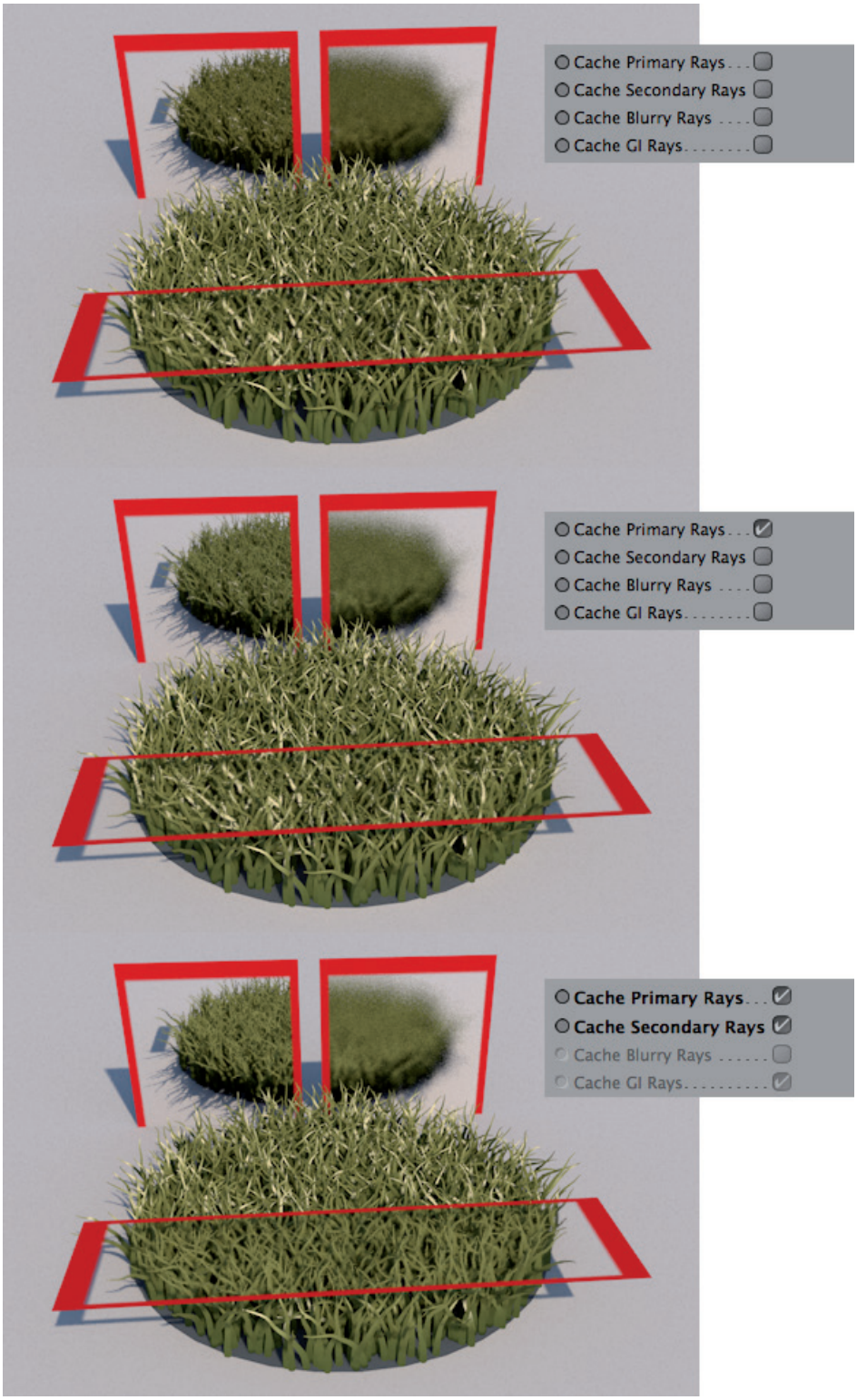
If the **Image** option is enabled, the hairs will be displayed in each saved image as they were defined in the **Render Settings** menu's **Save** setting. We will discuss this in the render settings section. The hairs' depth information and alpha information, respectively, can be integrated using the corresponding image channels if set to **Standard** or saved individually as **Separate** layers. If **None** is selected, this additional information will not be calculated. When using the **Standard** option, make sure that the **Alpha Channel** option is enabled in the **Render Setting** menu's **Save** menu. The remaining options affect the **Hair** material's unique properties and make it possible to save these to separate layers in the **Multi-Pass** file. This is a separate **Hair** material that has nothing to do with the **Grass** effect. This is a separate Hair material that has nothing to do with the Grass effect. The **Diffuse** option affects the diffuse shadows of the hairs with no specular highlight. The **Illumination** option on the other hand only displays the hairs' brightness that is generated by light.

8.7.1.4 Different Settings when Using the Physical Renderer

If the **Render** option is set to **Physical** (**Physical Renderer**) in the **Render Settings** menu, the **Hair Render** settings will change. The **Cache** options are also available. These changes are necessary because the hairs are created using normal polygon geometry if the **Physical Renderer** is used.

These can, among other things, be used to speed up rendering because CINEAM 4D will often only have to reference the saved cache file and not calculate everything anew. Cache files can even be created for animations since not all calculations are dependent on the current camera view. For example, if a force is calculated in an animation in which all other light sources and objects remain unchanged, a cache can be used to dramatically increase render speed.

Caches can also help speed up rendering of still images when using **Team Render**. They can be used to reduce the number of samples required for a given effect. For example, for **Hair** and **Grass** caches, only individual points along the hair lines are made and then interpolated across the length of the hair (**"Grass"** project).



Note that hair and grass are made up of curves, similar to splines, that run through the individual points. The number of points can, however, not be defined on an individual bases for the **Grass** effect.

Cache Primary Rays affect the shading of hairs by light, shadows cast by hairs and their specular highlight. This only applies to the rays that are emitted from the observer's point of view directly onto the hair or grass. If a glass pane would lie in front of the grass, the rays will no longer be primary rays. The same applies for reflective surfaces in which the hairs are mirrored.

Cache Primary Rays uses a reduced number of rays for shading, shadows and specular highlights, which are then applied to the hair or grass in a blurred look. This means that details will be lost when applying complex gradients to hairs. Since these are secondary rays, the hairs will, however, be visible in full quality in reflections or behind transparent objects.

If **Cache Secondary Rays** is enabled, hairs that lie behind transparencies or can be seen in reflections will be sampled with reduced quality and blurred in accordance with their color range. Here the gradients should be as simple as possible as well.

The **Cache Blurry Rays** option affects the blurry reflections or transparencies with which the hairs interact. This can, for example, be a matte metal surface in which grass is reflected or a satin pane of glass behind which grass or hair grows. Caches are especially well suited for use with such surfaces because grass or hair will be rendered in a blurry style anyway.

The **Cache GI Rays** option only affects scenes with **Global Illumination**, which is enabled separately as an effect in the **Render Settings** menu. **Global Illumination** (**GI**) supplements the scenes direct light with diffused light and reflections and can even be used to create bright objects that can in turn be used as light sources. This option is enabled by default because it only slightly affects quality but can tremendously speed up the GI rendering of hair and grass. Scattered diffused light is softer by nature and can be more easily simulated by the often softer calculation by the cache. The settings below let you manually fine-tune the rendering a little more. This cache uses the **GI Sampling** method defined and does not only save each point along the hairs, as the other methods do.

The **GI Sampling** setting defines where along the hair or blade of grass the diffused GI lighting will be calculated. If **Vertex** is selected, a per-point calculation will be done. Especially for long hair and grass, this is generally a shorter distance than the length in pixels for the given resolution, which means it will render faster. The distance between these hair or grass points will be interpolated and the in-between color values will be blurred. If **Pixel** is selected, each of the hairs' visible pixels will be sampled, which is very precise but will also take correspondingly longer to render. This option should be used for extreme close-ups or if complex color or brightness gradients are used along the hair.

If **Root** is selected, the **GI** lighting will only be calculated at the base, i.e., on the object surface on which the growth takes place, of the hair or grass, and at the tip of the hair or grass and then be interpolated. This method can also be applied to short hair and is also the fastest method of rendering diffused **GI** light.

There are various methods available for calculating **GI**, which we will discuss later. What they all have in common, though, is that only a limited number of Samples, i.e., calculation rays, are used to simulate the diffused light. The **GI Quality** value is used as a multiplier for these **Samples**, which search for light information in the region around a rendered point. Simply put, the more **Samples**, the more precise the dispersion of light can be calculated, which can be particularly important when working with complex scenes with numerous angled and rugged shapes.

The **GI Quality** value can also be used to reduce – or to increase – the number of **Samples** used. Note that this function will automatically be overridden by the **Cache GI Rays** option, if enabled.

▶ *See: Exercises for Environment objects*

SUMMARY: ENVIRONMENT OBJECTS

- **Environment** objects make it much easier to create common outdoor environment elements such as skies, floors or grass.
- The **Floor** object is a surface whose infinite size is first seen when the scene is rendered.
- The **Sky** object is a spherical shape with an infinite radius that encompasses all objects in the scene. This object must have an image or material assigned to it – otherwise it will be monotone.
- The **Environment** object can be used to create simple fog. An environment light can also be enabled that will illuminate all scene objects evenly and can be given any color.
- The **Background** object serves as a type of canvas that is always located behind all other objects in the scene, thus giving the impression of a finite distance. Images or colors can be assigned to this object. To use a **Background** object and a **Sky** object simultaneously, a **Compositing** tag must be added, which we will discuss later.
- The **Foreground** object is always the front-most object in the scene for rendering, which means that it will obstruct the view onto other objects. The **Foreground** object must, therefore, have a material with an alpha channel assigned to it. The masked regions will then be transparent, which makes objects behind it visible.
- The **Stage** object is primarily suited for animating objects, e.g., for creating hard cuts between camera angles.
- The **Physical Sky** can be used to simulate various properties of a real sky as well as sunlight.
- The sun will be positioned automatically according to the time of year, time of day and location defined.
- The sky and sun appear in the rendered image and also generate light.
- Clouds can be created as two-dimensional layers or as volumetric elements.
- Atmospheric effects such as fog, haze or visible sunbeams and rainbows can also be rendered, if desired.
- Grass can be grown on polygonal objects.
- Length, color, direction of growth and dispersion of density for grass can be defined using a custom **Grass** material in the *Material Manager.*
- The render quality is defined using the **Hair Render** settings in the **Render Settings** menu.
- The **Hair Render** settings vary according to the Cinema 4D renderer selected. If **Physical Renderer** is selected, the available options can be used to optimize render time and quality via caches.

9 The Material System

So far we've discussed the shapes of objects in detail and how to work with various modeling functions. Of course, an object's shape is very important but its surface color and texture is also very important for its final look. Cinema 4D offers different material systems that can be configured either via a dialog window or a node system. The node-based material systems are targeted at users with a higher level of technical knowledge who want more control over the material creation process and are willing to apply mathematical formulas or tackle manual calculation of rays, etc. This doesn't mean that the dialog-based material systems can't be used to create realistic-looking and beautiful surfaces – these systems are also extremely powerful in and of themselves! You will in fact need a little time to get to know the system and how to use it, e.g., how to work with **Shaders**.

Shaders are an important part of the material system and can be compared to tiny programs that can simulate certain patterns or complex surface properties, which means that you don't always have to use images to create a certain surface.

9.1 The Material Manager

Creating and managing materials is done in a separate Manager, the ***Material Manager***, which is located at the bottom left of the Cinema 4D interface, below the Viewport. Let's create a material to get an idea of how materials work. You can create a material by selecting **Create / New Default Material** from the ***Material Manager's*** menu or by simply double-clicking in the empty space below the menu. The material type selected in the **Preferences** menu under **Material / Default Material** will be created automatically. You can select either the **Default** material or the very similar **Physical** material. The **Uber** material type is based on a node material but also offers a dialog window for modifications. This material is particularly well suited for learning how to work with a node-based material system. In the end, a purely **node-based** material can be defined. Note that basically all material types within a given Project can be used. However, since their calculation and therewith their quality can differ, you should stick to a single material type for your Project. Once a Default material has been created, a small preview image will appear in the ***Material Manager***.

9.1.1 The Edit Menu

The size of the material preview is defined in the **Edit** menu. You can select from **Mini**, **Small**, **Medium** or **Large Icons**. You can also select from various display modes. If **Material** mode is selected, all previews will be arranged next to each other in the ***Material Manager***. If all materials don't fit in the first row, a second row will be added and so on. A scroll bar will be made available, if necessary. If **Material List** mode is selected, the materials will be listed vertically. This is only advisable if the materials with which you are working have very long names. If a **Layer Manager** mode is selected, previews of the images and textures used for each material will also be displayed and columns will be used to show the various material channels.

Material channels are the various effects that can be added to a material. The **Layer Manager** modes will display all elements as a list.

If **Layer Manager (compact)** is selected, all layers and textures of a given material will also be listed. In addition, the textures will be displayed in a column at the right of the material preview. If **Layer Manager (expanded / compact)** is selected, Alpha channels and layer masks will also be displayed. If **Layer Manager (expanded)** is selected, the texture column will be omitted. If **Layer Manager (active texture)** is selected, only the texture currently selected in the ***Object Manager*** will be displayed.

As a rule, you will work in Material mode when creating materials. The **Layer Manager (compact)** mode is, for example, often used when working in BodyPaint 3D. BodyPaint 3D is Cinema 4D'S integrated UV painting system.

The **Edit** menu contains commands for deleting, copying, cutting and pasting materials as well as selection commands, e.g., for selecting or deselecting all materials. Materials can also be selected by clicking on them in the ***Material Manager*** or in the ***Object Manager***. You can also **Shift** + click on multiple materials to make multiple selections, or you can click and drag a selection box around multiple materials in the ***Material Manager***.

Materials can also be duplicated by **Cmd/Ctrl** + dragging and dropping them onto an empty space in the *Material Manager* or *Object Manger*. If you drop a material onto an existing material, it will be replaced (and therewith deleted) by the one dropped onto it. If the replaced material had already been assigned to an object it will be replaced on all surfaces to which it had been assigned.

If one material is **Cmd/Ctrl** + **Alt** + dragged onto another material its properties will be transferred to the material onto which it is dropped and the material will not be deleted. As is the case with objects, materials can also be managed using layers. This, for example, lets you hide a material from view in the *Material Manager* using the *Layer Manager*. The actual benefit of using layers is that each material layer will be assigned a tab in the *Material Manager*, which makes it easier to sort materials. For example, if you create one material layer named Glass and one named Metal, the tabs will be named accordingly and you can navigate between the two material types very quickly. We will show you how to create material layers at the end of this section when we discuss editing materials. The **Edit** menu's **Layers in Single Line** option is used to define how the material layers will be displayed in the *Material Manager*. If enabled, all layer tabs will be positioned next to each other, even if the *Material Manager* has to be scrolled horizontally if there are a high number of tabs. If this option is not enabled, the layer tabs will be displayed in multiple rows to avoid having to scroll horizontally.

9.1.2 Material Functions

To edit a selected material, go to **Edit>Material Editor** or **Edit>Node Editor** if you're working with a node-based material. This can be done by dragging & dropping the preview image from the *Material Manager* onto the object in the *Object Manager* or onto the object in the Viewport. The dialog-based materials can also be edited in the *Attribute Manager*. Alternatively you can select the material in the *Material Manager* and the object in the *Object Manager* and then select **Apply** from the **Function** menu in the *Material Manager*.

Of course each material has a unique name. By default, each material name begins with **Mat.** and is each additional texture is numbered sequentially. The material's name is located below its preview image in the *Material Manager*. Double-click on the name to change it. In addition, the name can also be changed via the **Basic** settings in the *Attribute Manager* or for dialog-based materials via the *Material Editor*.

Especially if a scene contains very many materials it can be easy to lose track of which materials have bee applied to which objects. This is where the following functions can help.

The **Select/Select Materials of Active Objects** command does exactly what it states. Objects to which a material has already been applied and have been selected, e.g., via the *Object Manager*, will be evaluated and their materials will be selected in the *Material Manager*.

Finding specific materials can even be difficult within the *Material Manager* itself, for example, if it contains a great number of materials. In this case you can select the **Select/Find First Active Material** command and the *Material Manager* will automatically scroll to that material.

If a material has been assigned to an object, a **Material** tag will automatically be created and can be found next to the corresponding object in the *Object Manager*. This tag contains information about how the material is applied to the surface. If a material is selected in the *Material Manager*, the object can be selected automatically in the *Object Manager* by selecting **Select/Material Tags/Objects**.

9.1.2.1 Rendering Materials

Materials can be used to create very complex effects that require additional time to refresh the preview image. Generally, preview images are refreshed very quickly once a change has been made to the material but you can force a refresh manually by selecting **Material/Render Materials**. This can, for example, be necessary if you have loaded a scene with new materials. Cinema 4D will then display all material preview images right away since they were saved with the scene.

It can occur that shaders or images used in a particular material are not installed on your computer, e.g., if the scene was copied from another computer. This will first be visible when the materials are re-rendered or if the preview images are refreshed. You can also refresh all materials in a given scene by selecting the **Reload all Textures** command.

With regard to the display of materials in the Viewport, i.e., if a material has been assigned to an object, there is a **Continuous Material Update** option in the **Preferences** menu, which will continuously update the material in the Viewport even while a color slider is dragged, for example.

9.1.2.2 Grouping and Sorting Materials

Materials can be re-arranged in the *Material Manager* by dragging and dropping them into place. This lets you arrange materials according to name or type without having to use the layer system to do so. You can also use the **Material/Sort Materials** command to automatically sort the materials alphabetically.

As already explained, materials can also be sorted to layers. To do so, use the **Edit/Add to New Layer** command. Open the *Layer Manager* using the **Layer Manager** command to rename the layer accordingly. You can find these as tabs in the *Attribute Manager* or in the main Cinema 4D **Window** menu.

In addition to the **Layer** tab there are two other tabs in the *Material Manager*. If All is selected, all materials in the scene will be displayed, regardless of whether or not they have been assigned to a layer. If a material has been assigned to a layer it will have a correspondingly colored triangle in the top-left corner of its preview image. If **No Layer** is selected, only materials that have not been assigned to a layer will be displayed. Materials can also easily be removed from layers. To do so, select the material and then the **Remove from Layer** command from the *Material Manager's* **Edit** menu.

It can happen that materials are created that end up not being used in the final scene, e.g., if objects to which these materials were assigned were deleted. These materials can be removed automatically by selecting the **Remove Unused Materials** command from the **Function** menu. Materials can also be deleted by selecting them and using the **Delete** command in the **Edit** menu or by simply pressing the **Del** or **Backspace** key on your keyboard. Removing duplicate materials is a little more complex. The materials' settings are compared and materials that have identical settings will be optimized so that only one material remains. For this command to work, all settings must be 100% identical.

The **Material Exchanger** can be very helpful if you regularly import files from other 3D applications or CAD programs. Each 3D program has its own material system, which is why only basic properties such ad color or a surface's specular highlight will be imported. However, if you already have a material library set up in Cinema 4D it can automatically be applied to the imported scene. The **Material Exchanger** will ask for a Cinema 4D scene that contains these materials. The material exchange will be done according to material name. Hence, the materials in the 3D program from which the scene was imported should have the same names as the corresponding Cinema 4D materials. For example, an imported material named 'wood' will be replaced by a Cinema 4D material also named 'wood'. After selecting the material scene you will also be asked if all materials should be automatically replaced. If you select 'No', Cinema 4D will ask this question for each material.

The **Reload all Textures** command will update all textures used in the materials. This can be useful if textures were modified, e.g., in Photoshop, after the material was loaded. Cinema 4D will then load the most current version of the texture to the material.

The **Texture** menu's settings otherwise apply only to the creation and management of masks and layers, in as far as you want to edit textures using BodyPaint 3D. We will discuss BodyPaint 3D later. First, we will take a look at the *Material Manager's* **Create** menu.

9.2 The Create Menu

The **New Default Material** command creates a material type that was defined as the default in the **Preferences menu** (as previously described). This command is the same as double-clicking on an empty area of the ***Material Manager***.

The **materials** sub-menu gives you access to all available material types such as the **Default** material with which we will primarily be working. This material is very versatile and can be used to simulate just about any real world – or fantasy world – surface properties. This material can be made to look like wood, water, glass, chrome or skin, and much more. As a rule, this will be the material type you will use for your objects.

Other default materials are also available that are specialized materials designed to simulate specific surfaces. The advantage of using these materials is that the desired surface can be created instantly and only requires fine-tuning to give it the desired look. For example, a **Banji** material from the **Create/Extensions** menu will automatically look like glass, without having to make any modifications to its settings.

You should note, however, that these materials can only be modified using the available settings, which allow only modification within the parameters of the purpose the respective material is designed to serve. Hence, the glass-like **Banji** material cannot be made to look like metal or wood. Furthermore, these **Shader** materials often take longer to render than default materials and these materials are not compatible with all available render methods.

Nevertheless, these shaders look good 'right out of the box'. Use them if they make it possible to achieve the desired effect more quickly but note that there are limitations when using **Global Illumination** and **ProRender** cannot be used with these shaders, i.e., is not able to render them.

The remaining **Edit** menu commands are for saving and loading materials. These can also be saved independently of your Project or loaded from another project without loading any of its objects or settings. The **Save Material As** command can be used to save the currently selected materials under a different name. These files can then be loaded to another Project, e.g., using the **Merge** command.

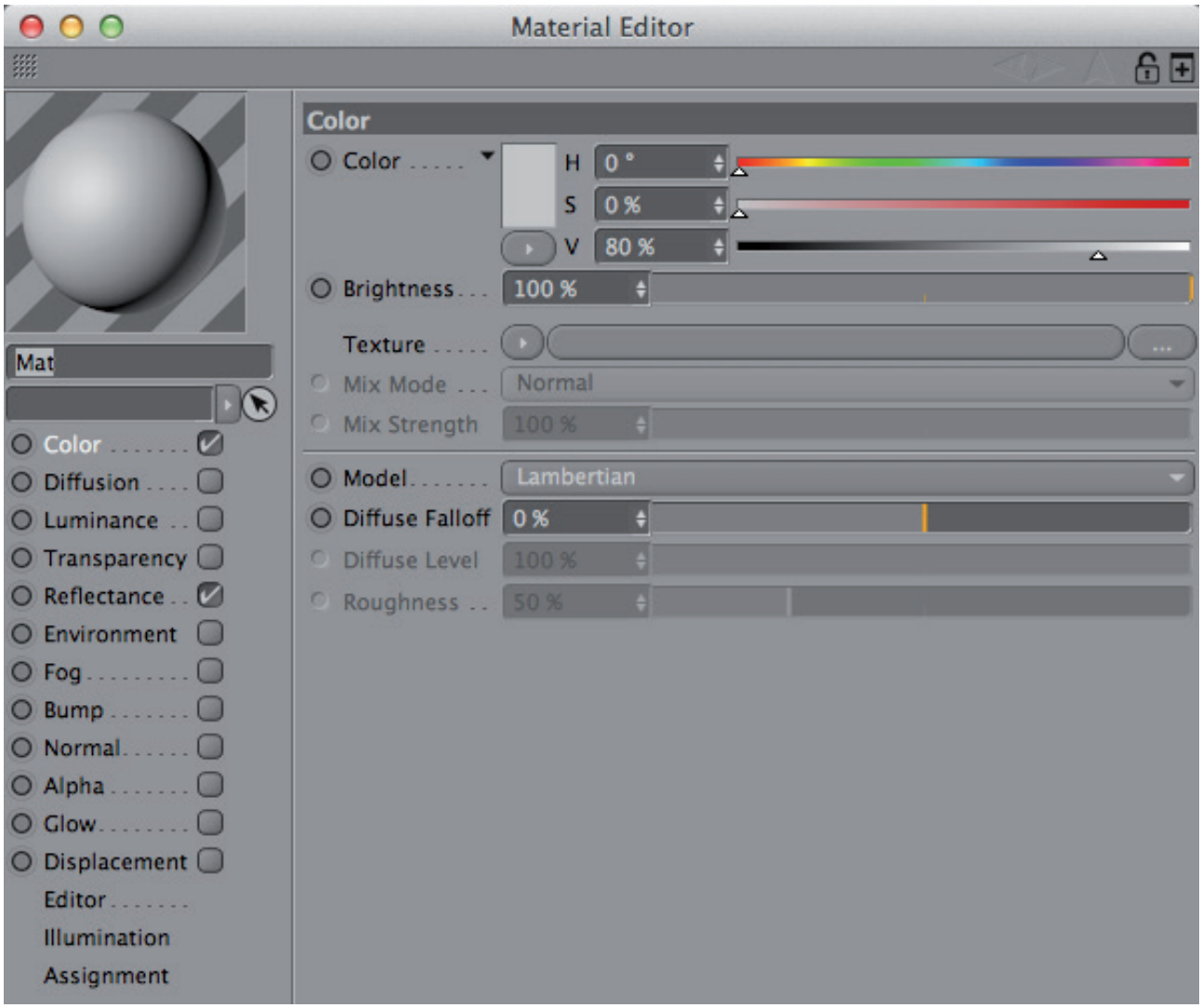
If you want to create a library containing the existing materials, this can be done using the Cinema 4D ***Content Browser***. Use the **Save Material Preset** command to save selected materials to a special **User** directory in the ***Content Browser***. Note that you will always be asked the name under which you want to save the material in the ***Content Browser***. If multiple materials are selected, these will be automatically saved using their current names in the ***Content Browser***.

Materials in the ***Content Browser*** can be added to a scene by double-clicking on their preview image or by dragging and dropping them into the ***Material Manager***. Otherwise they can be loaded using the **Edit** menu's **Load Material Preset** command.

All images or films contained in the materials saved will automatically be saved to the ***Content Browser*** and can be found in a special **tex** folder within the user presets directory.

9.3 The Material Editor

Generally speaking, there are two ways to edit **Default**, **Physical** or **Uber** materials. You can click on a material's preview image in the *Material Manager* to make the material's settings available in the *Attribute Manager* or you can double-click on the preview image in the *Material Manager* to open the *Material Editor* window. For node-based materials, a double-click will open the **Node Editor** where you can edit the material. This window can be scaled more easily than the *Attribute Manager* and can, for example, also be moved to a second monitor. Because some materials have very comprehensive settings, this helps save space in the Cinema 4D interface.



In any case, the dialog-based material's preview image will be at the top left of the window. To be exact, this preview image is in fact a tiny Cinema 4D scene that is also rendered. Otherwise, effects such as transparencies or reflections could not be shown. The shape of the preview object and its lighting can be defined individually. The object only serves to give you an impression of how the material looks on a given surface. Of course the actual model will look much different from the way the preview looks. This is why you need to be able to view the preview in abstract terms to be able to better judge a material's look. An exact representation in the preview is impossible because the scene is also illuminated in its own way.

To change the type of object used in the preview window, right-click on the image to open a selection menu that contains a list of several object types from which you can choose. In addition to **Sphere**, **Cube** or **Torus** you can select from several special shapes. You can also choose from different types of lighting and render modes, which are noted in parenthesis next to the corresponding shapes: **GI** stands for **Global Illumination**, which can be used to reflect light between surfaces and lets luminous surfaces be used to cast light on other objects; **Soft Shadows** means that the object will also cast shadows. This render method requires longer render times and should therefore only be used when absolutely necessary.

Since the preview image is an actual rendering of a tiny scene, its camera angle can also be adjusted individually. To do so, right-click and drag the mouse button on the preview image in the ***Material Editor***. Dragging horizontally will move the camera left or right and dragging vertically will move the camera up or down. The object will always remain centered. This lets you view the object from all sides. To reset the camera's position, right-click on the image and select **Reset Rotation** from the menu that appears.

The last command in this menu is the **Project Settings** command. It's important to note that several material properties also work with measures, such as the **Displacement** effect that moves the points on a surface. The scene's scale must be known in order to judge this affects impact on objects.

This is why there is a **Relative Scale** value directly below the **Preview Size** value. The size of the object in the preview image will not change because the camera's position will be adjusted automatically. The **Original Size** of each selected object will also be displayed for comparison purposes. The difference between the **Original Size** and the **Preview Size** you define will be displayed as the **Relative Size**.

Since materials also contain settings that can be animated, the **Animation Start** value can be used to define at which frame the material's animation should start. This can help reduce the amount of memory required, which would otherwise be reserved for the material preview.

The last two settings can be used to activate additional environment lighting for the preview object. If **Environment Strength** is set to more than 0%, additional light will be activated using the defined **Environment Color**. This special type of light produces no shadows and therefore results in an overall brightening of the object. This is the same effect that was explained in the **Environment** object and light sections. Values that are too high will eventually result in surface shading being lost entirely and the object will look two-dimensional.

9.3.1 Material Preview Type and Size

The material preview's display in the ***Material Editor*** is already much larger than in the ***Material Manager***. It can, however, be made even larger. Right-click on the preview image and select **Huge** from the menu that appears (**Default** = **Large**). You can go even further and select **Open Window**, which will open the preview image in a separate window that can be scaled to just about any size and repositioned. The larger the window the (much) longer the image will take to refresh. This is especially the case when working with the **Blur** effect, for example. You should therefore only use this feature if absolutely necessary. Don't forget that the final judgment of a material can only be made after it has been assigned to an object in the lit scene and rendered. The object type in the floating window can be defined independently of the preview in the ***Material Editor*** window.

If the **Auto** option is enabled in the floating window, the preview image will be updated automatically each time a material property is modified. This option can also be disabled if you want to save render time while modifying a material. Then the preview will only be updated when you click on the **Update** button. All other preview images in the ***Material Editor*** and in the ***Material Manager*** will continue to be updated automatically. The preview can also be animated. To animate the preview, right-click on the image in the ***Material Editor*** and select the **Animate** option from the menu that appears. The animated preview is based on the settings in the **Project Settings** menu. If you want to animate the object itself, right-click on the image and select **Animate**. The preview object will then automatically rotate around its Y axis when the **Animate** option is enabled.

All of the material preview's size and shape settings can also be used for the preview image in the ***Attribute Manager***.

9.3.2 The Basic Tab

As with the objects we've already discussed, materials also have basic settings. For materials, these include the **Name** and **Layer** fields. Selecting a material will make the material's settings available in the ***Attribute Manager***, including the **Basic** tab. In the ***Material Editor*** you will see two fields below the preview image. One contains the material's name and can be edited at any time. The field below it displays the name of the layer to which it is assigned, if any. The small arrow next to the text field lets you add the texture to a new layer or to an existing layer, for example. The **Show in Manager** option opens a ***Layer Browser***.

The remaining **Basic** tab settings in the ***Attribute Manager*** or below the preview image in the ***Material Editor*** affect the material's properties directly. These are the **Material Channels**. Each channel controls a specific material property. If a specific real-world material should be simulated, you must therefore first consider which properties the surface has and add the corresponding channels to the material. As a rule, you will only have to use a few of the available channels to achieve the desired effect. This is especially true for default materials, which are so versatile that they can be used to simulate just about any real-world surface.

In any case, only those channels whose check box is enabled will be active for that material.

In many material channels, images and shaders can be loaded as textures in addition to using simple colors. For example, you can load an image into a Default material's **Color** channel by clicking on the button with the three dots (…) next to the **Texture** field. A shader can be selected by clicking on the small arrow next to the **Texture** setting.

9.3.2.1 The Basic Settings

Clicking on the preview image of the loaded shader or image or clicking on the large button that shows the path of the loaded texture will make additional options available. A **Basic** tab with corresponding settings will also be available. Under **Name** you will find either the term **'bitmap'** if an image was loaded or the name of the **shader** that was selected. The **Layer** option will also be available with which the image or shader can be assigned to an existing layer or added to a new layer. This will, however, not be necessary since the material itself can be arranged in a layer and sorted accordingly. Locking a layer can be helpful in order to prevent an image or shader from being inadvertently replaced or deleted.

Below you will find the **Blur Offset** and **Blur Strength** sliders. These only work in conjunction with **MIP** or **SAT**. Modifications made here will automatically be carried over to the material channel's texture area. Therefore, it doesn't matter if you use the slider here or in the **Basic** settings.

In the **Shader** tab you will find several settings, including **Interpolation**, which are also accessible in the material channel's texture area. Note that when shaders are used, the highest quality **Interpolation SAT** will automatically be used, independent of these settings. In addition, the **Shader** menu will adapt correspondingly. This will be discussed in more detail in ths Shader section.

The **File** setting defines the file path of the loaded image. The button with the three dots (…) can be clicked to select a different image. If the image was edited after it was initially loaded, clicking on the **Reload Image button** will update the image. Click on **Edit Image** to open the image in the external application defined for opening images of that format.

Cinema 4D also supports images with layers, as they can be created in BodyPaint 3D or Photoshop. The **Layerset** option lets you define if existing layers and masks should be displayed.

The options at the top of the dialog window let you select image components that will then be listed below. **Layers/Layer Sets** shows these elements of the image. Layer Alpha will convert the layers selected below to alpha channels. If **Layer Masks** is selected, only those layers that belong to the **Alpha channel** will be shown. If Alpha Channels is selected, will only list the alpha channels that affect the entire image.

The **Generate Alpha** option is only relevant if the loaded bitmap is used within a **Layer** Shader where it is used as a mask. **Show Layer Content** will display a small preview image next to the name of the respective layer in the list.

After selecting the desired layer or mask at the bottom of the dialog window, close the window by clicking on **OK**. The layer selection can also be made via **Cmd/Ctrl+click** to select multiple layers. The **Shift** key kan also be pressed to select a range of elements – all layers that lie between the 1st and 2nd click will be selected and the texture will only show the selected layers as per your selection.

The **Color Profile** setting offers options for defining the color profile of the loaded image. This was already discussed briefly in the **Project Preferences** section. The default setting will attempt to evaluate the color profile of the loaded image. If this does not exist, the standard RGB color space will be used. Alternatively you can use the **sRGB** or **Linear** option.

Below this setting are a few settings that let you define the brightness, gamma or contrast of the texture. The original image file will not be affected, only the evaluation within Cinema 4D will be adjusted correspondingly. The **Exposure** setting is particularly interesting in conjunction with HDR images (High Dynamic Range). Values in excess of 0 will brighten the image; values less than 0 will darken the image. **HDR Gamma** shows the gamma value applied to the texture Gamma value control the display of various brightnesses, e.g., on a monitor. Current Windows and macOS operating systems use a common gamma value of **2.2**.

The **Black Point** and **White Point values** define the brightest and darkest points within the tonal range of the loaded image. A **Black Point value** in excess of 0 will darken the image overall; a **White Point value** below 1 will brighten the overall image. Clicking on **Reset** will set these settings back to their default values.

9.3.2.2 Animation Settings

In addition to loading images or selecting shaders, Cinema 4D also lets you load image sequences or films in QuickTime or AVI format. Additional settings will be made available, which can, for example, be used to define when the animation should start or the frame rate, etc.

The **Mode setting** is used to define if the film should be played **once**, **loop** (repeating) or **ping-pong** (back-and-forth). Ping-pong will play the film forwards and then backwards until the beginning is reached, at which point the process will repeat.

The **Timing** defines the relationship between the rendered Cinema 4D film and the loaded film or series of images. The **Exact Frame** setting will use one frame of the loaded image for each Cinema 4D frame. The framerate must therefore be exactly the same for both. If **Exact Second** is selected, different framerates can be mixed. Exactly one secone of the loaded film will be shown for one second of Cinema 4D animation. If **Range** is selected, a range within the Cinema 4D animation can be defined. **Range Start** defines the start and **Range End** the end of the range. If the wrong values are entered, the speed of the texture film can be modified if the number of images between **Range Start** and **Range End** are not the same as the number of frames of the loaded film.

The **Loop** setting can be used to define the number of times the **defined range** should loop. This is only possible in conjunction with the **Loop** or **Ping-Pong** options.

In order for a sequence of images or a film to be recognized and played back correctly, Cinema 4D must, of course, know the framerate and length of the film. This is recognized automatically for QuickTime or AVI films. For image sequences, Cinema 4D needs manual assistance to define the desired **framerate**. For example, after the first image has been loaded, you can click on the **Calculate** button, which will prompt Cinema 4D to count the images in the target directory. Enter the **Framerate** manually. You can return to the main menu of the ***Attribute Manager*** or ***Material Manager*** by clickint on the Up arrow at the top right of the respective manager.

SUMMARY: 3D VOLUME SHADERS

- All materials are managed in the ***Material Manager***, which is located under the Viewport.
- Shader materials are located in the **Create**/**Extensions** menu and can be used to simulate volumetric materials.
- Shader materials are specially designed to simulate specific materials and therefore cannot be edited as freely as default materials. You should also note that they cannot be used in conjunction with ProRender.
- Double-clicking on the material preview of a **Default**, **Physical** or **Uber** material will open the material in the ***Material Editor***, which can be positioned freely or even moved to a second monitor.
- Node materials have their own Node Editor for editing.
- Materials can be assigned any name and can also be grouped in layers.
- Materials can be saved and loaded separately from the Project's objects.
- The shape of the material preview and the method with which it is rendered can be defined by right-clicking on it in the ***Material Editor*** or ***Material Manager***.
- Generally speaking, a material is assigned to an object by dragging it from the ***Material Manager*** onto the object's name in the ***Object Manager***.

9.4 The Cinema 4D Default Material

Selecting **Create/New Default Material** will create a default material. This material is not geared towards any specific material type. Default materials can be modified to simulate any type of surface.

As you already know, double-clicking on the material preview in the ***Material Manager*** will open the ***Material Editor*** window. This is why we will discuss specific material channels. As we already mentioned, you will never need to use all of a material's channels to create the desired texture.

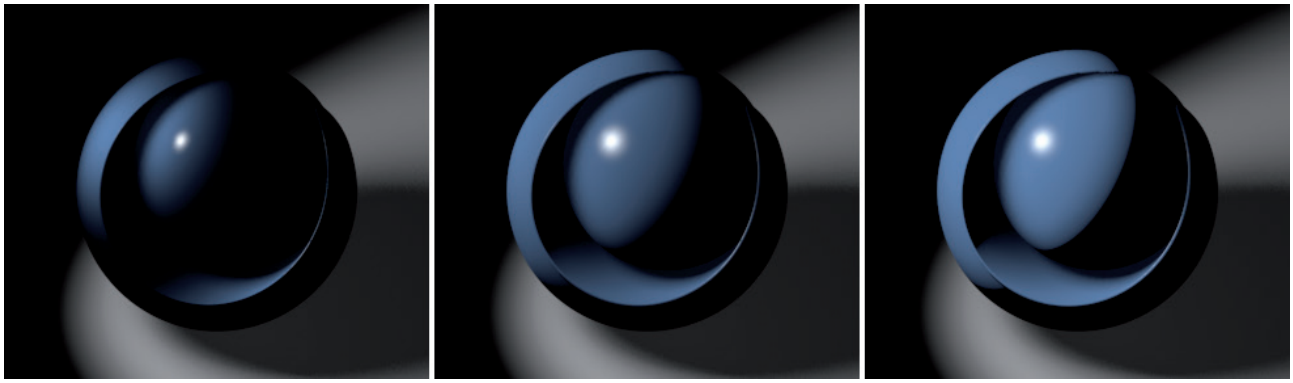
9.4.1 Color Channel

This is where the material's color is defined or where a texture can be loaded. These elements will interact with the scene's lighting and create the surface's shading characteristics. The **Color** defined here can look different when rendered, depending on a light's color or intensity in the scene.

The **Mix Mode** can be used mix the loaded texture with the color defined above. If set to **Normal**, the texture will lie over the color like a solid layer. Reducing the **Mix Strength** value below 100% will lessen the textures opacity accordingly and reveal correspondingly more of the object's **Color**. The remaining **Mix Mode**s are **Add**, **Subtract** and **Multiply**. Multiply can be used to color loaded gray scale images or shaders. Increasing the Brightness value to over 100% will add an artificial look to the brightness of the surface, which can be used to generate strong brightness even for very weak light.

9.4.1.1 Shading Model

The **Model** option defines the type of shading that will be generated on a surface. You can select between **Lambert** and **Oren-Nayer**. The **Lambert** model generates a clearly visible, high-contrast transition between the surface regions lighted directly and those merely grazed by light. The **Oren-Nayer** model is based on the same calculation but offers an additional **Roughness** setting that produces a more homogenous shading. The surface will appear more matte and rougher. The **Diffuse Falloff** setting is available for both models.

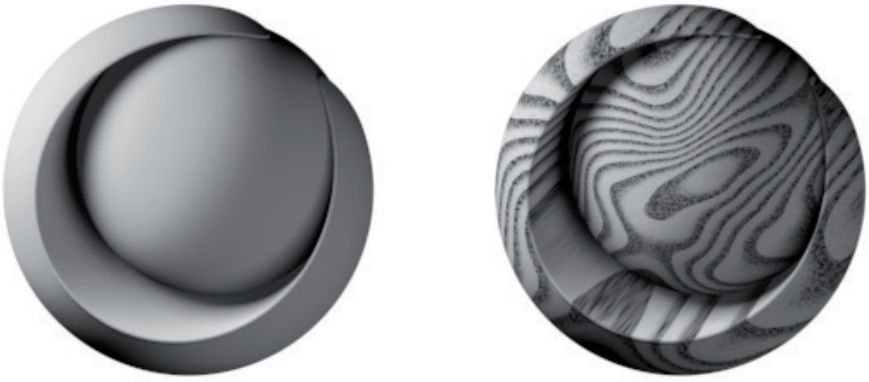


This value defines the shading's intensity falloff. Values in excess of 0% will brighten the surface accordingly, including those regions that are merely grazed by light. This can be useful for materials such as plastics that are penetrated slightly by light, which is subsequently diffused within the material. Values less than 0% on the other hand reduce shading correspondingly until only those regions that are lit directly will be illuminated. This effect is common on polished metal surfaces, for example.

If **Oren-Nayer** is selected, the **Diffuse Level** and **Roughness** settings will be made available. The **Diffuse Level** setting is a general multiplier for surface brightness. The **Roughness** setting defines the light's degree of dispersion. Higher values will produce correspondingly more matte and darker-looking surfaces. Defining the right **Model** is the first step in defining the type of material you will create – smooth and shiny or rough and dull.

9.4.2 Diffusion Channel

Diffusion is a measure of visible brightness on a surface. As a rule, as surface never reflects 100% of the light that hits it. In the real world, light can be swallowed by a surface and not does not affect the surface's shading. This swallowing of light can be compensated for by reducing the **Diffusion** channel's **Brightness** value. This channel can also be used to actively darken the surface, e.g., to simulate dirt or shadows in fine grooves between objects, which is what the remaining settings are designed to do.



Normally, the **Diffusion** channel's darkening will only affect the surface color, i.e., the material's **Color** channel. Enabling additional options, **Diffusion** can be made to also affect the **Luminance**, **Specular** and **Reflectance** channels, which will be discussed later.

As a rule, these channels are used in conjunction with at texture to restrict the darkening to specific regions of the surface. Otherwise, the result would be an overall darkening of the surface.

9.4.3 Luminance Channel

This channel offers the same settings as the **Color** channel. However, this channel does not interact with the scene's lighting. The **Color** or texture will have an additive effect on the surface, independent of any existing scene lighting. Hence, shadowed and illuminated regions will both be brightened. This channel is particularly well suited for simulating light that is dispersed by the material. This effect is called **Subsurface Scattering**.

Numerous types of materials benefit from this effect, including liquids, plastics, marble and human skin. The shaders designed for simulating these surfaces will be described later. Using luminance without such a shader will result in a general brightening of the surface and can lead to a reduction of shading and depth.

Another use for a luminous material is found when **Global Illumination** is enabled in the **Render Settings** menu. Luminous materials will no only add brightness to a surface but will actually emit light, which makes it possible to illuminate surrounding objects. Saturation, brightness and sample precision can also be defined in the material's Illumination channel. We will discuss this in detail later.

9.4.4 Transparency Channel

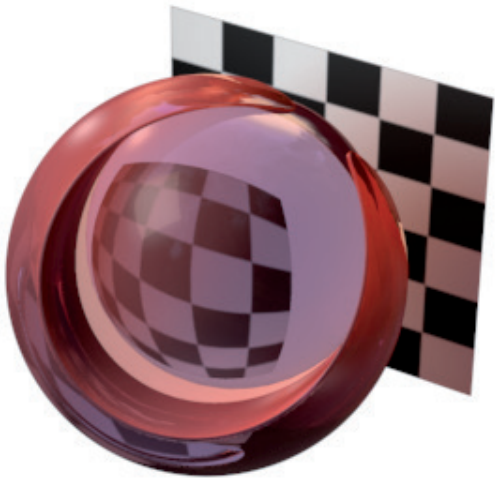
This channel is a key element for creating liquids, glass and similar surfaces. Brightness and **Color** are used to define the degree of transparency. The brighter the color, the more transparent the material will be.

In addition, the **Color** also affects the color of the material itself. If the **Transparency** option is enabled, the material's **Color** channel will automatically be weakened. The more intense the **Transparency** is, the less surface color from the **Color** channel that will be visible. This dependency can be disabled by selecting the **Additive** mode, which will, however, often result in an over-exposed surface. The **Refraction** value should already be familiar from the shader materials. It is used to define the density of the transparent material, which in turn refracts the rays. A fitting value can often be selected directly from the list of common materials via the **Refraction Preset** menu A refraction value

can also be entered manually. The **Total Internal Reflection** option enables or disables the **Fresnel Reflectivity**, i.e., the **Fresnel** effect, which was already discussed earlier in the curriculum. The transparency is automatically reduced in those regions that curve away from the observer. The environment will be mirrored, instead. Hence, the material's **Reflectance** channel does not have to be enabled to achieve this effect. For this curvature-based reflection, an additional ***Transparency*** layer will be added to the **Reflectance** channel. Here you can fine-tune the intensity and quality of the reflection. The **Exit Reflections** supplements the mirroring on the backside of the object. Even though this is realistic behavior, it can lead to double reflections, e.g., on thin glass. If this occurs, simply disable the option.

You already know about the **Texture** setting, which lets you load either an image or a shader to affect the transparency's color. Black regions on the texture will be completely opaque; only regions with a brightness of more than 0% will affect transparency.

Thick glass and liquids in particular change color according to their volume. A single drop of water, for example, is almost perfectly transparent but looking into a deep, clear sea will not reveal the seabed. This is due to the diffusion of light in the water. This effect can be controlled using the **Absorption Color** and **Absorption Distance** settings.



The latter defines the minimum distance that a ray of light must travel before it assumes the color defined by the **Absorption Color** setting. In order for the effect to be visible, the **Absorption Color** must be changed from the default white to a different color. Thin regions of the object will then receive the color from the top **Color** setting and thicker regions of the object will receive correspondingly more of the **Absorption Color**.

Because not all transparent materials let light pass through unhindered, this effect can also be simulated using the **Blurriness** setting, similar to frosted glass or a glass full of milk. In either case, transparency must be used even though you don't have an unobstructed view through the glass. The diffusion of the material increases accordingly with the **Blurriness** value. The increase in render time, however, is even more dramatic. Additional render samples are generated whose total number lies between the **Min Samples** and **Max Samples** values. You should be familiar with this principle from the shader materials section or the **Area** shadow for lights section.

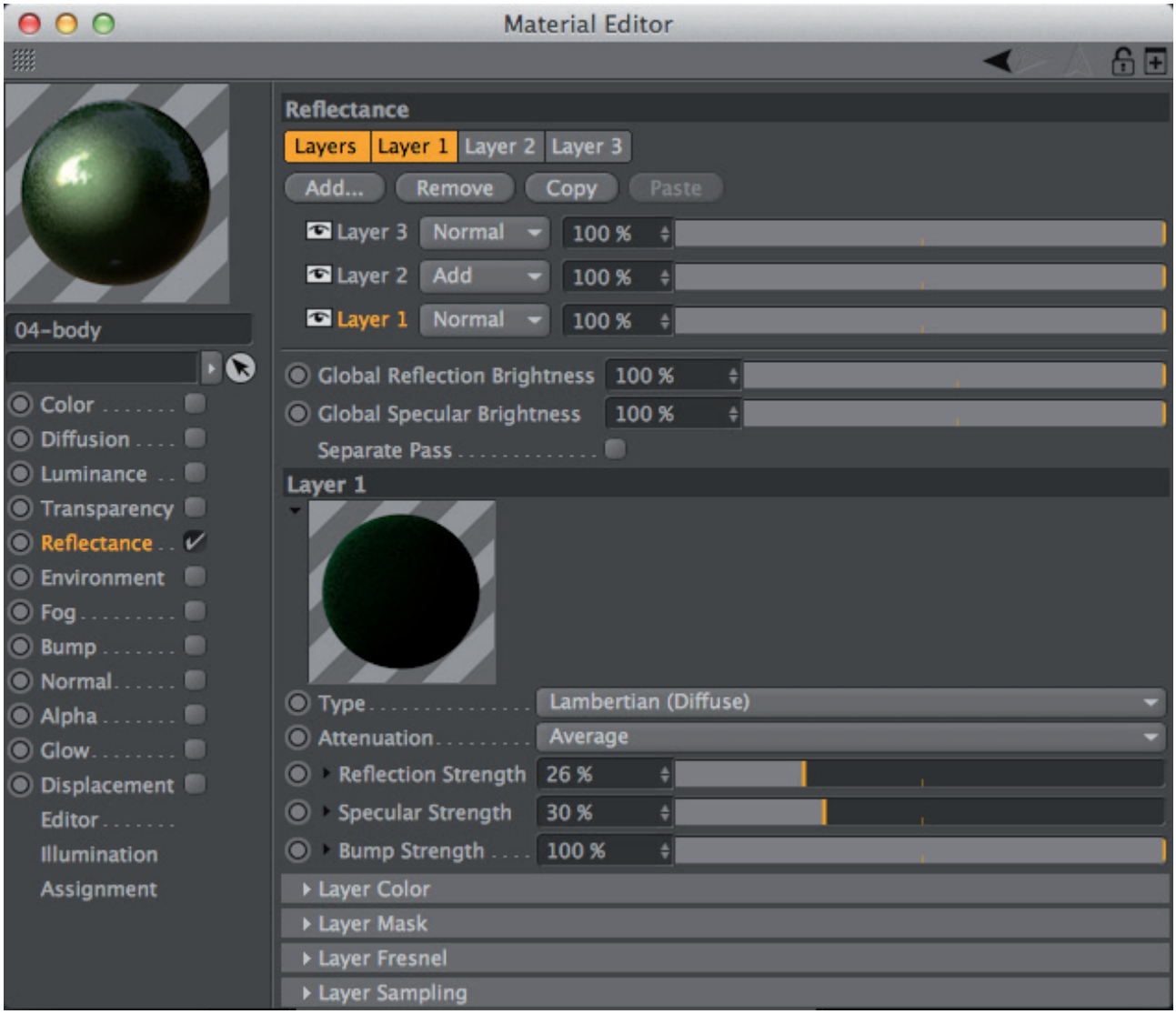
The **Accuracy** value helps Cinema 4D decide whether to use the minimum or maximum number of samples.

A **Blurriness** value of 0% will disable this calculation completely. Note that the blurriness effect only affects transparencies and not the **Total Internal Reflection**'s reflection, which remains unaffected and sharp. To blur these reflections as well, the Roughness value in the **Reflectance** channel's ***Transparency*** layer must be increased accordingly.

9.4.5 Reflectance Channel

This material channel contains various options for creating shiny and reflective surface effects. Various methods are available for creating anisotropic highlights and scratches or shiny fabrics. This channel can be used to create complex layered materials such as car lacquer, for example. Various effect layers are added to create highlights and reflections. Clicking on the **Add** … button will add a new layer, clicking on the **Remove** button will delete the selected layer. Each layer can be set to **Additive** or **Normal** mode. **Additive** is better suited for creating shiny layers and **Normal** is designed to simulate reflections.

In order to better examine the quality of the reflectance directly in the Viewport without rendering the scene, enable the **Enhanced OpenGL** option as well as the **Reflectance** option in the Viewport's **Options** menu.



Each layer's opacity can be defined individually. Beneath this option you will find two sliders: **Global Reflection Brightness** and **Global Specular Brightness**. These can be used to mix the effect of all active layers. To get an impression of a layer's effect you can click on the eye icon at the left of the layer's name to toggle its visibility.

9.4.5.1 Layer Types

Every layer can be basically configured via the **Type** menu. The first four options **Beckmann**, **GGX**, **Phong** and **Ward** are very similar. These are different render processes for calculating the dispersion of render samples on a surface. The slight differences between the various modes become more visible when the surface's **Roughness** value is increased. If no other settings are modified, the **GGX** mode in particular will then render a more wide-ranging effect than the other modes. With its optimized algorithms, the **Beckmann** mode offers a good compromise between realism and render speed.

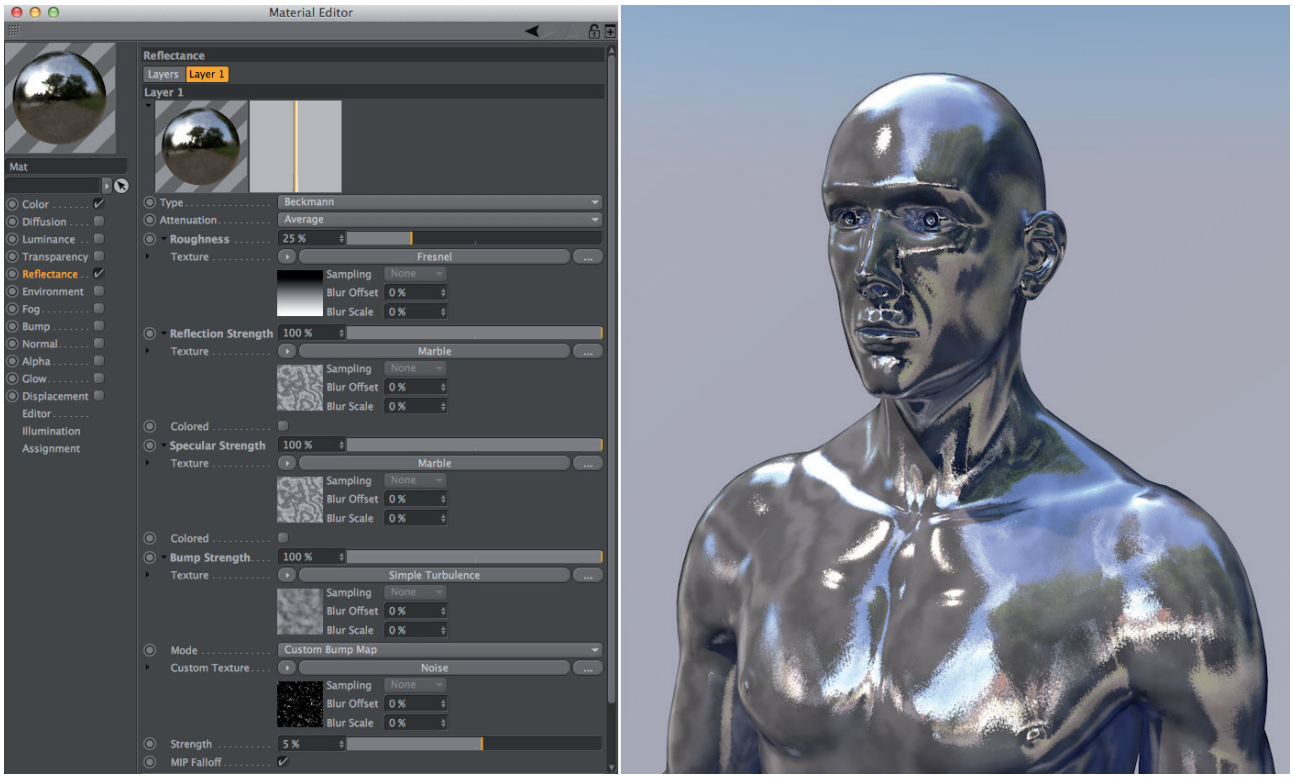
The previously mentioned **Roughness** value spreads the render samples and simulates a porous and rough surface. Specular highlights increase in size as the **Roughness** value increases and reflections become more intense.

The **Reflection Strength** and **Specular Strength** can be defined individually for each layer, which makes it possible to create layers that only use one or the other effect. A preview graphic shows the ratio of specular highlight to reflection, which are separated by a vertical yellow line. The gray curve on the left represents the intensity and dispersion of the reflection on the surface.

Many materials also use the **Bump** or **Normal** channels. These channels change the angle of the surface Normals so create slight irregularities to simulate scratches, pores or wood grain. The object's shape itself will not be affected. A **Reflectance** channel's **Roughness** setting defines the degree to which the surface Normals affected by **Bump** or **Normal** channels should be taken into consideration for rendering highlights or reflections. A **Bump** strength of 0% will use the object's original shape for the reflectance effect. The **Bump** and **Normal** channels' effects will then be restricted to modifying the Color channel's surface shading and the effect of the **Transparency** channel. The Bump and Normal effects are well-suited for simulating small surface undulations.

A unique Bump or Normal texture can also be used for a reflectance effect. To do so, click on the small black triangle next to the **Bump Strength** setting and load a corresponding texture into the **Texture** field. Use the underlying **Mode** menu to select the type of texture.

The same method can be used to define the **Roughness**, **Reflection Strength** or **Specular Strength** using a texture. As a rule, grayscale images or shaders are used that define the intensity of a layer's individual properties, as shown in the example below:

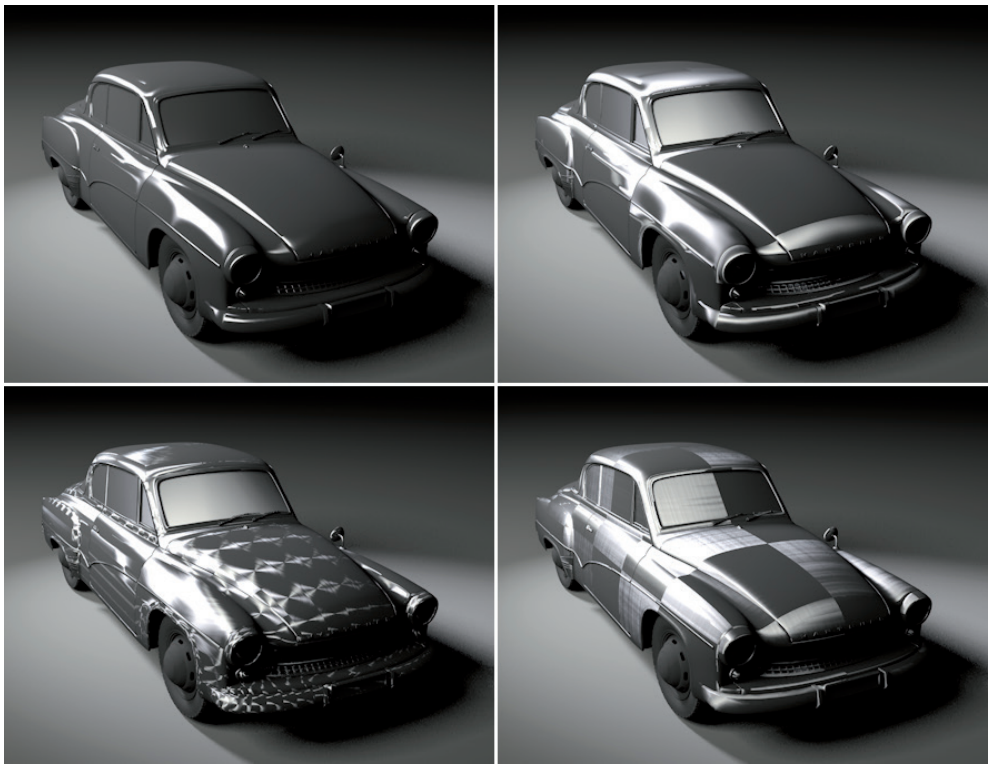


Enabling the **Colored** option for **Specular Strength** or **Reflection Strength** will cause the respective effect to be colored using the color defined in the Color channel. Otherwise, the **Layer Color** settings can also be used to color a layer individually.

9.4.5.2 Anisotropy

Anisotropy is the simulation of fine scratches and ridges on a given surface. In the real world, these very fine irregularities cause highlights to be distorted in the direction of the scratches. This effect is commonly seen on brushed metal or on the ridged side of a DVD. You first define the direction in which the scratches should run using the **Re-projection** menu settings. If **None** is selected, the direction of distortion and the direction in which the ridges run will depend strictly on the viewer's angle of view and the position of the light source. Select **Planar** if you want to define the direction in which the distortion and ridges run. You can then use the **Angle** value to rotate the scratches in any direction. The **Scale** value defines the distance between the scratches but does not affect the distortion but does play a role when modifying the **Scratches** menu's options. The same applies to the **Offset U/V** settings, which can be used to move the scratches on the material's projection plane.

If you want to create circular ridges like those found on the underside of a DVD, set **Re-projection** to **Radial**. The center of the radially distorted ridges can be positioned at the center of the texture tile by setting **Offset U** and **Offset V** each to 50%. You will get concentric circles if the material is applied with **Area** mapping. Here the **Scale** and **Offset** values also primarily affect the ridges, which must be enabled in the **Scratches** menu. The **Count** setting (if **Re-projection** is set to **Radial**) is designed for use with the **Pattern** setting because it defines the number of times the selected pattern will be repeated on the tile. The following image shows the difference between the various **Re-projection** types. Note that the result also depends on the type of texture projection defined in the Texture tag.

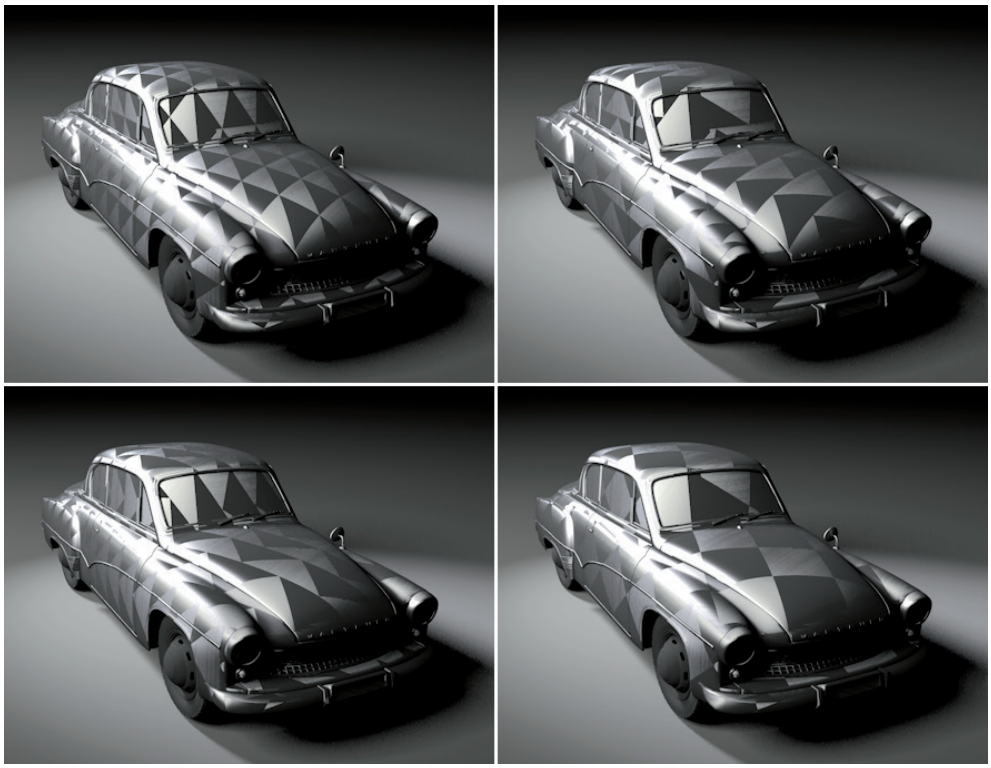


Using the **Pattern** menu is easier if **Scratches** is set to **Primary**. An additional preview window will be displayed in which the design of the simulated scratches is displayed. The scratches will also be made visible on the respective surface. If **Scratches** is set to **None**, only the specular effect itself will be distorted and the surface will remain smooth. Changes made to the **Scale** and **Offset U/V** settings will be displayed in the preview window. If **Pattern** is set to **Circular** in conjunction with **Re-projection** set to **None** or **Planar**, overlapping concentric radial ridges will be created. This is a typical pattern found on stainless steel, for example, as well as on high-end wristwatch casings.

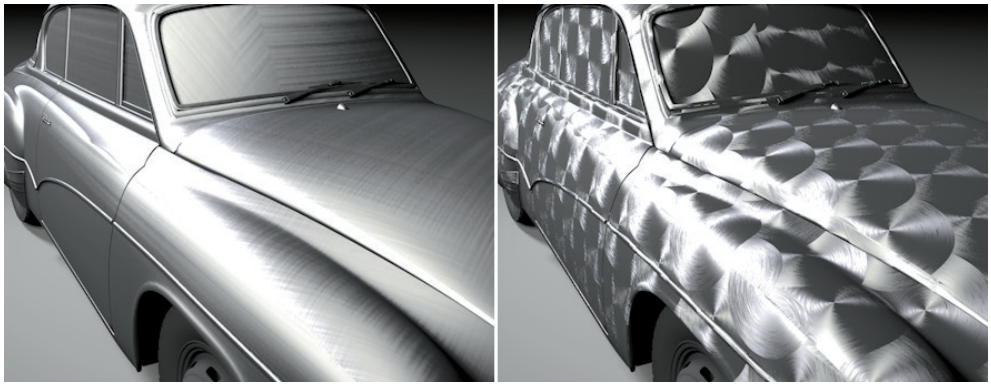
If **Pattern** is set to **Box**, the orientation of the scratches will rotate in steps of 90° and will remain parallel to the edges of the texture tile. These patterns are shown in the image below:

If **Pattern** is set to **Diagonal**, the orientation of the scratches will rotate by 90° when it reaches the edge of the texture tile. If set to **Lattice**, rectangular regions filled with scratches will be created; the orientation of the scratches will rotate 90° for each region. You can also create a **Custom** pattern and load your own texture to define the orientation of the scratches. Click on the small black triangle next to the **Max Angle** setting to access the **Texture** field into which you can load your texture. The Max Angle value defines which direction will be represented by the white regions in the texture. Black generally represents 0°, i.e., horizontal scratches in the texture. The following images shows an example:

If such patterns are tiled multiple times on a surface, e.g., when small values are defined, hard outlines can be produced where two tiles meet. However, you can also mirror individual patterns using the **Mirror** setting's options: **Mirror U**, **Mirror V**, **Mirror U+V**. The following image shows an example of a mirrored pattern:



In addition to the primary scratches, additional scratches can be simulated that lie at a 90° angle to the primary scratches. These secondary scratches can be rendered alone or in combination with the primary scratches. The **Secondary** option can be found in the **Scratches** setting's drop-down menu. If **Scratches** is set to **Primary + Secondary**, an additional preview window will be displayed. The secondary scratches' rotation is linked to the primary scratches' orientation, which means that both scratches will rotate together. The following image shows various patters with primary and secondary scratches:



The direction of the distortion of the specular and reflection can be rotated independently of the orientation of the scratches using the **Orientation** value. The **Anisotropy** setting defines the degree of distortion that should take place. Higher values will produce correspondingly thinner highlights and lower values will produce more circular highlights, which can also be created without using an anisotropy layer. The only difference compared to the other layer types is that the primary and secondary scratches will still appear as a bump map on the surface. Both anisotropy and the orientation can be varied on the surface using grayscale textures. Click on the small black triangle next to the respective setting to make the **Texture** field available.

Note that the scale of the anisotropic highlight distortion is mainly dependent on the layer's **Roughness** value.

The depth of the scratches can be defined using the **Primary Amplitude** and **Secondary Amplitude** settings. The **Primary Scale** and **Secondary Scale** settings define the distance between the scratches. The previously mentioned **Scale** setting works in a similar fashion but it scales the defined pattern and does not affect the spacing of the scratches.

The **Primary/Secondary Length** settings can be used to define the length of the scratches. Short scratches produce more overall variation.

Primary/Secondary Attenuation are used to adjust the intensity of scratches that lie farther away from the camera when viewed at a flat angle. The farther away the scratches lie, the less intense they will appear. This reduces the danger of encountering the Moiré effect and flickering when an animation is rendered.

The following image shows various types of scratches:

9.4.5.3 Diffuse Layers

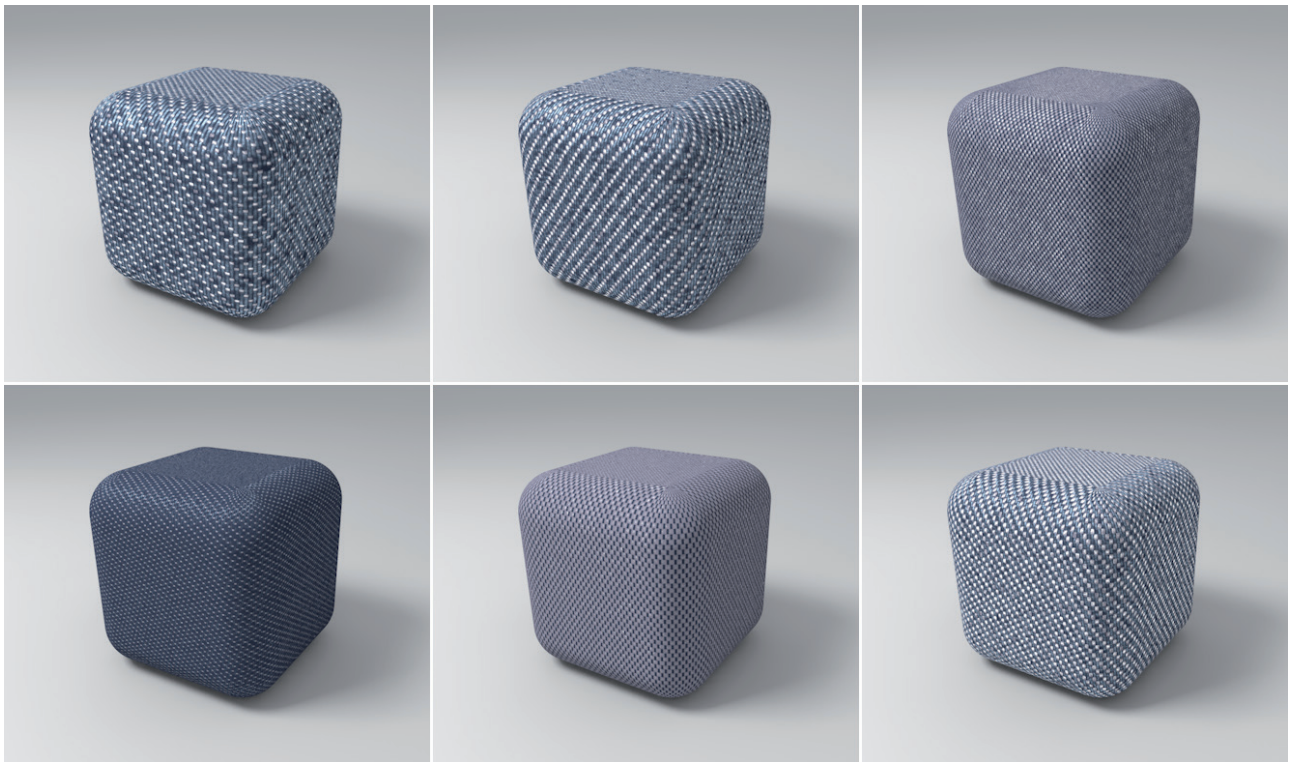
In principle, surface colors can also be rendered using an extremely widely dispersed reflection or a very rough highlight. To do so, **Area** lights with visible reflective properties, Physical lights or HDR images, or luminous materials for scene lighting must be used. This effect can be supported by soft diffuse highlights that work with the remaining light sources, e.g., omni or spot light sources. For these instances, Cinema 4D also offers diffuse layers in the **Reflectance** channel: **Lambertian (Diffuse)** and **Oren-Nayer (Diffuse)**, which mainly use specular and reflection strength to define the intensity of reflection and highlights. These effects are very blurred in both modes, i.e., very diffuse. The **Oren-Nayer (Diffuse)** mode offers an additional **Roughness** setting that can be used to intensify the diffuse effect. In principle, this can replace the material's **Color** channel completely and an entire material can be set up within the **Reflectance** channel itself. In fact, this forces us to use a physically-based material, which can produce especially realistic-looking surface properties. If this is what you want to achieve, you can go directly to the **Create** menu in the *Material Manager* and use a pre-defined **Physical Material**. This material will already contain a diffuse Reflectance layer in the **Reflectance** channel and the **Color** channel will be deactivated.

However, the fact that rendering real diffuse reflections can take longer than using the **Color** channel and the restrictions when used in conjunction with Global Illumination makes this option less feasible. GI automatically reduces the sample precision when transparent or reflective surfaces are rendered. This reduction in precision cannot be completely deactivated. This means that you can stay more flexible if you define the basic surface color using the **Color** channel and use reflective or specular properties to adjust the reflectance. The diffuse modes are also well suited for special effects since the **Bump Strength** setting and the modes can be used to fine-tune bump maps individually.

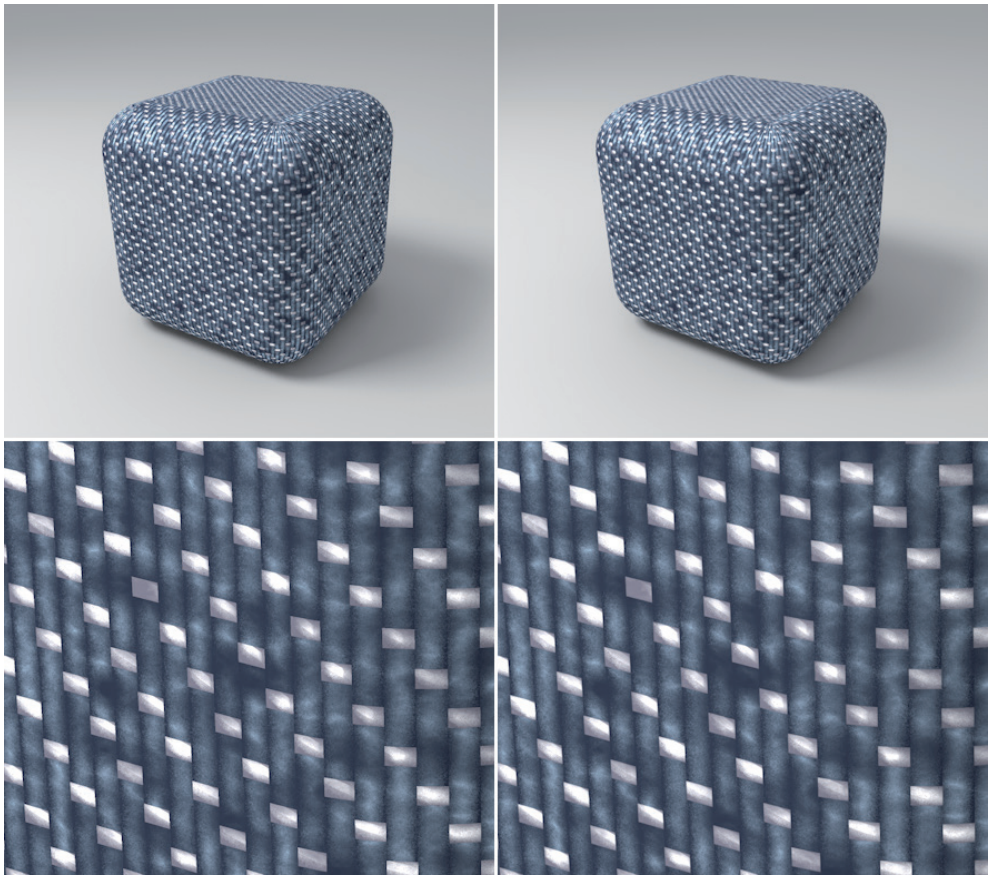
9.4.5.4 Irawan (Woven Cloth)

This option can be used to simulate woven cloth and various types of weave patterns. This layer can render both diffuse colors as well as highlight and reflection. As a rule, the material's **Color** channel should be disabled when using this layer **Type** to make sure the correct colors are maintained for the cloth. The **Layer Cloth** menu's **Preset** menu offers several types of common weave patterns and color combinations. When a preset is selected the values in the Layer Cloth menu's settings will change accordingly. Each setting can, however, be modified individually, if desired. The first setting defines the cloth's **Pattern**.

The following image shows the available patterns:



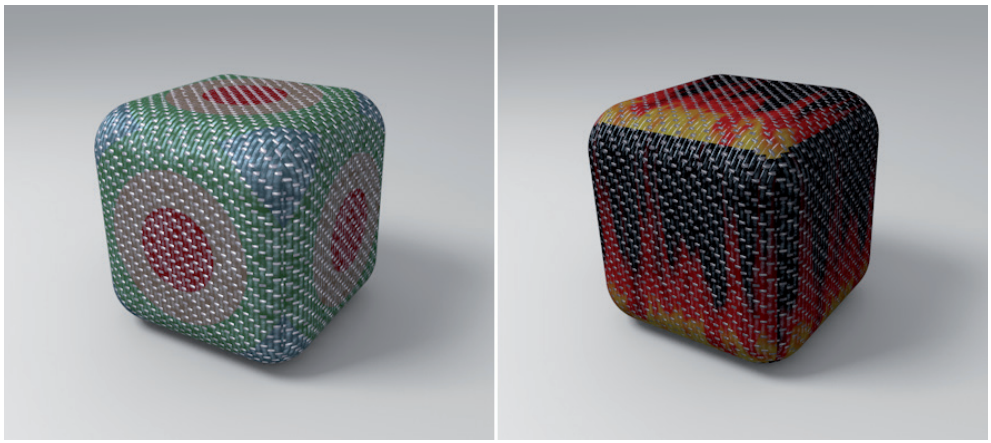
The render quality of the cloth can be defined using the **Quality** setting's options. You can select from **Low**, **Medium**, and **High**. For extreme close-ups, the **High** setting should be used. Otherwise **Medium** or **Low** will be good enough. The better the quality, the longer rendering will take. The following image shows and example of a rendering with **Low** quality (left) vs. **High** quality (right). The difference is marginal and is noticeable primarily by the sharper highlights.



The cloth pattern can be rotated individually using the **Orientation** setting and scaled – or distorted – using the **Scale U/V** settings. The size of the highlight on the thread can also be adjusted using the **Highlights** setting. Larger values will make the highlights softer and thus make the yarn look correspondingly rougher.

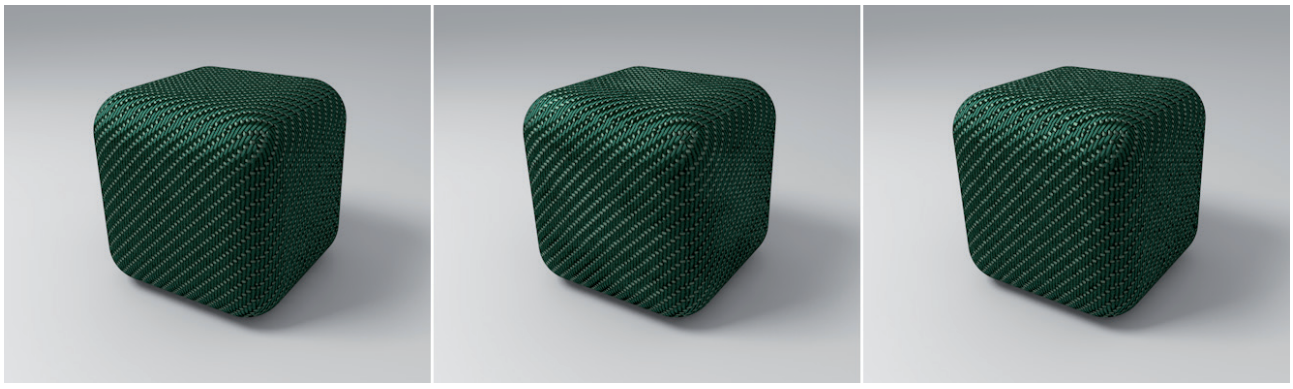
The **Scattering – Uniform** has a similar effect because it can be used to strengthen the cloth's highlight or reflective properties. The **Scattering – Forward** setting on the other hand only affects those regions within highlights. This setting can be used to adjust the contrast of the highlights. The smaller the value, the more intense the highlights and reflections will be.

In addition to the pattern and scale of the cloth, the color also plays an important role in the overall design. This is what the **Diffuse Warp** and **Diffuse Weft** settings are for. These settings and the ratio between warp and weft threads define the color for the entire cloth, which in turn is defined by the **Pattern** setting. The thread's highlight properties can also be defined using the **Specular Warp** and **Specular Weft** settings. As usual, all properties can also be controlled using textures, which can be loaded in the respective setting's **Texture** field (which can be accessed by clicking on the small black triangle next to the setting). When using textures, note that they will always be multiplied by the respective color value. If you want to avoid any deviation from the texture's colors, use 100% white as the diffuse or specular color. The following image shows examples for a uniquely colored cloth:



Finally, a realistic-looking effect is also achieved by adding slight deviations and variations. This can be done using the settings at the bottom of the Layer Cloth menu. The **Noise Strength** adds variation to the thread color. A noise pattern is used to generate random variations in brightness. This pattern can be scaled using the **Noise Scale** value. Lower values will produce correspondingly more variations in a smaller area. Note that the size of the noise pattern can be scaled using the Scale U and **Scale V** settings.

The **Yarn Noise (Warp)** and **Yarn Noise (Weft)** settings can be used to add variation to the weave pattern. The higher the value the more the weft and warp threads will deviate from their original pattern. The **Yarn Noise Scale** setting can be used to adjust the frequency of this effect. The following image compares an original weave structure with the same structure to which a strong noise effect has been added for warp and weft threads.



9.4.5.5 Legacy Modes

Specular and reflective effects were handled differently in older versions of Cinema 4D. To ensure that there are no visible differences in these effects when older projects are loaded, the **Reflectance** channel's **Type** menu offers legacy options. These should not be used when creating a new material. Besides, the other options offer many more design possibilities.

The **Reflection (Legacy)** option simulates the Reflectance channel's effect in older versions of Cinema 4D. The **Specular – Blinn (Legacy)** and **Specular – Phon (Legacy)** options simulate the older Specular channel and its effect on the Phong and Blinn shading models. These have been replaced by the Lambertian shading model.

9.4.5.6 Layer Color Settings

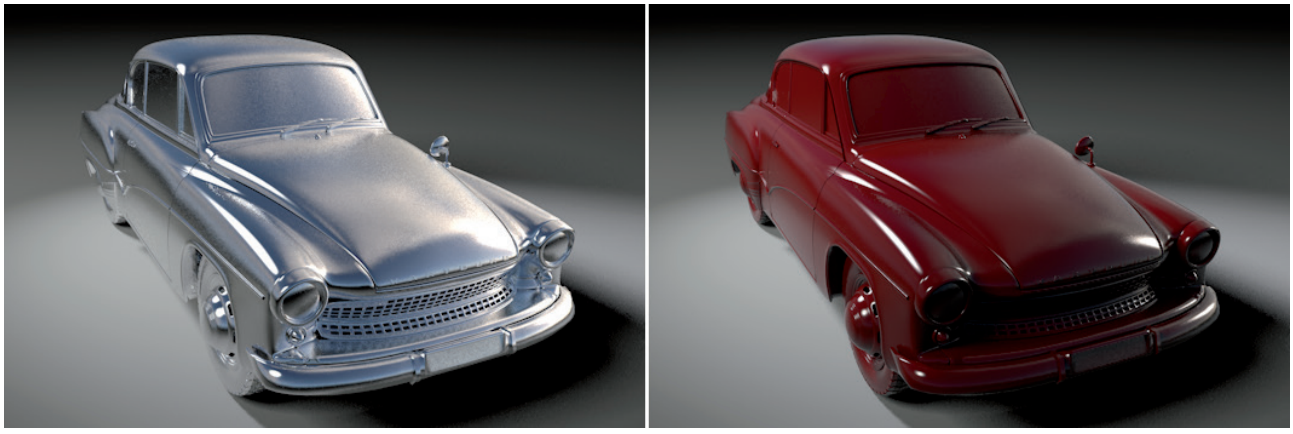
The **Layer Color** settings can be used to color the Reflectance layer's highlights and reflection. The **Color** that is defined will be multiplied with the highlight's and reflection's intensity. If a dark color is defined, the layer's intensity will automatically be reduced. An image, video or shader can also be loaded into the **Texture** field, which can be used to mix various colors in the highlights and reflections. These settings are the same as those in the material's **Color** channel, which is why they also offer **Mix Mode** and **Mix Strength** settings for mixing a loaded texture with the defined color.

9.4.5.7 Layer Mask Settings

Sometimes the highlights and reflective strengths on a surface need to vary, e.g., if part of a shiny metal surface is rusty and dull. The **Layer Mask** settings can be used to partially subdue or completely hide the Reflectance layer's properties. The easiest way to do this is by adjusting the **Amount** value, which is multiplied by the defined Color and any texture that may have been loaded. The remaining brightness will then correspond to the layer's opacity. A texture can be used to simulate gradations in opacity. The **Color** and **Texture** can also be used to add color to the highlights and reflections. As usual, a texture can be mixed with the defined Color via the **Mix Mode** and **Mix Strength** settings. As you can see, these settings have a lot in common with the Layer Color settings, which can also be used to adjust the layer's opacity. The Layer Color settings can be used to adjust the layer's color and the Layer Mask settings can be used to mask regions and/or fine-tune the layer's opacity.

9.4.5.8 The Fresnel Effect

The Fresnel effect simulates the visual change of surface properties depending on the angle from which the surface is viewed. With regard to the material system, this effect is suited for use on reflective surfaces because it is a characteristic of these types of surfaces. This is why numerous presets are available, which are divided into two categories: **Dialectric** primarily for fluids and glass and **Conductor**, primarily for metal surfaces. The refraction index (**IOR**) defines the degree to which a visual sample will be diverted from the surface. For example, common values are 1.333 for fluid water and 1.6 for normal glass. Since these are physical values they can also be found by looking them up online, for example, for the material with which you are working. If a material's IOR is known it can, of course, also be entered manually. A Fresnel effect for a Conductor material also uses an **Absorption** value, which defines the surface's overall reflective strength. Higher values increase the surface's overall reflective strength accordingly. The **Strength** value defines the individual Fresnel effect's overall strength. The **Invert** option will invert the dispersion of the highlights and reflections. The **Opaque** option will suppress underlying Reflectance layers in those regions that are weakened by the Fresnel effect. Deeper highlight and reflection layers will then only be visible at this level in regions hidden by the Layer Mask settings. The following image shows how the Fresnel effect works. At left without Fresnel, at right with Fresnel. You can clearly see how the Fresnel first makes the Color channel visible beneath the reflection.



9.4.5.9 Layer Sampling Settings

As soon as you add blur effects, rendering has to be done on a per-pixel basis. This method is used, for example, when rendering antialiasing, area shadows, subsurface scattering, motion blur or blurred reflections and transparencies. The **Sampling Subdivisions** value indirectly defines the maximum number of render steps per pixel. The higher the value, the more precise rendering will be and the longer rendering will take.

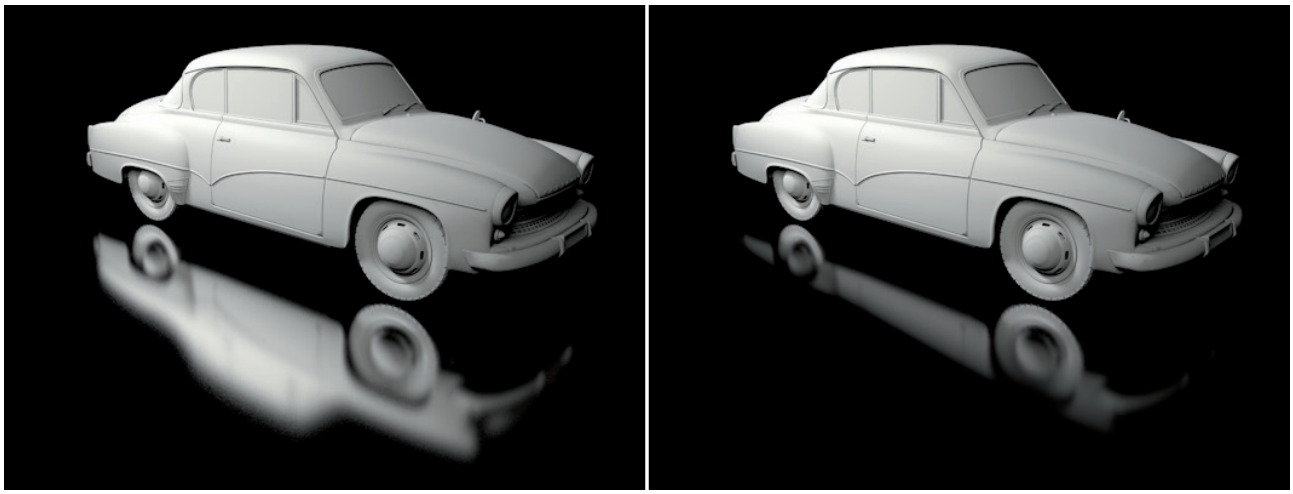
Since reflection on a reflective surface can also contain other reflections, the **Clamp Secondary** setting can be used to help alleviate a problem that can occur in scenes with intense HDR images and multiple reflective surfaces. In some cases, extremely bright pixels can appear in reflections. These are called fireflies. Normally, these can be suppressed by increasing the sample count, which also slows rendering. The **Clamp Secondary** setting lets you filter out such extreme contrasts. The higher the value, the more contrast that will be filtered out. The **Cutoff** setting affects the opposite end of the spectrum – it lets you remove weakly lit areas from reflections. The larger the value, the greater number of weakly lit areas that will be filtered out. This speeds up rendering but should not be overdone. In the real world, everything will be visible in a given reflection. The following image shows the effect of the **Cutoff** setting with increasing values.

To better handle extreme situations, e.g., two facing mirrors, a limit for the number of reflections per render ray can be defined in the **Edit Render Settings** menu using the **Ray Depth** value in the **Options** menu. If this value is exceeded, the color defined in the Layer Sampling menu's **Exit Color** setting will be used to render these pixels. This color is black by default because the default empty space in Cinema 4D is also black.

The **Separate Pass** option can be enabled if you want to save individual reflections or highlights when using the Multi-Pass rendering option. This can make the compositing phase a lot easier. When defining a **Multi-Pass** layer for reflections or highlights you can decide if you want to output separate materials or if all reflections and highlights should be combined in a single layer. An additional option is available for rendering reflections and specularto a separate layer. This will affect all Reflectance layers that have the **Separate Pass** option enabled. Enabling the **Separate Pass** option will only have an effect if the Multi-Pass option is also enabled in the Edit Render Settings menu and configured for **Reflection** and **Specular**.

9.4.5.10 Distance Dim

In the real world, there is no limit to the distance from which a reflective surface can reflect another element. If a distant element is not reflected in the surface, this is because of the low quality of surface's reflective properties. However, there is a method with which distance objects can be prevented from being reflected in highly reflective surfaces. This is done using the **Distance Dim** option. If enabled, you can use the **Distance** value to define the maximum distance at which elements will be reflected in the surface. The intensity of the reflection in relation to the distance from the surface will be displayed in a preview image. This will be linear by default. However, this can be modified using the **Falloff** value. If set to less than 0 the intensity of reflections will diminish very rapidly and diminish more softly near the defined **Distance** value. The **Distance Color** setting is similar to the previously described **Exit Color** setting. It lets you define the color with which pixels should be rendered that lie beyond the defined **Distance** value. As a rule, black should be used because nothing should be reflected beyond this point. Other settings can, however, be useful for creating special effects. The following image shows the effect of **Distance Dim** on a reflective floor.



9.4.6 Environment Channel

Even reflections with a maximum intensity can only be seen on objects with a reflective surface. Because 3D space is itself empty and black, an environment must be created in a 3D scene. This is where the **Environment** channel comes in. An image can be loaded into the channel's **Texture** field, for example a panorama image, to simulate a reflected environment. This is easier than actually modeling an environment in 3D if all you need it for is the reflection.

The **Tiles** values are used to define the number of times the texture will appear on the surface. By default, the texture will cover the entire surface of the object, which is why both **Tiles** values are set to 1 by default. Increasing the **Tiles X** value to 2, for example, will scale the texture so it appears twice along the X axis; increasing the **Tiles Y** value will affect the texture's repetition along the Y axis.

If the **Exclusive** option is enabled, the texture will only be visible where no real 3D objects are reflected in the object's surface. If the **Reflectance** channel is also enabled, the **Environment** channel's **Texture** will overlap the surface's actual reflection.

9.4.7 Fog Channel

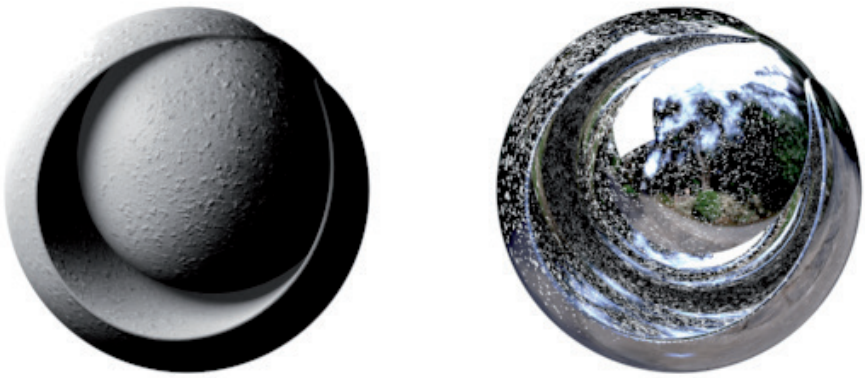
Fog has already been discussed quite a bit in this curriculum. As with the **Fog** shader, this channel can also be used to fill an object with fog. However, here you have far fewer settings and options from which to choose. For example, the options for adding variation and billowing are missing completely, as is the option for letting the fog move along the Y axis.

The fog's density is determined exclusively by the shape of the object to which the material is assigned. The **Distance** value defines the required density of the object so the fog looks volumetric in the color defined. If the object is smaller, the fog will look transparent. Therefore, this property is only suited for simple effects such as fog in ditches and similar depressions.

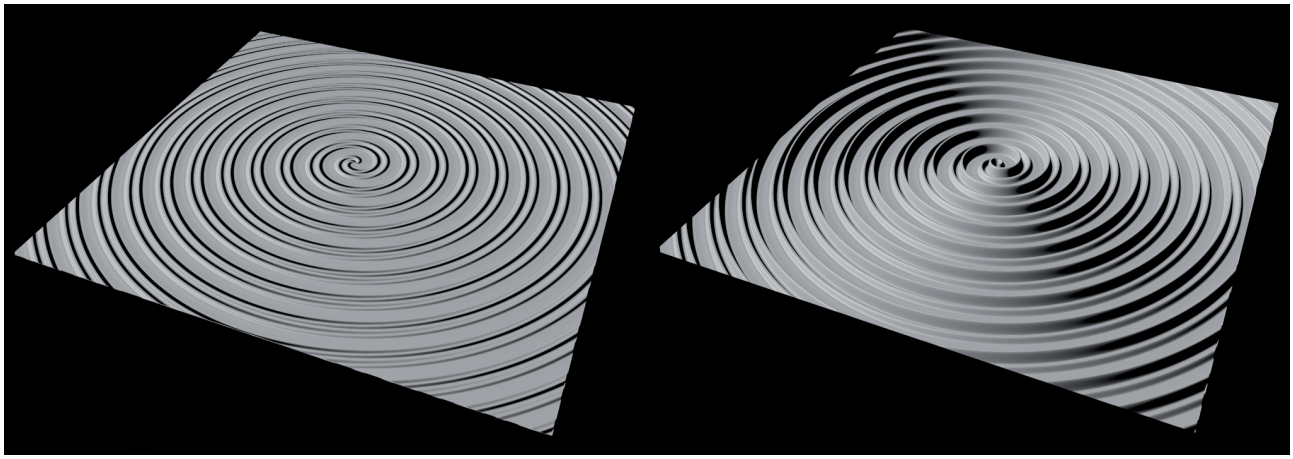
9.4.8 Bump Channel

You already know what this channel does from the shader materials section – where it had a different name: **Roughness**. This channel requires a **Texture** to be defined, whose brightness properties will be used to create the effect.

Bright regions will appear convex and dark regions will appear concave. However, this is only an illusion that is created by manipulating the surface Normals. The object's shape will itself not be modified. This channel should therefore only be used to simulate slight surface irregularities or bumps. The **Strength** value defines the intensity of the effect. Negative values can also be entered, which will invert the effect. If **Sampling** is set to **MIP**, the bump effect will automatically be reduced the farther it is away from the camera, which avoids problems with anti-aliasing. The bump cannot only change the surface shading but also affects the rendering of transparencies and reflections. The latter can be controlled by the **Use Bump** option in the **Reflectance** channel.



This effect is made even more impressive if the **Parallax Offset** value is increased. The bump effect is amplified even though the surface remains unchanged.



The **Parallax Samples** setting defines how precisely the sampling of brightness on the surface will be.

9.4.9 Normal Channel

To use this channel, a texture must be loaded into the **Texture** field. However, a normal image or shader cannot be used. A **Normal Map** has to be used, which contains color-coded information about how the Normals should be oriented on the object's surface. For example, the red regions of a texture can represent Normals in the **X** direction.

These types of textures cannot be created but are generally extracted from an object's surface. There are always two objects necessary to do this – a high-resolution model and a low-resolution object on which the **Normal Map** will be used.

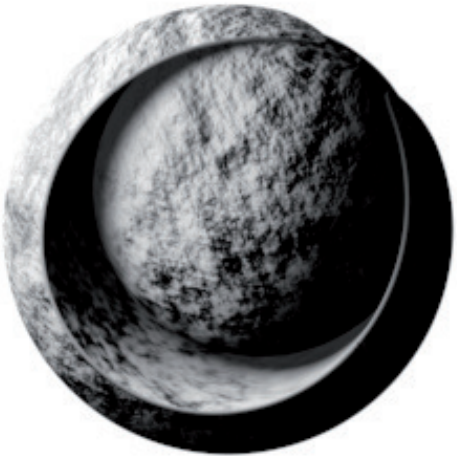
The low-res object will then have the detailed surface of the high-res object without the high polygon count, which makes this channel ideal for game developers, for example, who need to keep the overall number of polygons down in order to achieve an acceptable frame rate.

Various **Methods** are available for evaluating Normal maps. If **Tangent** is selected, the Normals will be oriented according to the angle of the surface polygons. This is the best method to use because Normals will still be oriented according to the Normal map even after an object has been modified. The **Object** and **World** methods use the object and world coordinate systems, respectively, to orient the Normals. Hence, the orientation of the Normals will not be affected by the object's surface.

Depending on the software that was used to create the Normal map, the evaluation of color values and their conversion can vary. The Normal channel's menu includes options that let you flip colors or switch axes.

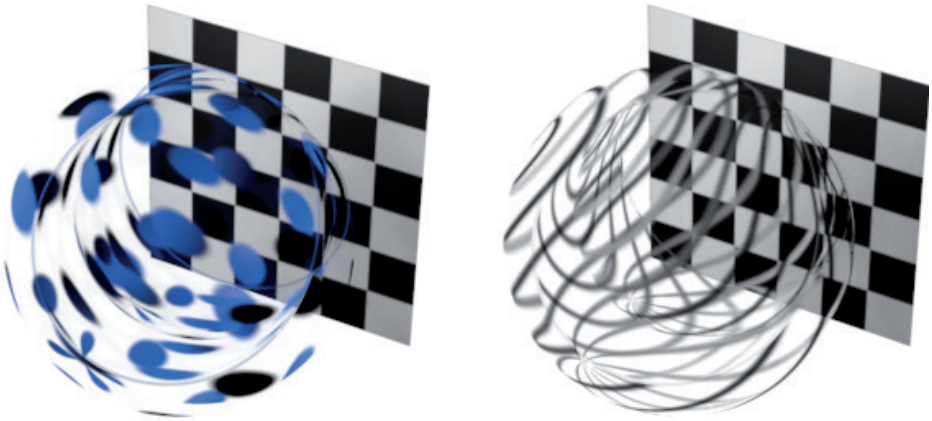
The **Strength** value scales the Normal map's effect. Note that only the shading will be affected as with the corresponding **Bump** channel's setting. The shape of the object itself remains unchanged. However, very flat angles of view will quickly reveal the effect. As with the **Bump** effect, Normal maps are also better suited for depicting finer details such as pores and wrinkles in skin or for fine scratches or decorative elements on objects.

A special **Normalizer** shader is also available that can be used to create a Normal map even from the most simple gray scale images. This shader can, for example, be used to improve the quality of a **Bump** effect if no special Normal map is available.



9.4.10 Alpha Channel

You are probably already familiar with alpha channels, e.g., from working with image editing programs. A gray scale bitmap can be used to make regions of the material partially or completely transparent.



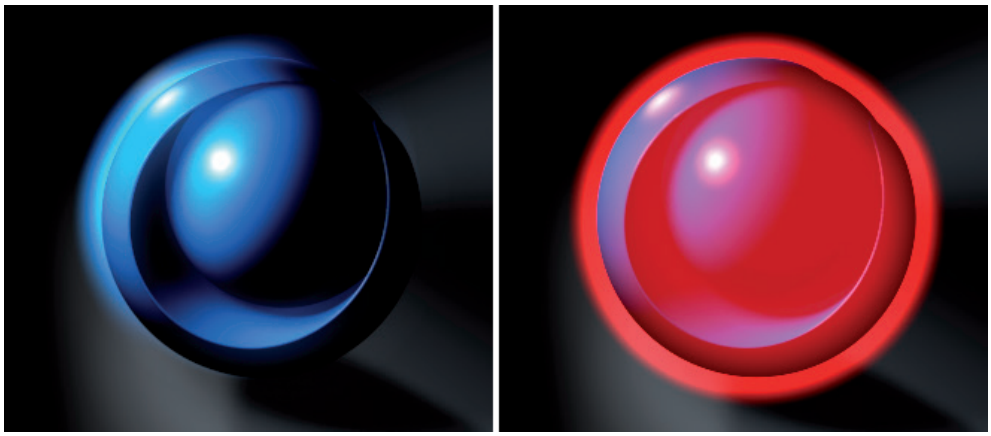
Black regions will be transparent, which differs from the **Transparency** channel's effect where white regions are transparent. The **Alpha** channel also overrides all other channels, whereas the **Transparency** channel can be used in conjunction with specular highlights and reflections.

If the **Soft** option is enabled, the texture's gray scale will be evaluated and its brightness values will be used as a mask for the material. If this option is disabled, the **Color** and **Delta** settings will be made available. The **Color** setting can be used to mask a specific color, which makes it possible to use color images as a texture. The **Delta** color defines the maximum allowable deviation from the mask **Color**. This makes it possible to define even slight deviations from the defined **Color** as masks. However, it is almost impossible to create gradually fading alpha masks using this method. You should therefore use the **Soft** option in conjunction with a gray scale image as a texture.

All images that already contain an alpha channel can be implemented directly. If the Image **Alpha** option is enabled, the image's alpha channel will automatically be used. If **Pre-multiplied** is enabled, the color and alpha values will be multiplied. If the same bitmap is used in the **Color**, **Luminance** and **Alpha** channels, color seams and visible outlines around masked regions can occur. You should therefore first try work with this option disabled. The **Invert** option will invert the texture's effect – white regions will be transparent and black regions will be opaque.

9.4.11 Glow Channel

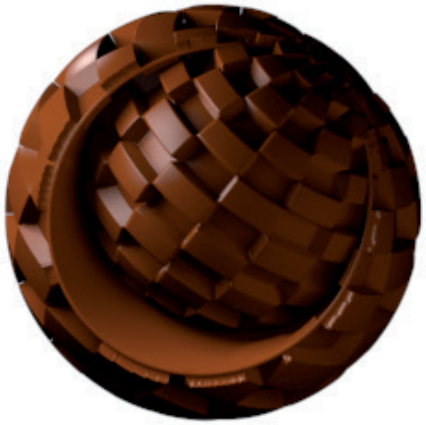
The glow effect is considered a 'special effect' because it adds a glow around the object's surface. This can, for example, be used to simulate very hot or intensely luminous objects. If the **Use Material Color** option is enabled, the effect will assume the shaded surface color. If this option is disabled you can define a custom **Color** with a custom **Brightness**, which will make the effect more independent of the object's lighting.



The **Radius** value defines the effect's spatial expansion from the object's surface. **Inner Strength** and **Outer Strength** can be used to define different levels of intensity near the surface and for the **Radius** distance. The **Random** value defines is used to vary the size of the glow during an animation and the **Frequency** value defines the rate of change, i.e., the speed of variation during the animation. Higher **Frequency** values will speed up the glow effect's flickering accordingly.

9.4.12 Displacement Channel

Whereas the **Bump** and **Normal** channels only simulated the distortion of the surface with shading effects but the **Displacement** channel actually modifies the shape of the object.



As with the **Bump** channel, a **Texture** must be selected here as well whose gray tones or colors are evaluated for the effect. The texture used depends on the **Type** setting that is defined. If set to **Intensity (Centered)**, gray tones with less than 50% white will be interpolated as concave and brighter regions as convex. If **Intensity** is selected, the original surface will remain unchanged in the bitmap's black regions. Brighter regions will result in a corresponding displacement of the surface along the surface Normals. The remaining modes use the **Texture's** color values and not its brightness values. If **Red/Green** is selected, the red regions will be concave and the green will be convex. If **RGB (XYZ Object)** or **RGB (XYZ World)** are selected, the red, green and blue regions will be bound to the object or world coordinate system's orientation, respectively. If **RGB (XYZ Tangents)** is selected, individual polygons and the orientation of their Normals will serve as a reference for the displacement. This is the only **RGB** mode that will displace the object's surface without the effect itself shifting on the surface.

The **Strength** value serves as a multiplier for the displacement. The **Height** value defines the actual scale of the displacement. This value must, therefore, be adapted to the scale of the object itself. Always remember that displacement can only occur in regions in which points are located on the object's surface – the displacement is created by moving these points. For example, this is why the **Render Perfect** option must be disabled if the **Displacement** effect is applied to a **Sphere** object.

To prevent each object whose surface is displaced from having to be subdivided a lot, the **Displacement** channel offers **Sub-Polygon Displacement** settings that will use the material itself to further subdivide the object for rendering. New points will be created on the surface, which produces a better-quality displacement effect – which can be seen only when rendered because the object itself will remain unaffected in the Viewport.

You are already familiar with the **Subdivision Surface** effect. The **Sub-Polygon Displacement** effect works exactly the same if the **Round Geometry** option is enabled. The additional subdivision will be used to round the object's shape. Enabling this option makes it unnecessary to make the object a sub-object of a **Subdivision Surface** object!

The **Subdivision Level** value defines the number of additional subdivisions that should be created. This is the same as the **Subdivision Surface** object's **Subdivision** value. The remaining options affect the type of geometry rounding for displacement. For example, the **Round Contour** value will round off the object's open edges. Otherwise a square surface would, for example, be made circular by the rounding effect.

As you already know from working with the **Subdivision Surface** object, the shape of the original object can be modified greatly by the smoothing effect, which brings up the question, which shape should be used for the material. The **Map Rounded Geometry** option defines which surface coordinates should be used – those of the original or of the rounded object. The **Map Resulting Geometry** option subsequently defines at the point at which the material's textures should be assigned to the surface. If enabled, the textures will be assigned after the surface has been rounded.

If **Keep Original Edges** is enabled, the surface's Phong shading will be modified so that the original object's hard edges are maintained as such on the smoothed object. These types of edges will then be made up of surfaces that lie at angles larger than the Ph**ong Angle** defined in the **Phong** tag or edges for which Phong smoothing was broken. These commands can be found in the **Mesh**/**Normals** menu.

As you already know, the displacement, as a rule, follows the direction of the surface Normals, which can be calculated individually for each polygon or interpolated. A point's Normals are calculated according to the angle of view of the surrounding Normals. Enabling the Best Distribution option will ensure that Normals interpolated in this manner are used, which can also improve the rendering of the displacement at hard object edges.

As a rule, displacement and sub-polygon displacement will only be visible when the scene is rendered. In order to view normal displacement in the Viewport, the **Displacement Deformer** can be used in **Emulation** mode. If you want to make **Sub-Polygon Displacement** visible in the Viewport, enable the **Tessellation** option in the Viewport as well as **Enhanced OpenGL** for Cinema 4D. The graphics card will assume the display of the additional subdivision. The degree of subdivision can be defined in the material's **Editor** menu.

9.4.13 Viewport Settings

These settings have nothing to do with the material's properties or rendering. They only affect the Viewport display. If **Animate Preview** is enabled, an up-to-date preview of the material will always be displayed on objects in the Viewport. This can be helpful if a material contains movies or animated shaders, for example. You can always see the shader's effect in the Viewport by moving the **Timeslider** (directly below the Viewport) to the corresponding frame of animation. The **Texture Preview Size** setting can also be very helpful. This was already discussed briefly in the **Physical Sky** section. The **Texture Preview Size** defines the maximum size with which each **Texture** used in the material will be displayed in the Viewport. This has nothing to do with the quality of the material for rendering. Of course the quality of the texture's display in the Viewport can only be as good as the quality of the **Texture** itself. Of course, the memory required will also increase as the level of quality increases. Therefore, you should only increase the display quality of the **Textures** if you really need the added visual information.

If you use **Enhanced OpenGL** in the Viewport, the **Eye** icons can be used to enable or disable individual material channels. If the **Editor Display** setting is set to **Combined**, the properties of all active channels that are also enabled via the **Eye** icon will be displayed in the Viewport. Otherwise you can select individual channels to be displayed in the Viewport.

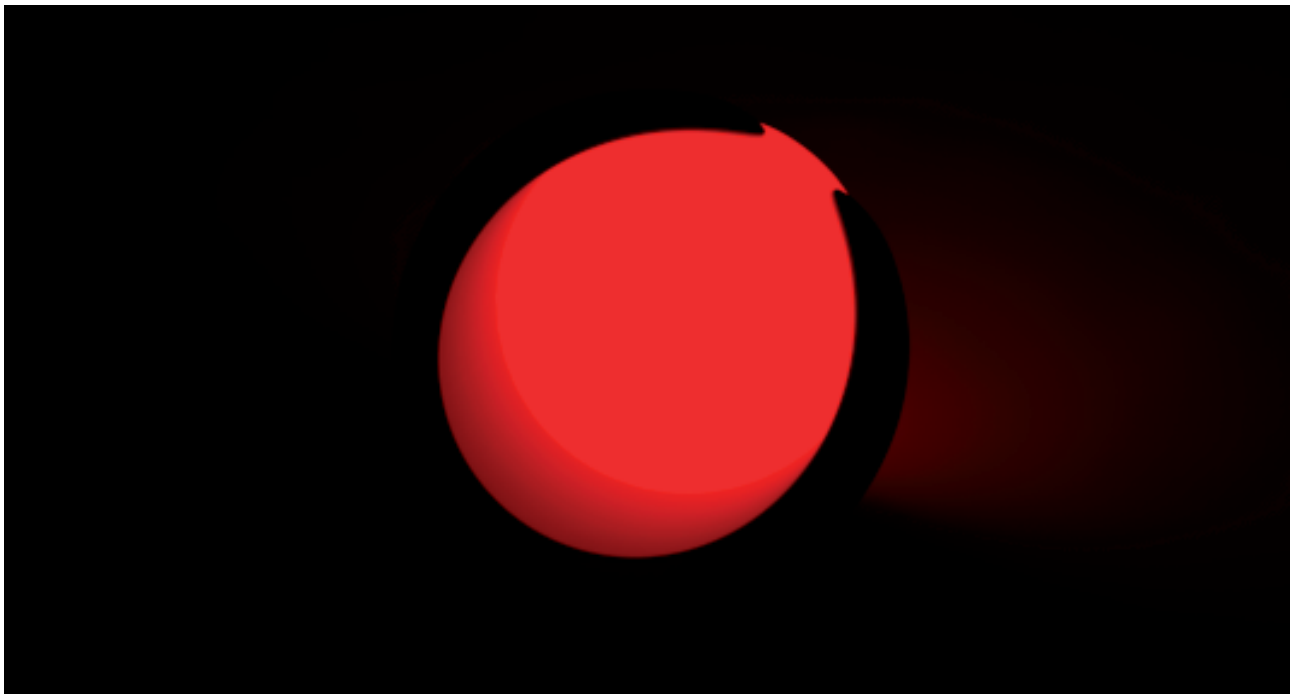
In order to use an individual environment image per material when the **Reflectance** option is enabled for **Enhanced OpenGL**, a new image or shader can be loaded in the **Environment Override** setting. This texture will then be applied to an invisible sky in the scene and can be positioned using the **Rotation** settings. If the **Standard**, **Physical** or **ProRender** are used, the object's actual environment will of course be seen in the reflection.

In the **Viewport Tessellation** menu you can define the method with which OpenGL should control tessellation. If **None** is selected, no additional subdivision will take place even if **Tessellation** is enabled in the Viewport. If **Uniform** is selected, a subdivision will take place independent of the distance from which objects are viewed. The **Projective** option refines the subdivision according to the distance from which the object is viewed. The maximum number of subdivision levels depends on the capabilities of the graphics card being used.

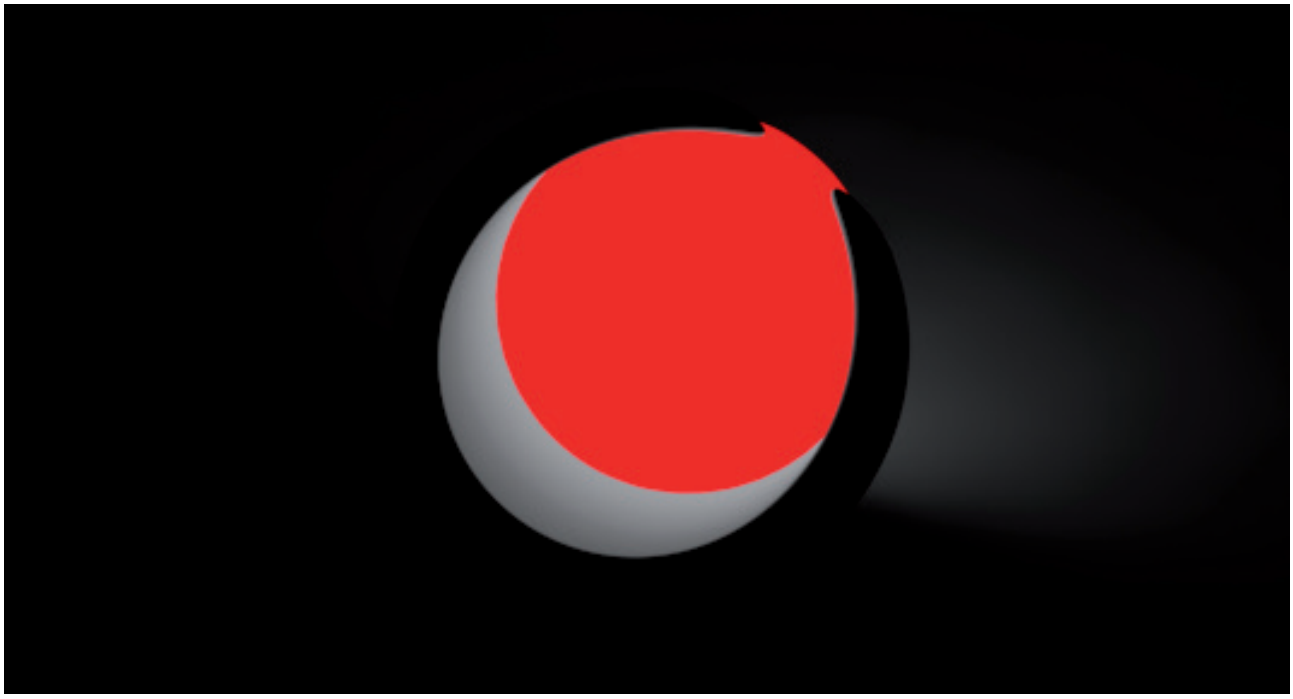
9.4.14 Illumination Settings

This channel primarily affects the material if **Global Illumination** is enabled for rendering. If you do not have **GI** enabled, the settings at the top of the **Illumination** menu will not apply.

Several of these settings will be familiar from the shader section. **Generate GI**, in conjunction with the **Strength** value, defines the brightness of the light reflected by the object's surface. This can be reflected light or light generated by the **Luminance** channel.



The **Saturation** value can be used to modify the color of the light reflected/emitted by the surface. A **Saturation** value of 0% will produce a white light, even if the object surface emits a red light.



The **Receive GI** settings will react accordingly. This is the effect that pure **GI** light has on the material. The direct illumination by normal lights will not be affected by these values.

Whenever you want to use a luminous material in conjunction with **GI** to light your scene, the material's GI sampling mode should be adapted accordingly. This is what the **Polygon Light** option is for. This option is only made available if the material's **Luminance** channel is active and should always be enabled if the object to which this material is assigned is relatively small in your scene, or if you want to improve its sampling for rendering with **GI**. You should be aware of the fact that **Global Illumination** emits numerous render samples into the scene that are used to ascertain the scene's lighting situation. Of course, luminous objects also illuminate the scene and should therefore also be sampled. However, if **Polygon Light** is disabled, this sampling will not be done and this material will be treated as any other material. If this material is applied to a **Sky** object, this option does not have to be enabled because the GI rendering for the sky as a light source can be optimized differently, which we will discuss in the **Render Settings** section.

If your material uses the **Transparency** channel, e.g., to render a pane of glass, the rendering can be improved using GI sampling. For example, when sunlight passes through a window into a room – the glass should let the sun's light and the sky's light pass into the room and not block it out. If **Portal** is enabled, this material will be included in the GI calculation. Note that the coloring and intensity of the transparency effect has an influence on the diffused light that passes through the window.

Regarding the **Illumination** channel's **Caustics** settings, they were already discussed in detail in the shader materials section. **GI** and **Caustics** settings only affect rendering if **Global Illumination** and **Caustics** are also enabled in the Render Settings menu.

9.4.15 Assignment List

This list contains objects to which the material has been assigned. The material will automatically be passed on to sub-objects. Objects can also be dragged into this list from the ***Object Manager***, which will also assign the material to that object. Otherwise you should already be familiar with how the material preview is dragged from the ***Material Manager*** onto an object name in the ***Object Manager*** to assign the material to that object.

SUMMARY: TEXTURE CHANNELS

- Default materials are not optimized for a specific material type and can therefore be used to create any type of material.
- The default material is defined using the available material channels, which control individual material properties.
- The **Color** channel defines the surface color and the shading model, which is then integrated with the shading using the light source or **Global Illumination**.
- The **Diffuse** channel is used to darken the surface. This darkening can be carried over to the **Luminance**, **Specular** or **Reflectance** channels. A special shader (**Ambient Occlusion**) can be used to automatically restrict the darkening effect to concave regions and covered surfaces.
- The **Luminance** channel adds its properties to the surface and is therefore independent of the light sources. This channel can be used to light the scene in conjunction with **Global Illumination**. Otherwise the channel can be used to load special Shaders that can simulate diffused light on objects (**Sub-Surface Scattering**).
- The **Transparency** channel can be used to simulate liquids or glass. Surfaces have their own characteristic refraction values, which can be selected from the drop-down menu.
- The **Transparency** channel can also be used to render reflections.
- The **Absorption** value makes the transparency's color dependent on the thickness of the object.
- The **Blurriness** effect blurs the transparency to simulate a rough surface or material noise.
- The **Reflectance** channel defines the reflective properties of the surface, including highlights and reflections as well as special effects such as anisotropy or simulated weave patterns and cloth. The **Roughness** setting can be used to define the surface's quality.
- When using a physical material setup, the color shading of the surface can be simulated via a diffuse layer in the Reflectance channel.
- The **Environment** channel can be used to simulate reflections using a loaded texture. A loaded landscape image can be used to make an object look as if the landscape is reflected in its surface.
- The **Environment** channel can be combined with the Reflectance channel.
- The **Fog** channel can be used to create very simple fog that can easily be combined with other material channels.
- The **Bump** channel can be used to simulate slight roughness, structure or scratches, as with the **Normal** channel.
- The **Bump** channel only requires a gray scale texture while the **Normal** channel requires RGB color values that are used to define the X, Y and Z direction of the Normals.
- The **Alpha** channel is used to mask regions or reduce their visibility.
- The **Glow** channel can be used to add a glow effect to objects.
- The **Displacement** channel is used to actually modify an object's surface by displacing its points using a bump or normal map.

9.5 Using Shaders and Textures

Shaders in the *Material Manager* menu **Create / Extensions** were already discussed in a previous section. A different variation, however, is the **Channel** shader, which can be used within material channels. These shaders can be used in place of loading bitmaps as textures. This bears the advantage that you don't have to worry as much about texture resolution and the required RAM. Generally speaking, shaders can also be used more flexibly due to their comprehensive range of settings.

Channel shaders can be found everywhere images can be loaded into default materials or shader materials. Next to the **Texture** field you will also find a button with a white triangle on it that will open a sub-menu when clicked upon. This menu including options for loading images. In addition to the usual **Clear**, **Delete** and **Paste** commands, all shaders that are installed will also be available.

Shaders can be used to create a wide variety of effects and can differ greatly in their complexity. Several only produce a simple color pattern and others can, for example, be used to render diffused light as a volume effect. The following explanation of the individual shader settings is designed to give you an overview of what can be done with shaders. After selecting a shader, its settings can be accessed by clicking on its preview image in the material's **Texture** menu. Clicking on the upwards arrow at the very top of the *Material Editor* or *Attribute Manager* will exit the shader settings and return you to the material channel's settings.

Several of the shaders described below require additional textures, e.g., to change their color or to create new patterns. If a shader is opened in which a **Texture** has been loaded in the material channel, this texture will automatically be used when a new shader is added. For example, if you want to load an image in a material's **Color** channel and then determine that the brightness needs to be increased, simply add the **Filter** shader to the **Color** channel. The image loaded into the **Color** channel will not be replaced by the **Filter** shader but will automatically be transferred to it.

Generally speaking all shaders also have a **Basic** tab with corresponding 'basic' settings.

Here you can **Name** the shader and assign it to a **Layer**. The **Blur Offset** and **Blur Scale** settings should already be familiar to you and can be used to blur the shader, if necessary. However, due to the way in which they are mathematically calculated, shaders are not affected by these settings quite the same way as a loaded bitmap would be. The actual **Channel** shader settings can be found in the **Shader** tab's menu.

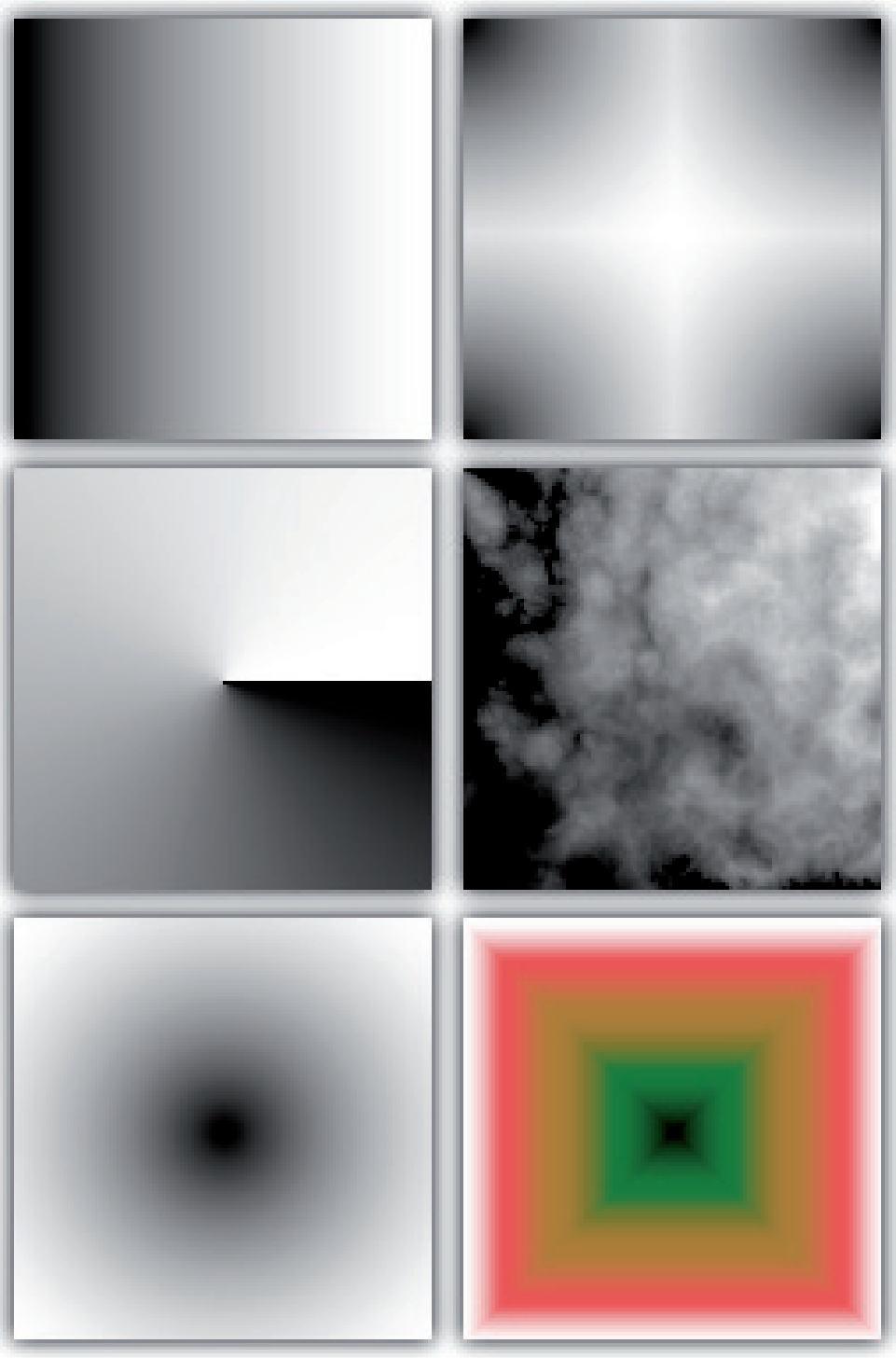
Note that when using **ProRender** for rendering many of the shaders must be baked prior to rendering. The shader will be converted to a bitmap whose size can be defined globally in the **Render Settings** or individual per shader via its **Base** settings. Several shaders, however, cannot be correctly converted to bitmaps, e.g., effects that are dependent on the angle of view onto a surface or the angle of light. Volumetric shaders also cannot be converted to bitmaps. It therefore makes more sense to use the node-based material system when rendering with ProRender since more effects can be used such as the realistic-looking rendering of smoke, clouds and fire.

9.5.1 Color Shader

This is the simplest **Channel** shader. Its only parameter is the **Color** setting, as was the case with the **Color** and **Luminance** channels. Its use is, therefore, quite limited but it can, for example, be used in the **Alpha** channel to reduce the material's overall opacity.

9.5.2 Gradient Shader

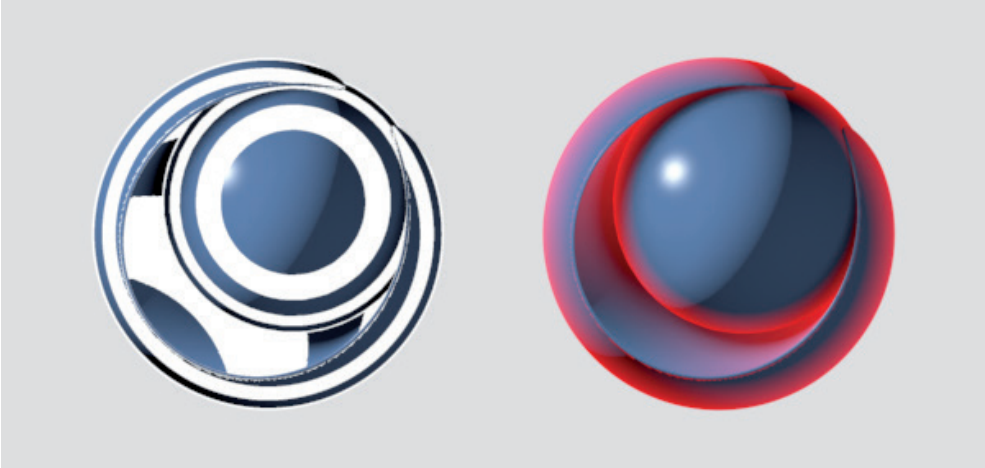
A much more useful feature is the ability to add complex color gradients. The **Type** menu defines the type of gradient that will be used, e.g., **Radial** or diagonal.



The **Angle** value can be used to rotate the gradient. Turbulence can also be added to the gradient. Even three-dimensional gradients can be used that expand from a specific point within an object, for example.

9.5.3 Fresnel Shader

This effect should already be familiar to you from the **Transparency** channel section. However, here you can link any color to the angle between the surface and the angle of view vector.

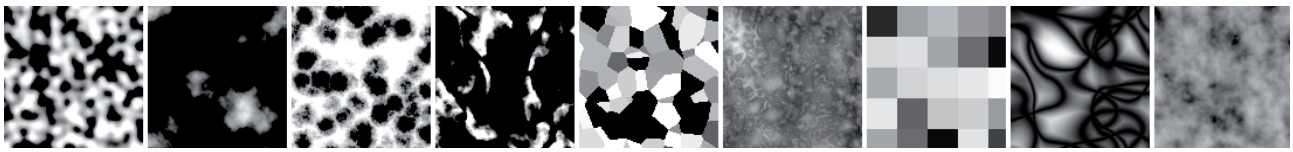


The color at the left end of the gradient will only be assigned to those regions of the surface that are grazed by the angle of view. The color at that right end of the gradient will be assigned to those regions of the object that are viewed frontally. This effect is, for example, useful in the **Luminance** channel when rendering fine cloth or materials. If **Use Bump** is enabled, the **Bump** channel's data will also be evaluated.

Enable the **Physical** option if the **Fresnel** gradient should also behave physically correct. You can then use either a custom refraction index (**IOR**) or select one from the **Preset** menu. This menu contains index values for various materials such as liquids, metals and gems. Note that this function is already integrated in the Reflectance channel. If enabled, it will generally lead to a reduction of reflectivity and will be concentrated more on objects that lie at a distance from the viewer. If Invert is enabled, the shader's effect will be inverted.

9.5.4 Noise Shader

You will probably end up using this shader very often because of its high level of versatility. It can be used so simulate color variations on a surface or for defining **Bump** effects, and much more.

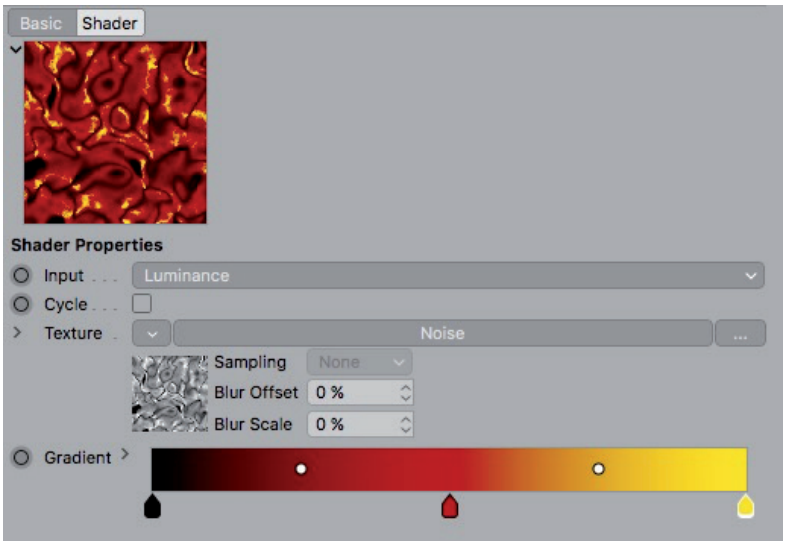


You should already be familiar with its settings from the shader materials section. The various **Noise** types can also be easily selected from an icon gallery, which is opened by clicking on the small gray arrow to the far right of the setting. The **Color** settings at the top of the **Shader Properties** menu define the color of the pattern. Remember that the **Seed** value is used to define the pattern's variation. This can be used to create interesting effects by adding multiple **Noise** shaders, e.g., in a **Layer** shader, with different color combinations. Modifying the pattern's global or relative size can be used to vary distortion and scale. The **Clip**, **Brightness** and **Contrast** sliders at the bottom can also be used to add variation.

The following shaders work somewhat differently because they require a texture to be loaded, which means that textures have to be edited and/or combined.

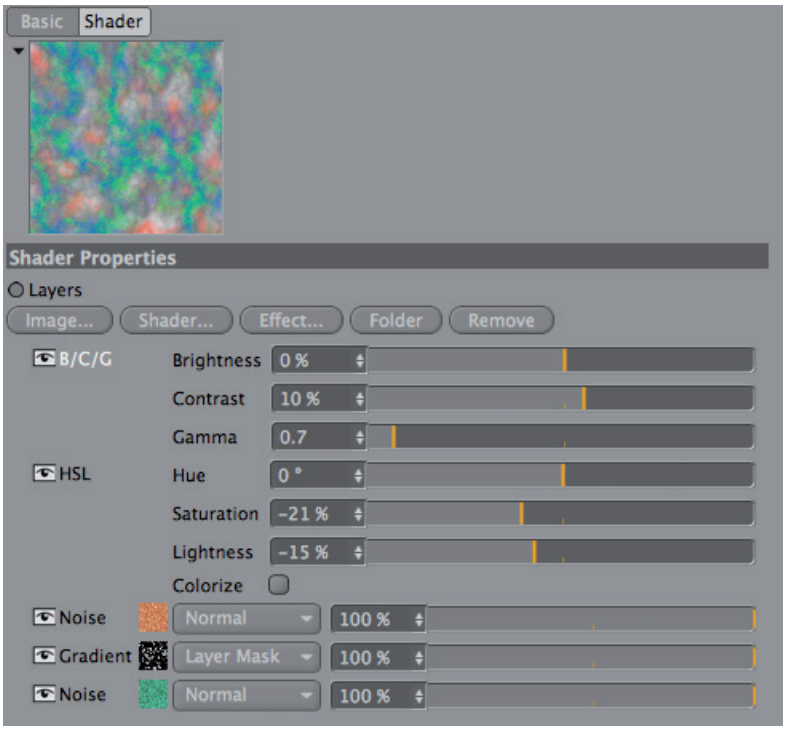
9.5.5 Colorizer Shader

This shader is designed for colorizing gray scale textures. Use the **Input** menu to define the loaded texture's properties that should be replaced by the **Colorizer** shader's **Gradient** colors. The color at the left end of the gradient will as a rule replace the texture's dark pixels – and the color at the right the bright pixels. Since a texture can, of course, also be a shader, you can also load a **Noise** shader, for example, to create a very colorful material.



9.5.6 Layer Shader

At first glance, this shader appears to be quite simple. However, it's actually one of the most powerful shader tools available because it can be used to combine any number of shaders. The principle is the same as with the layers of an image editing program. The buttons at the top of the **Shader Properties** menu can be used to add images or shaders. The **Texture** at the bottom of the list represents the lowest layer. The layers that lie above can be mixed using the **Multiply** or **Overlay** modes. The sliders next to each layer can be used to adjust the strength with which that layer is mixed or its opacity.



Clicking on the **Effect** button will let you choose from various settings for editing a layer's **Brightness/Contrast/Gamma** or **Hue/Sat/Lightness**. The **Posterize** effect can be used to reduce color depth. Its **Levels** value defi-

nes the number of colors. The **Clamp Color** effect can be used to restrict the number of colors used and the **Clip Color** effect can be used to manipulate the brightness values.

Brightness below the **Low** value will be reduced to 0% and brightness in excess of the **High** value will be set to 100% intensity. The **Transform** effect is used to scale, move, mirror or rotate the shader. The **Distort** effect distorts the underlying layer using a **Noise** pattern.

Finally, layers and effects can also be grouped in folders. Click on the **Folder** button to add a folder. Items can be placed into or arranged in folders via drag & drop. Right-click on an item and select **Remove** or click on the **Remove** button to delete an item. Right-clicking on an item will also make additional options available from which you can choose.

9.5.7 Filter Shader

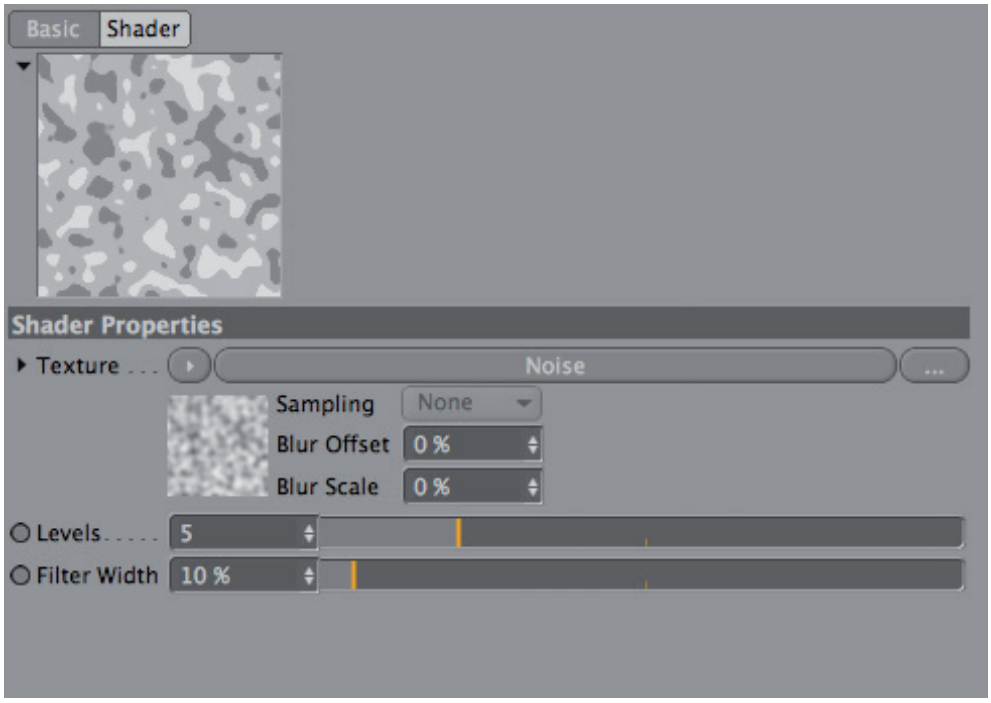
This shader offers various options for affecting a texture's RGB values, brightness, contrast or saturation. Gamma and clipping options are also available. The **Filter** shader supports up to 32-bit color depth and is therefore also suited for editing HDR images. Otherwise, this shader's options should be familiar to you from the **Layer** shader section.

9.5.8 Fusion Shader

This shader works like a watered down version of the **Layer** shader – it lets you mix only two textures. An additional **Mask Channel** can be used to mask the **Blend Channel**. The resulting texture's lowest layer is always represented by the **Base Channel**.

9.5.9 Posterizer Shader

You should already be familiar with how this shader works from the effect layer of the same name in the **Layer** shader section. The number of colors in the **Texture** will be reduced according to the Level settings. The color transitions can be defined using the **Filter Width** value.

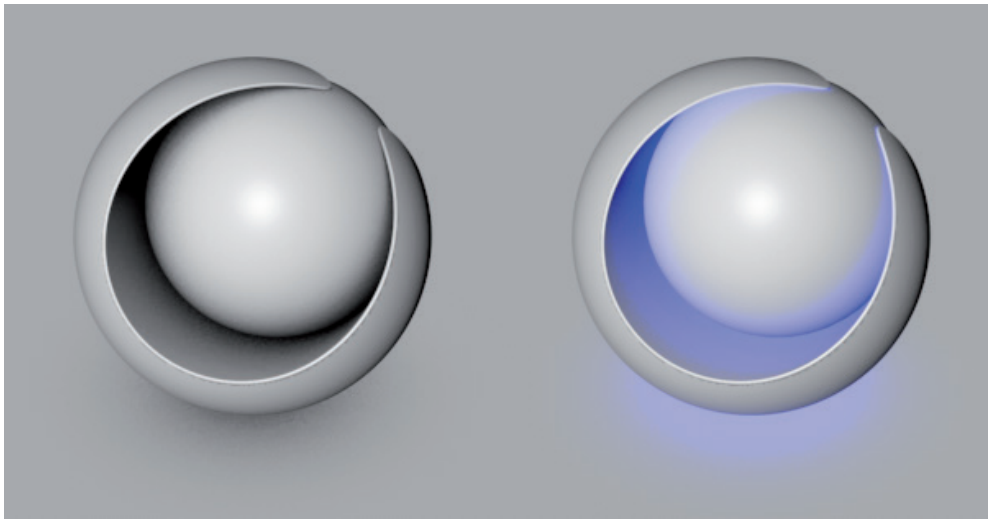


9.5.10 Effects Shaders

This menu contains various shaders that make it possible to create special effects. Several of these can, for example, be used to simulate the dispersion of light within a material. We will take a brief look at the most important of these shaders.

9.5.10.1 Ambient Occlusion

This effect is often abbreviated simply as **AO**. This shader's effect works on the premise that less light reaches regions in the scene that are surrounded by several surfaces. This shader solves this problem by sampling the scene and measuring the distance between surfaces. The closer a surface lies to another, the less light that will be allowed to reach this region. This has nothing to do with the actual illumination of these objects but rather the mathematical probability of light reaching certain regions of the surface. As a rule, this shader is used in a material's **Diffuse** channel to darken the surface at corresponding locations. This primarily creates the fine shadows where objects touch, which would otherwise not be created by normal shadows.

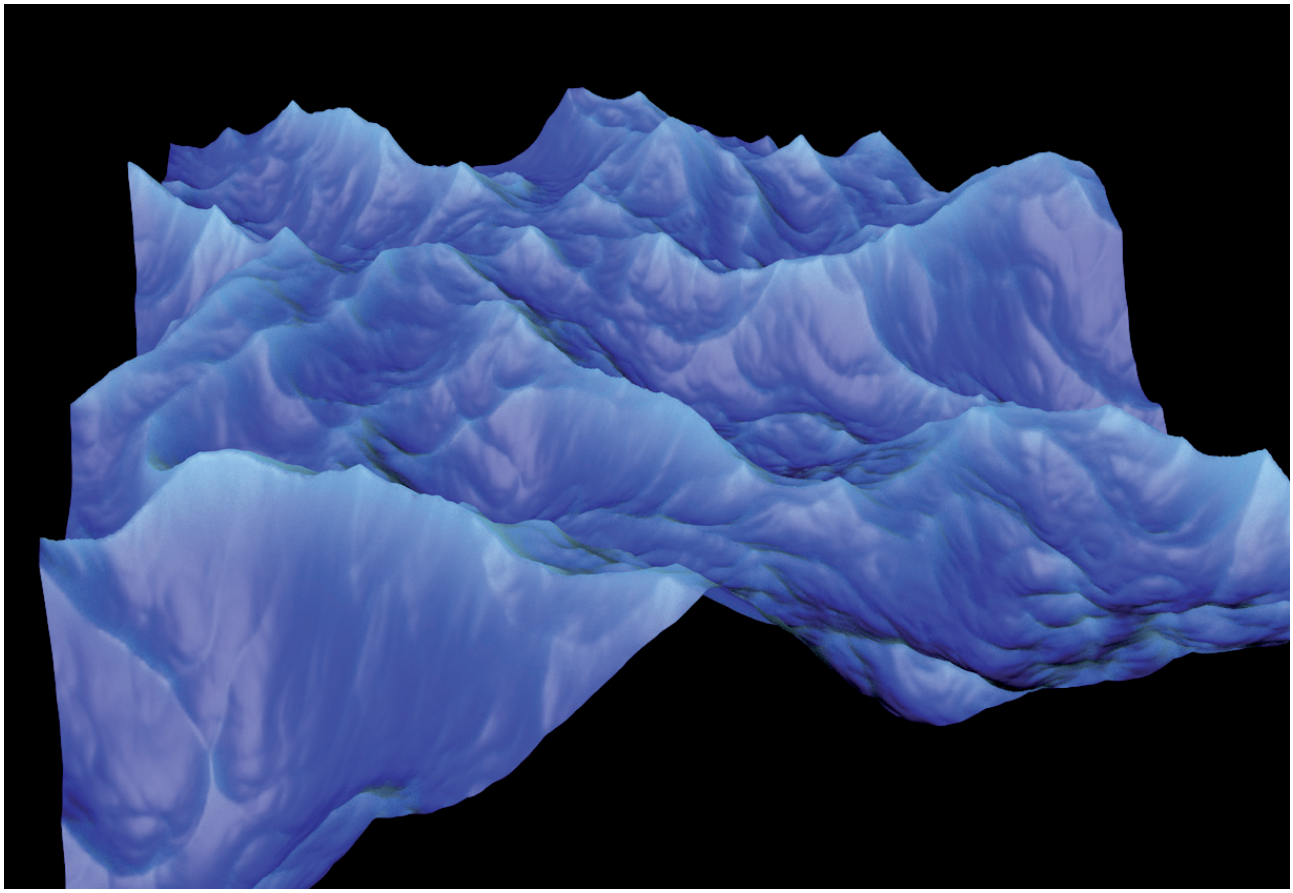


Because this effect is so useful, e.g., for increasing the contrast between objects or shapes, it can be enabled in the **Render Settings** menu, which will apply the effect to all surfaces. If enabled in the **Render Settings** menu, this option must not be enabled separately in each material and the effect can be rendered faster by using **Caches** or rendering it as a **Multi-Pass** layer. The **Ambient Occlusion** effect enabled in the **Render Settings** menu is identical to the **Ambient Occlusion** shader's effect.

The **Minimum** and **Maximum Ray Length** values are used to define the length of the render rays used to measure the distance between surfaces. The **Color** gradient is linked to the result of the distance measurement. If the calculated distance is less than or equal to the **Minimum Ray Length** value, the color value at the left end of the gradient will be output. The color at the gradient's right end will be output at distances in excess of the **Maximum Ray Length** value. The dispersion of render rays is defined using the Dispersion value. If set to 0%, the rays will only be emitted along the surface Normals. Larger values will spread the emission correspondingly in random directions, which will produce a more diffused result. As always when generating render rays, their number is defined using the **Minimum** and **Maximum Samples** values in conjunction with **Accuracy**. Enabling the **Use Sky Environment** option will lengthen the rays until they reach an existing **Sky** object.

The **Sky** object's texture will also be evaluated and multiplied by the **Ambient Occlusion** effect. If this shader is used in the **Luminance** or **Environment** channel, it can also be used to create an illumination similar to **GI** for the object. Enabling the **Evaluate Transparency** option will automatically reduce the **AO** effect if the object on which the rays land is transparent. If **Self Shadowing Only** is enabled, only those rays will be used that land on the surface of the same object from which they were sent.

The Ambient Occlusion calculation can also be inverted by enabling the **Invert Direction** option. The effect will then affect the outer regions instead of the inner regions.



If applied properly, this effect can, for example, be restricted to an object's outer edges by using Ambient Occlusion as a layer mask in a Layer shader. Even simple Subsurface Scattering effects can be simulated by using the **Invert Direction** option for Ambient Occlusion in the **Luminance** channel.

9.5.10.2 Light dispersion Shaders

Several **Effect** shaders use various methods of simulating dispersed light within an object. How much light a material receives and disperses depends on its shape and size. The physically correct rendering of such is therefore quite elaborate and tedious. Often, this degree of precision is not even necessary and a rough approximation of the effect will suffice. This is why there are three different shaders that can be used for this effect. This type of shader is primarily used in the material's Luminance channel, which ensures that the effect will also be visible in shaded surface regions. Note that a realistic preview of this effect is not possible in the ***Material Manager*** or ***Material Editor*** because they are far too dependent on the scene's illumination and the shape of the object. You will have to do test renderings to see what the effect looks like.

9.5.10.2.1 ChanLum Shader (Abbreviation for Channel Luminance)

This shader does not use the complex volume rendering but instead uses a different method of estimating the brightness of the incoming light. Render rays are not sent into an object. Only the area around the surface will be sampled for light. The **Sample Radius** value defines the distance from the surface, starting at the surface's **Seed** distance, at which light will be looked for.

Since light is searched for in a great distance from the object, light can also be found if the surface already lies in a shadowed area. The **Sample Type** setting defines if light should be searched for **Along Normals** or in a random **Area**. The more **Samples**, the longer and more precise rendering will be.

If several light sources should not be evaluated along the way, set the **Light** setting to **Exclude** and drag the lights you want to exclude from the ***Object Manager*** into this list. However, if you want the **ChanLum** shader to evaluate only a couple of lights in the scene you can use the **Include** setting and drag those lights you want to include into the list. This will automatically exclude all other lights.

Note that an empty list is also evaluated. If the list is empty and set to **Exclude**, all lights in the scene will be evaluated by the shader. If set to **Include**, no lights will be included (because the list is empty), which is paramount to disabling the shader.

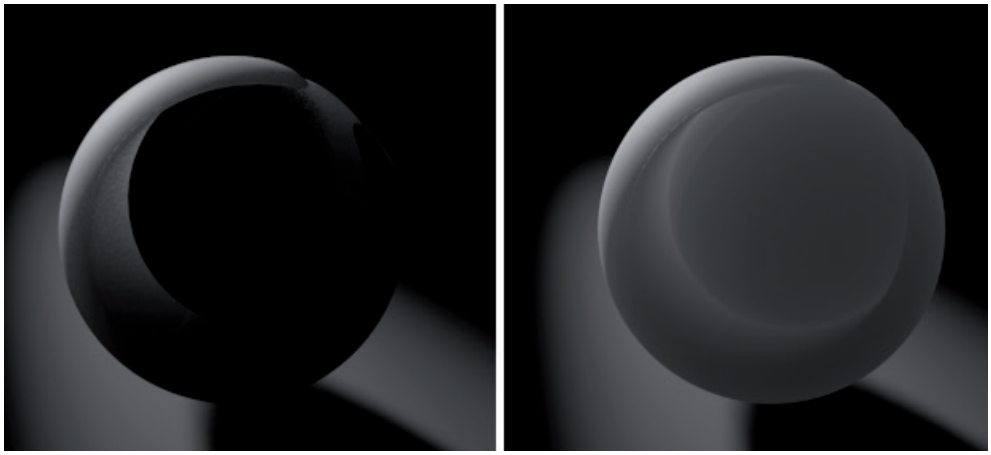
9.5.10.2.2 Backlight Shader

As the name suggests, this shader provides light for the backside of an object. Think of a piece of paper, for example, that is held in front of a light. Some light will pass through the paper and the backside of the paper will also be brightened. A volume rendering would be useless due to the paper being so thin. In such cases, the **Backlight** shader delivers better results in less time.

The **Color** setting is used to color the light that passes through. The **Algorithm** defines the type of surface shading. If set to **Simple**, additional shading will not be added and only the light on the opposite side will be shown. The **Illumination** value defines the intensity of the light that passes through and **Shadow Intensity** defines the intensity of any shadows on the backside of the paper. The **Clip** option is only there for compatibility reasons with older Cinema 4D scenes. It lets the brightness of the light that passes through be restricted to a maximum of 100%. If **Clip** is enabled, the **Contrast** value can be used to define the transition between the shadow and the surfaces through which the light passes. The **Roughness** value is made available if **Algorithm** is set to **Oren-Nayar** and can be used to affect the brightness of the light passing through the surface.

9.5.10.2.3 Subsurface Scattering Shader

This shader renders a physically correct effect that simulates the dispersion and absorption of light by surfaces.



A good place to start fine-tuning this shader is in its **Presets** setting. Here you will find common materials such as milk and human skin as well as ketchup and apples. Of course you can also create custom settings. The **Color** setting defines the object's interior. If you want to render human skin, for example, you would use red tones since blood and tissue take on this color when skin is illuminated. You are surely familiar with the effect that occurs when a strong light is pressed against a hand – the light appears to shine red where the hand is the thinnest, e.g., at the fingertips. The **Strength** value is a simple multiplier for the intensity of the light dispersed in the material and can be manually set to values in excess of 100%. A texture or shader to represent the object's core can also be added to supplement the **Color**. A slight blur effect can also be added to the texture.

The **Path Length** value is the actual measure of the depth to which light can penetrate an object. The larger this value is, the more transparent the object will look. The right value depends on the object's dimensions to which the material is assigned. Note that smaller **Path Length** values take longer to render. Therefore, you should also check to see if you can achieve the same results with shorter render times using the **ChanLum** shader. The **Path Length** can be defined separately for the light's **Red**, **Green** and **Blue** portions by expanding the **Path Length** setting using the small black arrow to its left. This can, for example, be used to increase the amount of red for light that shines through skin. The percent values are in reference to the **Path Length** value. The actual render precision is defined using the settings in the shader's tabs. There are various methods from which to choose. First, let's take a look at the **Multiple** method, which randomly disperses the light. The result is very soft and can often be compared to wax. The effect is achieved by emitting render rays. If **Enabled** is checked, two **Modes** will be made available: **Cache** and **Direct**. If **Cache** is selected, an adaptive pre-calculation of the object will be made whereby measuring points will be distributed on the object. The results of the measurements made will then be interpolated. This process is very fast and well suited for longer **Path Lengths**. However, due of the limited number of measuring points and the large gaps between them, details can end up being lost. The **Direct** mode should be used for short **Path Lengths** and if more detail is required. If this mode is used, all points on the surface will be sampled intensively. An estimation of measuring points will not be made and some of the settings will be removed. What's left will be the **Minimum Threshold** setting, which is used to define the minimum distance for the **Path Length**. Each measured distance must be at least as long as the **Minimum Threshold** value. This avoids extreme brightness at sharp edges or very thin regions of the object from being created. If the light's **Red**, **Green** and **Blue** portions have different **Path Lengths**, the **Separate Color Channels** option can be enabled for more precise rendering in conjunction with the **Minimum Threshold** value, which will also affect the three Color channels.

The render precision in **Direct** mode is defined via the **Sampling Subdivision** value. The higher the value the more precise and slower the rendering will be. If the Physical **Renderer** is used, this value will be applied globally via the **Render Settings** menu. Enable the **Custom Sampling** option only if you want to define an individual sampling for the **Subsurface Scattering** shader, which will also make the **Sampling Subdivision** setting available. The **Direct** mode is often faster than the **Cache** mode described below when using short **Path Lengths.**

Sample Density is a unit of measure for the number of measuring points in **Cache** mode. The higher the value the longer the rendering will take but the more precise the effect will be, especially for finer details. As a rule, increasing the

value beyond 100% will rarely, if ever, be necessary. Instead, try to use smaller values to save render time. The samples made with then be offset against each other and smoothed using gradients. The **Smoothing** value should only be increased if spotting or gaps occur. Otherwise, **Smoothing** will often result in a less detailed effect. The **Threshold** value defines the termination point for the shader. Smaller values will result in more precise but longer renderings; larger values will speed up rendering but can also produce faulty results. The preset value of 0.1 should be a good compromise between quality and render time in most cases. Finally, the **Fast Evaluation** option will activate an alternative algorithm that is optimized for low numbers of samples. Hence, if you are working with a **Sample Density** of less than 100%, enabling this option can help improve render quality or shorten render time.

The **Single** tab's settings can be used either to augment or in place of the **Multiple** tab's settings. If enabled, this effect's settings can also be used to simulate penetrating light but will not disperse it as broadly as the **Multiple** algorithm. When the light rays pass through a surface they will primarily run in the same direction as when they hit the surface, which means that this effect will not produce a waxy look. This effect also uses samples whose number defines the quality. This value is defined using the **Sampling Subdivision** value. If the **Physical Renderer** is used, this value will be defined globally in the **Render Settings** and not by the shader. The shader's individual **Sampling Subdivision** value will only be used if the shader's **Custom Sampling** option is enabled. As previously discussed, special functions are offered when rendering with ProRender via its node-based system, which also makes several Subsurface Scattering calculations available.

The **Phase Function** value defines the direction in which the light should be diverted within the object. Negative values will reflect the light rays correspondingly more back towards the light source; positive values will divert the light through the object in the original direction of the light in accordance with the value defined. Note that the values -1 and +1, respectively, will deactivate the **Simple** effect completely. A **Phase Function** value of 0 will produce an even dispersion in all directions but without achieving the randomness and softness of the **Multiple** effect. If objects are located between the light source and the object to which the **Subsurface Scattering** shader is assigned, the **Trace Shadow Rays** option will ensure that these cast shadows will also be included in the effect. As with the Multiple effect, the Separate Color Channels option ensures that differing Path Lengths for Red, Green and Blue color portions will be taken into consideration. Both **Subsurface Scattering** effects are augmented by the **Advanced** tab's settings.

The term **Fresnel** should already be familiar to you. Here, the **Fresnel Reflectivity** slider can be used to reduce the light dispersion in regions that curve away from the angle of view. The contrast is increased in these regions, as is the case with almost transparent materials, for example. Otherwise, this slider can remain set to 0%. The **Dithering** slider can be used to add a fine noise to brightness and color gradients. This can help reduce "banding" – the visible color differences in gradients.

The **Index of Refraction** was already discussed in the **Transparency** channel section, for example. Here it affects the object's interior. Many translucent objects are largely made up of water, which is why the default value is set to 1.3. If you want to simulate alabaster or marble, for example, a larger value can be used. You already know where you can get refraction tables with corresponding material values.

The shader's Light tab settings can be used to **Include** or **Exclude** individual lights. The diffuse light rays will only be included if **Compute GI Contribution** is enabled – in conjunction with **Global Illumination** being enabled in the **Render Settings**.

9.5.10.3 Distorter Shader

This helper shader requires two textures. The base texture is loaded into the **Texture** field and an image – or better a gray scale shader, e.g., a **Noise** shader – is loaded into the **Distorter** field. The **Distorter** shader's brightness is used to create the **Texture's** turbulence. We already discussed this type of effect in the **Layer** shader section. The **Type** setting defines the type and direction of the turbulence. **Amount**, **X**, **Y** and **Z** values define the direction and intensity of the distortion. For example, of two **Noise** shaders are combined, adjusting these values can be used to create a completely new pattern.

If the shader is used in the Bump channel, the **Delta** value can be used to define the precision of sampling. Smaller **Delta** values will increase precision but will also reduce the intensity of the bump effect accordingly.

9.5.10.4 Thin Film Shader

Some surfaces are covered by a very thin oily or fatty film. A good example of this would be olive oil on a plate or motor oil on the surface of a puddle of water. There are also materials that consist entirely of soapy or oily substances such as soap bubbles or cooking oil. Light that hits these surfaces is split into the different spectral colors, which results in a colorful shimmering on the surface, as with nacre (mother of pearl).

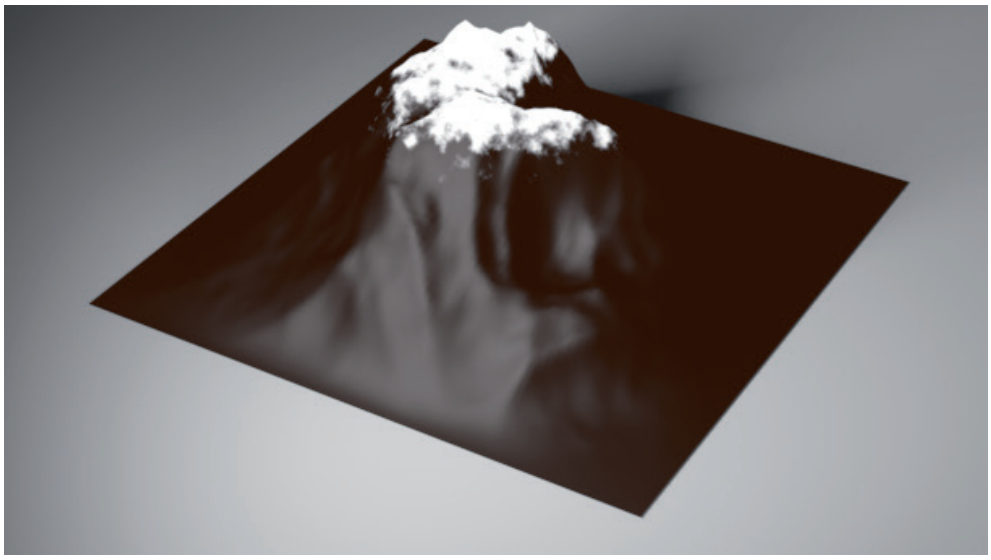
The refraction index and thickness of the film both affect how this shimmer looks. This effect is what the **Thin Film** shader simulates. Select the substance you want to simulate from the shader's **IOR Preset** menu or define a desired **IOR** value. The **Spectral Samples** value is used to adjust the quality of the color refraction. Higher values produce more precise results and also result in correspondingly longer render times. The **Thickness (nm)** value defines this layer's thickness. To add more variation to the spectral colors, a **Texture** can be loaded, e.g., a Noise shader, and the **Variation (nm)** value adjusted accordingly. The loaded texture's grayscale values are multiplied by the Variation (nm) setting to simulate a variation in the film's thickness. Values greater than 50% gray will amplify the effect; values less than 50% gray will diminish the effect correspondingly. A **Color** value can also be defined with which the spectral colors can be multiplied. This color should, however, be set to white to achieve realistic-looking results. Another **Texture** can be loaded at the top of the **Shader Properties** menu used in conjunction with the **Mix Strength** setting to add even more variation. The Thin Film shader is most effective when used to color reflections in a material's **Reflectance** channel.

9.5.10.5 Falloff Shader

This shader is very similar to the **Gradient** shader but makes it easier to orient the gradient in a specific direction using the **Direction** values and **Space** setting. A **Direction** of 0, 1, 0 will produce a vertical gradient, which can, for example, be used to color a **Landscape** object so its mountain tops are white and its sides and valleys are green.

9.5.10.6 Terrain Mask Shader

This shader is similar to the **Falloff** shader, e.g., because it also lets you define a gradient at a specific position in space. However, this shader's settings are more comprehensive and also let you include the dependency on surface shading. This shader is designed to be used in the **Alpha** channel, for example, or as a layer mask in the **Layer** shader for transitioning between different textures.



An interesting application can be for texturing a landscape to automatically assign a rocky texture to mountains and grass to lower lying regions.

This shader uses two types of calculations that can be activated separately but also used together. The **Enable Altitude Masking** method outputs the **Gradient** colors at the right end between the **Min Altitude** and **Max Altitude** values. The **Gradient** colors at the left end will be assigned to the remaining regions. This is all done according to the object's coordinate system of the object to which the material is assigned. Alternatively, the **Use Global Coordinates** option can be enabled, which will cause the world coordinate system to be used. Use the **Soften Min** and **Soften Max** values to create a softer transition between colors. The percentage values define the width of the transition. The **Noise Height** value can also be used to add irregularity to the gradient's edges. The **Scale** value defines the density of detail for this effect. Smaller values will produce correspondingly finer variations. As its name states, the second method, **Enable Slope Masking**, evaluates the surface angles. All angles that lie between the **Min Slope** and **Max Slope** values will be assigned the color at the left end of the **Gradient**. All other regions will be assigned the color at the right end of the **Gradient**. The slope is calculated relative to the **Direction** or the **Custom Direction** defined, which is defines as a vector in the object's coordinate system. Here, you can also use the world coordinate system by enabling the **Use Global Coordinates** system. **Soften Min** and **Soften Max** work the same way as for the altitude masking effect. An existing **Bump** channel can also be used to calculate the slope. To do so, enable the **Use Bump** option. If **Altitude Masking** and **Slope Masking** are used together, the shader will multiply both results together. The **Slope Masking** will then only be visible at regions in which the **Altitude Masking** calculates a **Gradient** brightness in excess of 0%. Enabling **Alternative Blending** will let you sharpen the gradient's edges. A **Hardness** value of 0% will produce no effect. However, the higher the value is set the softer the gradient's edges will be. The shader's effect can also be inverted by enabling the **Invert** option.

9.5.10.7 Lens Distortion Shader

Lens distortions are distortions created by the camera's lens. The degree of distortion depends in part on the focal length and the quality of the lens itself. The **Lens Distortion** option lets you measure and define the distortion of a specific lens. A photo can be loaded over which lines are positioned and then manually or automatically un-distorted so that all elements that are supposed to be straight do in fact appear straight. This information can then be saved as a custom lens distortion and subsequently loaded into this shaders. Similar options can be found in the **Camera Calibrator** tag, in the **Lens Distortion** Post Effect and in the **Motion Tracker**. This shader, in conjunction with the right lens distortion profile, can be used to un-distort a loaded image that can, for example, be applied to a **Background** object.

9.5.10.8 Lumas Shader

This shader offers almost all the settings of a normal material. If you read the description of the **Danel** shader carefully you will find numerous similarities. The **Lumas** shader also makes it possible to create up to three separate specular and anisotropic effects, which also lets you use these functions in default materials.

Note that the Reflectance channel can also render anisotropic effects. Since many of this shader's properties simulate lighting effects, it is well suited for use in the **Luminance** Channel. The **Color** channel can even be disabled in some cases. Using this shader in other channels can also produce interesting results, e.g., if you only use the **Lumas** shader's settings and multiply these using other textures. This can be done quite easily using the **Fusion** or **Layer** shader, which lets you restrict material effects to an object's illuminated regions.

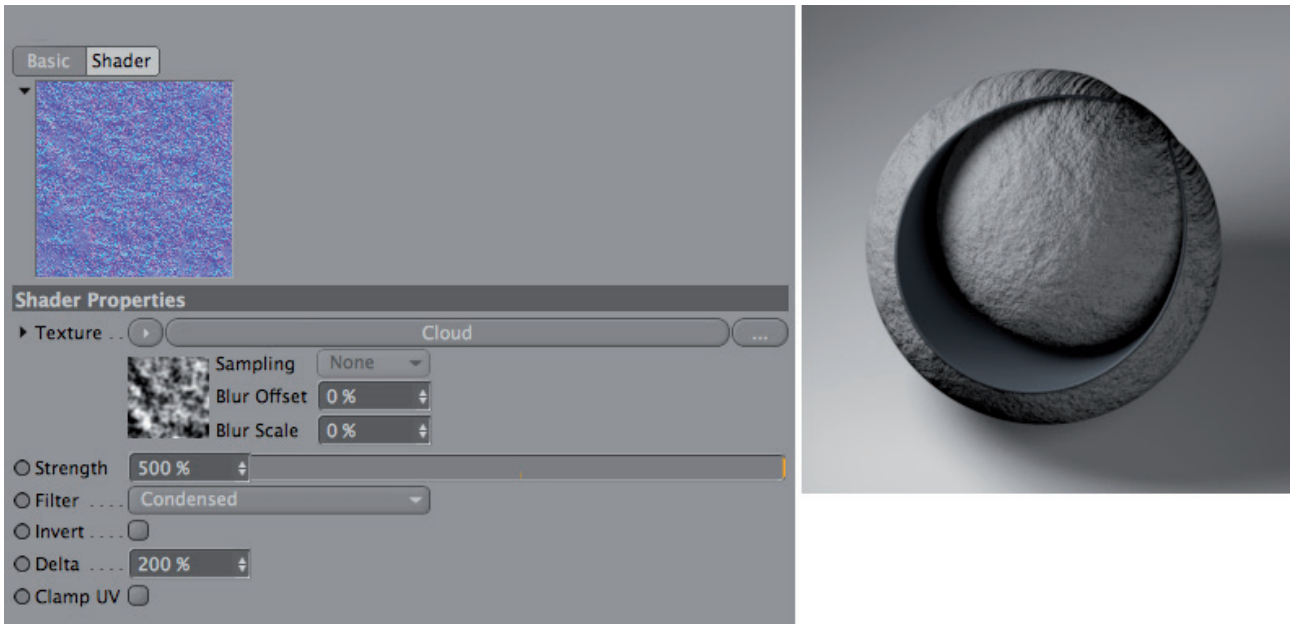
9.5.10.9 Normal Direction Shader

As you already know, each polygon has a Normal, i.e., a vector that stands perpendicular to the polygon's front surface. This Normal is, for example, used to calculate the surface's shading, serves as a directional indicator for polygon tools, or serves simply to differentiate a polygon's front or backside. This shader makes it possible to ascertain the position of all Normals of the object to which this material is assigned. Surfaces whose Normals point in the direction of the camera are assigned **Color 1** and those pointing away from the camera will be assigned **Color 2**.

This makes it possible to create many types of interesting effects, for example, if the shader is used in the Transparency or Alpha channels. Surfaces can be made to automatically change their visibility, depending on the direction from which they are viewed.

9.5.10.10 Normalizer Shader

We have already discussed the differences between **Bump** and **Normal** maps. Normal maps can be also be used to define the angle of a surface for shading using RGB color values. However, RGB textures are required, which cannot consist of just any image. The **Normalizer** shader helps with this calculation by using normal gray scale images, which are normally used in the **Bump** channel, as textures.



The **Strength** value scales the calculation, i.e., makes it more – or less – intense. The Filter setting can also be used to increase contrast. The **Condensed** mode is better suited for fine details and the **Sobel** modes are better suited for wide-ranging or rough patterns.

Enabling the **Invert** option will invert the loaded texture's brightness values before the RGB values are converted.

The **Delta** value has the same function as in other shaders and intensifies the effect correspondingly with increasing values. Enabling the **Clamp UV** option will help avoid the texture's edges from being affected, e.g., pixels from the texture's left edge appearing as a color seam on the opposite side, which can occasionally occur.

9.5.10.11 Pixel Shader

This shader also requires a separate texture. The loaded texture is divided into square sections and the color and brightness of each sector are interpolated to generate the **Pixel** shader's effect. The number of sectors is defined using the **Tiles U** and **Tiles V** values for the **X** and **Y** directions, respectively. The more tiles you use, the more similar the final effect will be to the loaded texture. Smaller values on the other hand will produce correspondingly pixelated results. This can, of course, also be used for animation, for example, if a high-res texture is pixelated in the course of an animation. In such cases, the **Smooth** option should be enabled, which will cause only whole values to automatically be used for the Tile values, even if the animation of these values causes decimal places to be generated.

9.5.10.12 Projector Shader

As we already discussed in the **Nukei** material's **Fusion** section, materials can be projected onto surfaces in a variety of ways. What if the material is applied spherically and one of the material's channels should be projected with a **Flat** projection? This is what the **Projector** shader is for. Load the texture with the deviating projection type into this shader and use the **Projection**, **Offset**, **Length** and **Tile** settings to position the texture correctly. As with the **Nukei** material's **Fusion** effect, the settings of a **Texture** tag selected in the *Object Manager* can also be copied using the **Copy** tag command and pasted by clicking on the **Projector** shader's **Paste Tag** button.

You can, for example, first use a normal material and position it on the surface using its **Texture** tag's options. This material can then be modified and the **Projector** shader can be used in the material channel that you want to use to project the material. Use the **Projector** shader's **Paste Tag** button to copy the **Texture** tag's settings. The Texture tag's projection can then be changed without affecting the **Projector** function.

9.5.10.13 Proximal Shader

This shader is somewhat similar to the **Ambient Occlusion** shader because it also measures the distance between surfaces, points or edges. The polygon objects must, however, be manually dragged from the *Object Manager* into the shader list. Particles can also be used. To do so, drag the **Emitter** or the **Thinking Particles'** particle geometry object into the list.

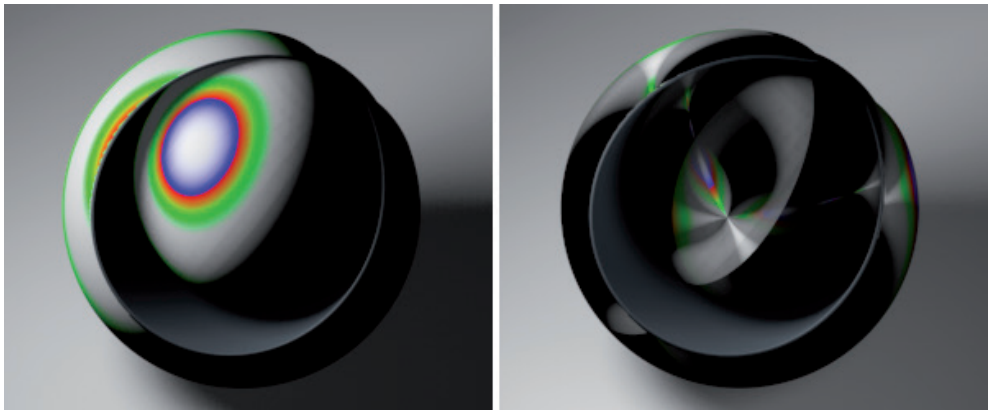
The following options can be used to define how the distances between elements should be measured. **Use Vertices** calculates the distance between the points of the objects in the list and the object to which the material is assigned.

Note that parametric objects cannot be used. Only polygon objects or particles can be used. Enabling the **Polygon Radius** option will sample a region around the center of the polygon for collision in accordance to the polygon's size. This can help improve the result if the points on the object surfaces are not dispersed evenly. Enabling **Use Edges** will measure the distance between the edges and the object. If all options are disabled, the distance between the objects' coordinate systems will be measured.

The measured distances are interpolated to gray scales with the help of the **Start** and **End Distance** values. The **Start Distance** value defines the distance between points, edges or polygons at which a maximum brightness will be produced. Elements that lie farther away than the defined **End Distance** value will have no effect – the shader will be black. If **Use Vertices** or **Use Edges** is enabled, or all options are disabled, an **End Distance** value of 100% will correspond to a distance of 100 units of measure, i.e., 100 cm or 100 m, depending on what's defined for Cinema 4D. If **Polygon Radius** is enabled, the **End Distance** value will orient itself to the size of the respective polygon. This function affects the distribution of the gray scale values that lie between the **Start** and **End Distance** values. The **Blend Mode** defines the type of blend between the gray scale value and the material channel in which the shader is loaded. Interesting effects can, for example, be created by using this shader in the **Displacement** channel: deformations will be caused by approaching objects.

9.5.10.14 Spectral Shader

This shader can be used to color specular highlights with a custom **Spectrum** gradient and distort them radially. This effect is similar to the coloring that can often be seen on the bottom of a DVD, for example.



The **Intensity** value defines the effect's brightness and **Variation** defines the number of times the **Spectrum** gradient will be repeated. The **Out of Range Type** setting defines the order in which the colors will be repeated. If set to **Stop**, only the last color value will be maintained and the entire gradient will not be repeated. If set to **Mirror**, the gradient's repetition will take place in mirrored order, and the **Tiling** mode will repeat the gradient in its original color sequence.

Enable the **Use CD Effect** option to simulate the color gradient on the bottom side of a DVD. The **Front Side** setting defines the object coordinate system's layer on which the front side of the DVD lies. **Width** defines the width of the stripes, **W Factor** defines the starting point of the spectral specular highlights. If set to 1, the stripes will start at the center of the DVD. Larger values will move the starting point away from the center accordingly.

The **Peak** value defines the degree with which the **Spectrum** gradient will be applied to the specular highlights. Larger values will generate a large color spectrum even for weak specular highlights. **Diffuse Intensity** can be used to make the specular highlights more intense. **Diffuse Variation** restricts the specular highlights peripherally, which will concentrate the spectral effect more towards the center of the specular highlight. This shader can be used in either the **Color** or **Luminance** channel. It can also be used in the **Specular** channel if the specular highlight is not large and intense enough.

9.5.10.15 Variotion Shader

This shader can be used to randomly color objects with a material assigned to them – or even randomly color an object's polygons. This is particularly useful when adding color variations to foliage, grass or stones, for example, because only a single material is used.

You must first determine whether **Object Variation** or **Polygon Variation** will be used, or both. If **Object Variation** is used, all objects to which this material is applied will be colored randomly, for example. If **Polygon Variation** is used, all or a defined number of coherent polygons will be varied in color (**Polygon Step** value). The shader's default color is white but can also consist of a loaded **Base Texture**. This can be varied using the **Contrast, Gamma, Invert, Hue, Saturation** or **Lightness values**. In addition, a **Secondary Texture** can be loaded which will randomly replace the **Base Texture** (e.g., a green leaf as a Base Texture and a wilted leaf as a Secondary Texture).

If you want to add even more variation, e.g., to simulate various leaf textures on a tree using a single material, an entire folder full of textures can be applied using the **Add From Folder** function. The shader will then randomly select textures from the folder in either a uniform relation or based on a percentage calculation.

The degree of randomness can be defined using the **Random Color** and **Random Color Mode** settings. A color gradient with its own colors can be used in addition or in place of this whose colors will be randomly applied in accordance with the **Gradient Blend** and **Gradient Mode** settings. The **Probability** setting defines the overall frequency of the variation. A texture can also be loaded as a **Global Mask** whose brightness can be used to mask those areas in which the variations take place.

9.5.10.16 Spline Shader

This shader can, for example, be used in conjunction with **Spline** curves to make text visible on a texture. If you want to use text, enable the **Text Spline** option and enter the text in the **Text** field.

You can also select a font from the **Font** menu below the **Text** field. One setting that you won't find here is **Kerning**. If you need this setting you should use a normal **Text** spline instead. To do so, disable this shader's **Text Spline** option and drag the spline object you want to use from the *Object Manager* into the **Spline** field. You can use either spline Primitives or manually created splines.

Note, however, that the **Spline** object's size will not be carried over 1:1 onto the surface. This is a shader that is assigned via the material's projection type and can therefore appear distorted on the object's surface.

The default size of a material tile is 100 units in the X and Y directions. If your Spline object was created with this in mind, the **X Scale** and **Y Scale** values can remain at 100%. Otherwise you will have to adjust these values accordingly. When doing so, it's important to select the correct **Plane**. This is the plane in the spline's coordinate system on which the curve lies. Therefore, the curves used should be as flat as possible and not, for example, helix-shaped.

The spline curve can be moved within the material preview using the **X Offset** and **Y Offset** values. A **Rectangle** spline with an edge length of 100 units can, for example, can frame the material preview if each **Offset** value is set to 50% and the **Scale** values are each set to 100%. If **Single Pixel** is disabled, the **Line Width** value can be used to define the spline curve's thickness in the material. **Smooth Width** will blur the sides of the lines accordingly. If **Single Pixel** is enabled, a thin line will be used to trace the spline curve. The thickness of the spline shape will be created using dashes that are strung together. The ends of these lines can be rounded using the **Caps** option. Otherwise the lines will end with a hard cut. If you want to use the shader in the **Bump** channel, the **Bump Width** value can be used to widen the transition between the spline shape and the **Background Texture**.

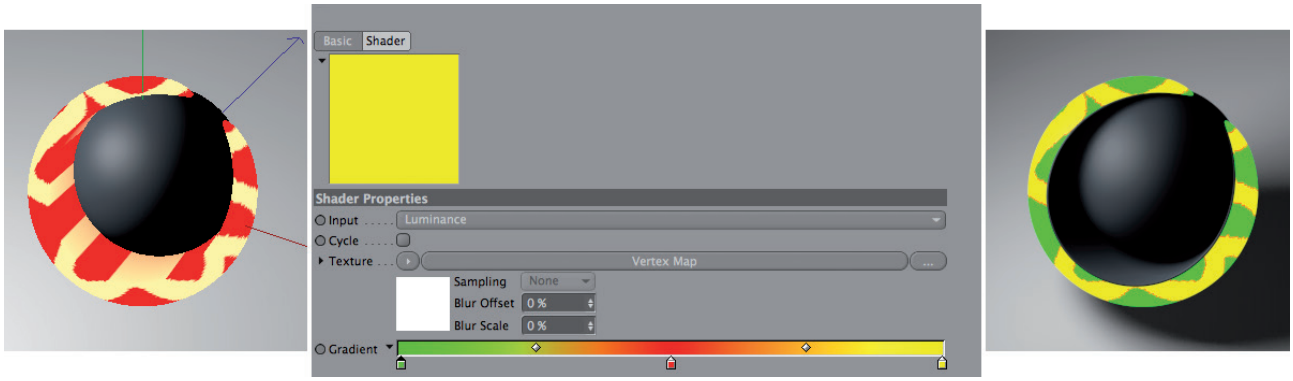
If the spline is closed, the **Fill** option can be enabled to fill its inner shape. If **Use Fill Color** is enabled, a texture can be used to fill the inner shape, otherwise the **Line Texture** will be used to fill the shape. If **Line Texture** is empty, white will be used. The background is black by default but can also be assigned a texture using the **Background Texture** option.

9.5.10.17 Vertex Map Shader

You should already be familiar with the **Vertex Map** from the **Live Selection** and **Brush** tool sections. With it, deformations can be restricted to certain surface regions and made to fade softly using a **Restriction** tag. This shader reads the **Vertex Map** tag's information and carries it over as brightness.

A vertex with a weighting of 100% will be white and a point with a weighting of 0% will be black. Soft transitions will automatically be made between vertices. **Vertex Maps** can, for example, also be used to create alpha masks in materials for transitioning softly from one material to another.

In the shader, all you have to enter is the name of the **Vertex Map** tag. Of course this tag must also be assigned to the same object to which the material is assigned. Enabling the **Invert** option will invert the shader's effect. This shader can of course be combined with a **Colorizer** shader if you want to add a custom color.



This will also let you use it in the **Color** channel, for example.

9.5.10.18 Weathering Shader

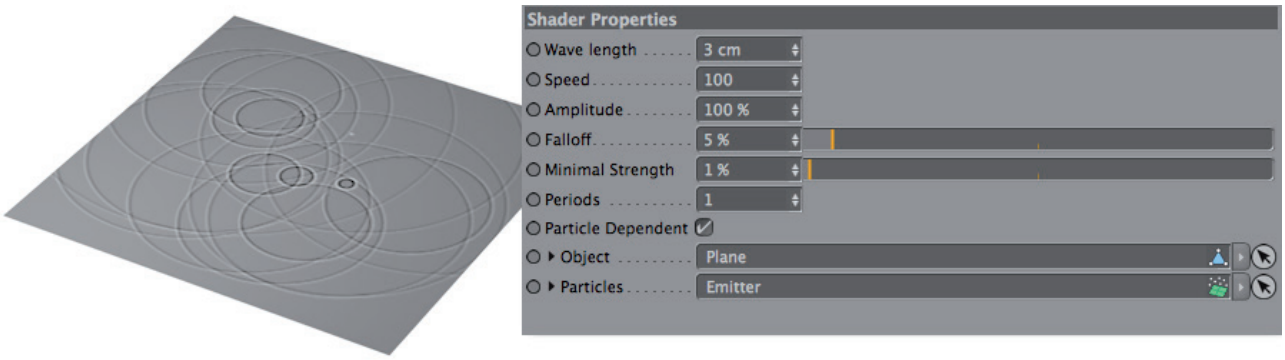
This shader can be used to make the loaded texture look washed out or weathered. After loading the texture, use the **Direction** setting to define the direction in which the **Texture** should be smeared. The **Intensity** value defines the effect's strength, whereby **Smoothing** defines the precision of the rendering. The higher the value, the more precise the rendering will be and the longer it will take to render.

The **Weighting** setting can be used to also make the effect dependent on the brightness within the **Texture** itself. For example, the **Dark** mode will cause those regions of the texture with a brightness of less than 50% to be affected more. Brighter regions will remain unaffected and the dark pixels will look like drips of paint passing over them.

Clamp UV coords defines how the image edges will be affected. If this option is disabled, those regions of the texture that are smeared off the image's edge will continue on the opposite side of the image. Enabling this option will prevent this from happening. We already saw how smearing or weathering can be varied depending on the texture's brightness. However, the effect cannot be controlled precisely enough when using color images or colored shaders as a texture. This is why a separate texture can be used whose brightness can be used to define the smearing. The **Intensity** shader's overall effect can be adjusted using its **Strength** value.

9.5.10.19 Ripple Shader

If you want rain drops to automatically create concentric circles when they hit a puddle, this shader can help.



You will need an **Emitter**, which is dragged in the shader's **Particles** field. When using **Thinking Particles**, the **Particle Geometry** object will have to be dragged into this field. Drag the object on which the concentric circles should appear into the **Object** field. As a rule, this will be the same object to which the **Ripple** shader is assigned. Note that this shader cannot be applied to Primitives. These must first be made editable (**Mesh / Conversion / Make Editable**). The **Particle Dependent** option is only relevant when using **Thinking Particles**. The speed with which the ripples expand will then be dependent on the size of the individual particles.

Wave Length defines the distance between ripples; **Speed** defines the speed with which the ripples expand; **Periods** defines the number of ripples per particle that hits the object. **Falloff** defines the height of the ripples as they move away from the drop's point of impact. **Amplitude** defines the height of the ripples at the time they are created. Ripples whose height lies below the **Minimal Strength** value will automatically be deleted, which helps save render time.

The calculated ripples will be output as gray scales. A ripple with maximum height will appear white. As the intensity decreases, the ripples will lose their brightness accordingly and will merge with the shader's 50% brightness.

This shader can be used in either the **Bump** or **Displacement** channel. The latter is only recommended for extreme close-ups if you actually have to visualize the rising and falling of a surface. Note that the shader's effect can only be seen for rendering – it cannot be seen when rendered in the Viewport!

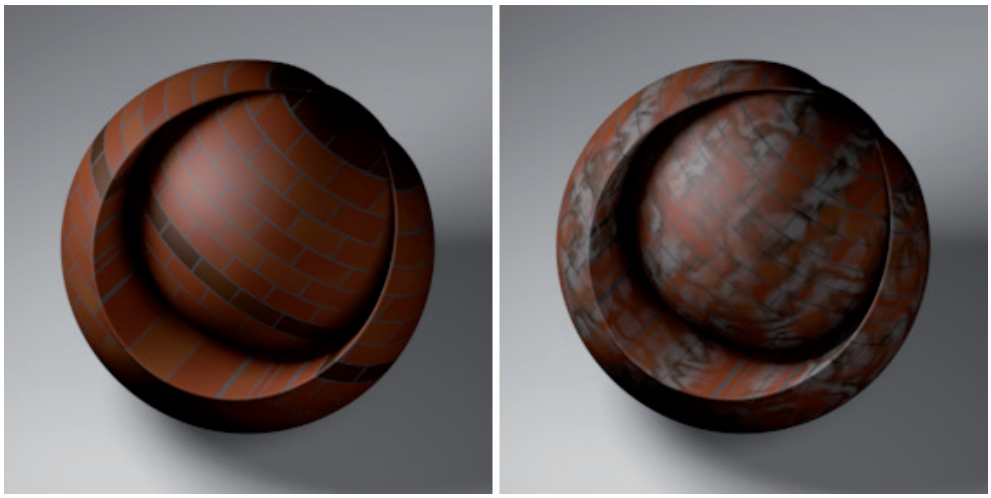
9.5.11 Surface Shaders

These shaders primarily generate patterns. Several of the functions will surely already be familiar to you. Others, such as the Tiles shader, create a very unique pattern. Generally speaking, these shaders can be used in all material channels with a **Texture** setting. However, the majority of these shaders are used in the **Color** or **Bump** channels.

9.5.11.1 Brick Shader

This shader contains all settings required to simulate a brick wall, including individual **Color** settings for **Brick**, **Gap** and an optional **Dirt** layer. In the **Shader** tab you can define the size of the bricks as well as the offset of the rows. Bricks with half width can also be included.

If all bricks should have the same size, set **Half Width Every nth Row** to 0. Otherwise the rows with halved bricks can also be indented using the **Shift** value. The **Balance Alt. Color Period** option halves the number of bricks with half width if these were assigned an alternative brick color. Even if bricks are not halved, they don't have to be aligned in a boring pattern. The rows can be offset using the **Shift** value. The **Reset Every nth Row** value will begin each row aligned on the left side. The general dimensions of the bricks is defined using the **Brick Width** and **Brick Height** values. The **Scale** value is simply a multiplier for this aspect ratio.



The **Color** tab's settings affect the coloring of the bricks themselves. Two colors can be configured independently of one another, **Brick Color** and **Alt**. (alternative) **Brick Color**. If you mix different colors in the gradients, the variation will be generated automatically.

The alternative brick color can be assigned to the bricks using the **Alt. Color Every n Rows** and **Alt. Color Every n Columns** values. Both colors can also be defined using **Textures**, which will in turn disable the gradients. When a **Texture** is loaded, a **Texture Details** menu will be made available below the **Texture** field. The **Offset** value can be used to define a random percentage offset of the **Texture**. If the **Random** option is also enabled, the effect will be amplified by an additional arrangement of the bricks' UV coordinates.

The **Scale** value scales the loaded texture. If set to 100%, the texture will appear exactly once on the **Brick** shader's tile. Smaller values will tile the texture seamlessly as long as it's an image. If a shader is used, the pattern will be made smaller or larger accordingly. Enabling the **Flip** option will rotate the texture 90° clockwise and mirror it vertically. If the **Opacity** value is reduced, color gradients can be made to show through the texture. Alternatively, **Blend Modes** can also be used.

The **Brick Noise Scale** setting affects the use of gradients on all bricks. Smaller values will produce a stronger contrast and more abrupt color changes within a smaller area. A noise pattern will also be used that lends the surface a slight roughness. The intensity of this nose is defined by the **Detail** value and its scaling via the **Detail Scale** value. This noise can be useful if you are using this shader in the **Bump**, **Normal** or Displabcement channel to give the gaps more depth and make the bricks look slightly porous.

9.5.11.1.1 Gaps Tab

Similar to the bricks, these settings can also be used to assign different random brightness or colors to the gaps using a gradient. The **Gap Noise Scale** value is used to define the randomness. Small percentages will create more variation. A **Texture** can also be used here, either as an alternative to the gradient or to supplement it. The **Depth** value defines the transition between the gaps and the bricks. Larger values will soften the transition accordingly. The **Size** value defines the width of the gaps and **Size Variation** can be used to create an irregular edge for the bricks. **Size Variation Scale** defines the shape of these irregularities. Larger values will produce correspondingly rough edges and indentations and smaller values will produce many irregularities within a small region along the edge.

If you want to use the shader in a **Bump** or **Displacement** channel, for example, the gaps' depth can also be accentuated by darkening them. **Groove** values of more than 0% will darken the center of the gaps. Negative values will darken the edges of the bricks.

9.5.11.1.2 Dirt Tab

Enabling the **Dirt** tab will add a color layer to the bricks and gaps, e.g., to simulate dirt. A **Color** gradient is used to define the dirt's color. If you click on the small black arrow to the left of the gradient, additional options will be made available that let you add editable alpha properties to the effect. Enabling **Edit Alpha** will let you define the **Dirt** layer's opacity using alpha values. The **Gap** and **Brick Blending** values are used to define the intensity and type of dirt relative to the underlying brick color. Values in excess of 0% will use a negative multiplier to blend and values less than 0% will use a dodge method. If set to 0% the colors will simply be overlain. The **Opacity** slider is a general multiplier for the dirt's visibility.

The structure of the dirt pattern is defined using the **Opacity Scale** and **Color Scale** values. Smaller **Opacity Scale** values will produce more details in the dirt's structure. The **Color Scale** value on the other hand only references the **Color** gradient and defines the size of the pattern used internally, which is responsible for selecting the color from the gradient. Smaller values will also produce more variation. The **Octaves** value defines the level of detail for the dirt structure, as is done with the **Noise** shader. The **Shift** setting can be used to offset the dirt structure. The X and Y values will move the pattern horizontally and vertically, respectively. Modifying the Z value will generate additional structures in the dirt pattern. Here you can also load a **Texture** to use in place of or as a supplement to the gradient. The **Texture** that is loaded will be used as an alpha mask so only the image's or shader's brightness values will be evaluated. The **Texture Details** menu can be used to **Shift**, **Scale** or **Flip** the loaded texture. The principle is identical to that of the shader's **Colors** tab. The **Shift** setting's **Z** value, however, is only meant for three-dimensional shaders such as the **Noise** shader. The **Multiply with Gradient** option references the **Color** gradient in the Dirt menu. If enabled, the **Texture** and **Color Texture**, if loaded, will be multiplied using the **Color** gradient. Otherwise the dirt layer's coloring can be loaded directly via the **Color Texture**, e.g., to create graffiti or posters on a wall.

Regarding the evaluation of the loaded texture as an alpha mask for the dirt effect, it can be inverted by enabling the **Alpha Invert** option and sharpened or blurred using the **Alpha Contrast** value. **Alpha Bias** can be used to increase or reduce the size of the regions affected by the dirt effect.

You can also use the **Rain** setting to give the dirt a washed out look. Values larger than 0% will cause the dirt to be smeared along the texture's V direction, i.e., vertically downwards, and simultaneously be made weaker. The **Samples** value defines the precision of the effect for rendering. More samples will produce a correspondingly more blurred effect. If you want to maintain a higher contrast in conjunction with the **Rain** effect, try using smaller Samples values first. If you use the **Color Texture** to generate the dirt, it can also be affected by the **Rain** effect by enabling the **Blur Color Texture** option.

Do you see the parallels between this effect and the **Weathered Shader**?

9.5.11.2 Checkerboard Shader

This shader creates a square tiled pattern. **U Frequency** and **V Frequency** define the density of the tiles in the X and Y directions, respectively. The **Color 1** and **Color 2** settings can be used to define a custom color for the surfaces.

9.5.11.3 Cloud Shader

This is also a pattern shader, which is used to create cloud-like structures. The Frequency values define the level of detail in both directions; Level defines the ratio of clouds to sky; the color of the sky and cloud are defined using the Color gradients. The Compatibility mode only needs to be enabled if you load a file created in an older version of Cinema 4D in which this shader was used. The rendering will then reflect that of the older version.

9.5.11.4 Cyclone Shader

This shader is very much like the **Galaxy** shader but it generates a cyclone effect on the entire surface. The **Rotation** value defines the number of complete rotations. The **T-Frequency** value defines the rotational velocity during animation – this shader is automatically animated. **Clouds** defines the number of surfaces that will be colored using the colors at the left end of the **Gradient**.

9.5.11.5 Earth Shader

If you need an abstract map, this is the right shader. This shader automatically generates oceans, land masses and mountain ranges and colors them individually. The percentage of land can be adjusted using the **Level** value and the **Frequency** value is used to vary the size of the land masses.

9.5.11.6 Fire Shader

This shader lets you define a custom color for fire using a **Color** gradient. The color at the left end of the gradient represents the background color and the remaining colors represent the fire. The **U Frequency** and **V Frequency** values define the number and length of flames. The **Turbulence** value defines the width of the flames – the larger the value, the more narrow they will become, and the more the flames will be interrupted along their length. Since this shader is automatically calculated for animation, the **T Frequency** value is used to define the speed with which the flames are varied.

The shader also masks the flames automatically, which means you can, for example, produce flames for a fireplace if the shader is used in a material's **Alpha** channel and **Luminance** or **Transparency** channels.

9.5.11.7 Flame Shader

Similar to the **Fire** shader, this shader can also be used to create flames of any color. This shader is also animated by default and creates its own alpha mask. However, this shader only creates a single flame. The **Turbulence** value can be used to add variation to the flame and the **T Frequency** value modifies the variation during animation. Larger values result in correspondingly faster variation.

9.5.11.8 Formula Shader

A formula can be used to calculate brightness values, e.g., based on an object's User Data, the U, V and W or X, Y and Z coordinates.

9.5.11.9 Galaxy Shader

This pocket version of the Milky Way can have any number of **Spiral Arms** and can be given any rotation **Angle**. A custom **Color** can be assigned using the gradient.

9.5.11.10 Marble Shader

This shader calculates patterns similar to the way the **Simple Turbulence** and **Simple Noise** shaders do. This pattern is designed to resemble stone or marble. The **Frequency** setting defines the shader's irregular three-dimensional density. The **Turbulence** value defines the pattern's randomness and the **Color** gradient can be used to define a custom color.

9.5.11.11 Metal Shader

Contrary to what you might think, this shader is not used to create a metal surface. This is a diffused patter that can, for example, be used to add variation to a surface color. The **Frequency** value defines the amount of randomness and the **Color** gradient can be set to any custom color combination.

9.5.11.12 Pavement Shader

This shader creates a pattern that looks like stone pavement – or even like a dry riverbed. The **Scale** value defines the density and therewith the scale of the stones. The **Gap** menu's **Width** value defines the width of the gaps between the stones. **Crookedness** is used to generate deviations in the otherwise straight stone edges.

Three gradients are used to define the color of the stone, grout and stone edges, respectively. In addition, random variations of rough and fine surface structures can be created using the **Structure** and **Grain** settings. **Rough Structure** supplements larger, brighter regions and **Fine Structure** adds a fine, light noise. The **Grainy Gap** works similarly but restricts its noise effect to the gaps/grout. The higher the value, the more visible the additional effects will be. Enabling the **Contrast Grain** option will modify the noise structure in the gaps so that they are more uniformly spread and not only located in the middle of the gaps.

The gaps can also be darkened or made to look dirtier using the **Smudgy Color** gradient. The **Smudgy Edges Size** value defines the degree with which the effect is spread, which can also be made to include the stones' edges. The **Smudgy Edges** value defines the opacity of the effect.

9.5.11.13 Planet Shader

How about creating a complete planet using a shader? The **Planet** shader lets you apply the typical color schemes for the planets **Saturn**, **Uranus** and **Neptune**. If the material is assigned to a sphere, the color gradients will be generated automatically on the surface. Even **Saturn's Rings** can be selected from the **Type** menu, which means that this shader will also have to be used in the material's **Alpha** channel. The material should then be assigned via **Flat** projection onto a **Plane** or **Disc** object.

9.5.11.14 Rust Shader

If you want to simulate flaking paint or rust, this shader is the one you should use. The **Color** gradient defines the base color (left) and the color of the rust or paint spots (right). The **Rust** value defines the amount of color from the right end of the gradient that will be used; the **Frequency** value is used to define the shader's irregular pattern.

9.5.11.15 Simple Noise Shader

This shader works like the **Simple Turbulence** shader but creates a noise pattern instead. The primary difference between this and the normal **Noise** shader is that you can define a custom gradient **Color**.

9.5.11.16 Simple Turbulence Shader

This shader creates a cloud-like pattern that can be scaled in the X and Y direction using the **U Frequency** and **V Frequency** values, respectively. The **Octaves** value is used to adjust the pattern's level of detail. The **Color** gradient can be assigned any color, which is also the main difference to the **Noise** shader that is used to create similar patterns.

9.5.11.17 Starfield Shader

This shader can be used if you only want to have a diffused dispersion of pixels of varying brightness. This shader makes no other settings available except for the **Basic Properties** settings, e.g., to add blur with the **Blur Offset** value.

9.5.11.18 Stars Shader

We've covered just about every sky object, whereby the **Star** shader creates more of an illustrated look. **Color 1** defines the background color; **Color 2** defines the fill color for the stars. The number of points the stars have can be defined manually using the **Streaks** setting. The **Inner** and **Outer Radius** of the stars can also be modified as well as the average number of stars per material tile (**Density**).

9.5.11.19 Sunburst Shader

This shader can be used to create a fire-like corona or ring. The color at the left end of the **Color** gradient defines the background color. The colors to the right define the flames' color. The **R Frequency** value defines the variation within the flame-like bands. Larger values can result in gaps being produced in the flames. The **A Frequency** value is used to shape the radial bands. Small values will only create a soft cloud and larger values will produce an irregularly-shaped ring. Since this shader is also animated, the **T Frequency** value is used to define the speed of the modifications over time. The **Turbulence** value defines the scale of turbulence for the flames. Often, larger values will create correspondingly more narrow flames.

The **Radius** value defines the size of the opening at the center; the **Height** value defines the length of the flames. This shader also automatically generates an alpha mask, which means it can also be used in the material's **Alpha** channel so only the corona remains visible. With a little fantasy, this shader can be used to create other interesting effects such as the iris of an eye.

9.5.11.20 Tiles Shader

This shader can be used to create numerous sub-ordinate patterns, including planks, tiles or stripes and circles in different directions and arrangements. The color of the tiles can be defined individually and various **Scale** settings let you create uniform or non-uniform tile sizes. Custom colors can also be defined.

9.5.11.21 Venus Shader

This shader looks similar to the surface of the planet Venus. The settings are the same as those of other shaders. The **Frequency** value is also used to define the level of detail along all three axes. The **Rotation** value is used to rotate the pattern and the **Color** gradient can be used to define a custom color.

9.5.11.22 Water Shader

The focus of the shader is less the transparent or reflective properties of water than the typical wavy look of water, e.g., that is common on the ocean. If scaled accordingly, this effect can, of course, be used for smaller bodies of water. As a rule, this shader is used in the **Bump** channel.

The **U Frequency** value defines the distance between the waves along the X axis. Larger values will produce correspondingly more waves. **V Frequency** varies the wave crests and creates more disruptions in the pattern's Y direction. **T Frequency** is the measure for the amount of modification for the wave over time since this shader is also animated by default. **Wind** defines the waves' forward velocity. Higher **Frequency** values can generate banding patterns, e.g., like bursting metal, the folds of drapes or even hair.

9.5.11.23 Wood Shader

This is the shader you should use if you only want a simple wood grain without the comprehensive settings offered by the **Banzi** material. The **Wood** shader even works three-dimensionally and realistically permeates the object to which it is assigned.

The **Type** menu can be used to select from one of several pre-defined gradients. You can also define a **Custom** gradient.

This shader offers a simple and a more complex mode. The latter can be used to create more realistic-looking wood grain. If **Legacy** is enabled, only the simple mode will be available. This mode's **Frequency** setting can be used to define the amount of detail for all three axes independently. The **Turbulence** value is used to define the turbulence for the annual rings, which lie on the texture projection's XY plane by default. For example, the annual rings can be positioned precisely on the object using a **Flat** projection in **Enable Axis** mode.

If **Legacy** is disabled, several additional settings will be made available. The grain's overall size can be adjusted to fit the object size. **Ring Size** defines the width of and distance between annual rings, which are used to render the wood grain. The **Stretch** value can be used to adjust the elliptical distortion of the annual rings. Larger values simulate correspondingly longer tree trunks.

Grain can be added to prevent the gradients from looking too simple and add a random dispersion of color values. The intensity or amount of grain is defined using the **Grainy** value, and the **Grainy Scale** defines the scale of the random color spots.

Wavy defines the random deviation from the original wood grain, which is primarily created by the stretched annual rings. The higher the **Wavy** value, the more distorted the structures will be. **Wavy Scale** defines the distance between these distortions.

The **Shift** value is the value on which all random modifications to the wood grain are based. Each time this value is change, the wood grain will be re-calculated. This makes it possible to add an endless number of variations to the wood grain to wood that would otherwise have a uniform color, which makes the application of this shader to different objects much easier.

If the **Annual Rings** option is enabled, the annual rings will be drawn on the texture projection's XY plane. This plane can, for example, be positioned freely on the object using the **Texture** tag's **Flat** projection in **Enable Axis** mode. The annual rings are circular by default but can be made more wavy using the **Wavy Rings** settings. The **Wavy Rings Scale** setting offers three percentage values that can be used to modify the wavy effect along each axis individually. Note that the **Wood** shader's preview image will look much different from the actual rendered result because this shader calculates its effect in 3D. You should therefore use the **Interactive Render Region** function when fine-tuning this shader.

▶ *See: Excersises for materials and shaders*

SUMMARY: CHANNEL SHADERS

- As a rule, **Channel Shaders** are used to create individual effects or patterns that can in turn used in a Cinema 4D default or Shader material.
- The **Surface Shaders** are generally only used for creating color patterns that can, for example, be used in a Color channel.
- The **Effect Shaders** analyze an object's shape or its environment and use this information to produce unique results.
- Light that penetrates, passes through or is diffused in a surface within a given material can often be rendered quickly and with sufficient precision using the **Backlight** or **ChanLum** shaders.
- Physically correct simulations of diffused light in a material are only possible using the **Subsurface Scattering** shader.
- The distance between objects can be calculated to ascertain brightness values. This is how the **Ambient Occlusion** Shader darkens surfaces in grooved regions, corners or where objects touch. This effect is dependent on the lighting of the objects and can only be used in the **Diffusion** channel.
- **Ambient Occlusion** can also be enabled as an effect in the **Render Settings** menu, which will apply the effect to all surfaces in the scene. The effect will render faster and can also be output as a separate **Multi-Pass** layer.
- Additional **Shaders** are divided into two groups: those that can be used individually such as **Color**, **Gradient**, **Noise** or **Fresnel**, and those that require the use of a texture. These include, among others, the **Colorizer**, **Layer**, **Filter** and **Fusion** shader. The latter group makes it possible to use complex shader trees with which any number of Shaders and images can be mixed.
- Shaders generally bear the advantage over image textures that no resolution has to be defined for them. They can also be edited relatively easily and quickly using the available settings.

9.6 Material Tag

Whenever a material is assigned to an object, a **Material** tag is automatically created for that object in the *Object Manager*. This tag establishes the connection between the material and the object and defines the projection type, position and scale of the material.

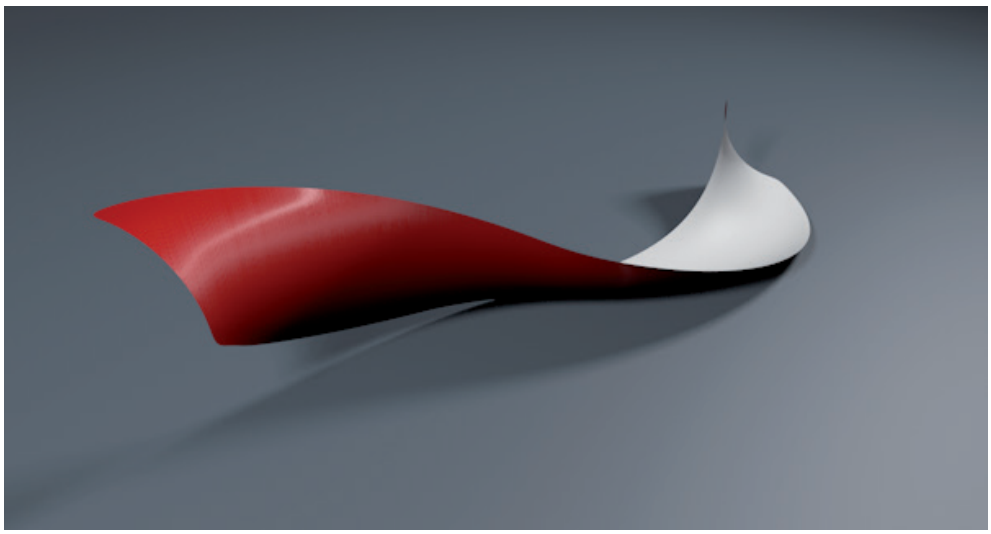
If the material does not contain any textures, e.g., that make up a given pattern, you don't have to worry about the **Material** tag at all. Many materials that only consist of color or only use basic properties such as reflection or transparency do not require the use of a texture. However, if a material does contain a texture – or textures – the projection of the material onto the object must be defined correctly. This can be done using the **Material** tag's **Projection** setting.

If the **Material** tag and the corresponding object are selected you can switch to U**se Texture** mode to display a preview of the projection type in the Viewport. This type of preview display can be selected from the **Projection Display** setting.

If **UVW Mapping** is selected as the **Projection** type, this mode will be useless because the adaption of the material to the surface will be done automatically by evaluating the object's **UVW** tag (if present). Note that this tag can be present even if it's not displayed in the *Object Manager*. This is the case for all parametric Primitives (cube, sphere, etc.) and for all objects that are created using splines (**Extrude**, **Lathe**, **Loft**, **Sweep**). If **Fontal** or **Camera Mapping** are used you will not be able to position or scale the projection freely because the material will use the observer's or camera's angle of view for the projection and will orient itself according to the **Render Setting** menu's **Output** setting for scaling the material. All other projection types can, however, be moved, scaled or rotated using the normal tools. The **Material** tag's **Coordinates** menu displays the texture's position, scale and rotation. Double-clicking on the **Material** tag will make the corresponding material available for editing. The material can be easily replaced by dragging the new material from the *Material Manager* into the tag's **Material** field. All of the projection settings of the material that was replaced will automatically be assumed by the new material. The **Selection** field can be used to load a saved polygon selection. A polygon selection can be created by selecting the polygon(s) you want to define separately for the texture and selecting **Set Selection** from the main Cinema 4D **Select** menu.

The resulting **Polygon Selection** tag can be renamed and dragged into the **Material** tag's **Selection** field. This material will then only be assigned to the surfaces saved for that **Polygon Selection** tag. Multiple **Polygon Selection** tags can be created for a single object, which is a good way of assigning different materials to different regions of a given object. A material can be assigned even faster if the polygons are selected in the Viewport. If you drag the material from the *Material Manager* onto the selected polygon(s), a **Polygon Selection** tag will automatically be added to the **Material** tag. If multiple material are assigned to an object the order in which they are arranged in the *Object Manager* will define which material will lie under another material. **Material** tags to the left will lie below **Texture** tags to their right on the object. If a material uses an **Alpha** channel, the material below it will be visible through the alpha mask. As you know, polygons have two sides. The **Material** tag's **Side** menu defines if the material should be assigned to the **Front**, **Back** or to **Both** sides of the polygon.

Note that the restriction to the back or front side is not always displayed correctly in the Viewport. Therefore, you should do a test render using the **Interactive Render Region** function to make sure the texture is assigned correctly.

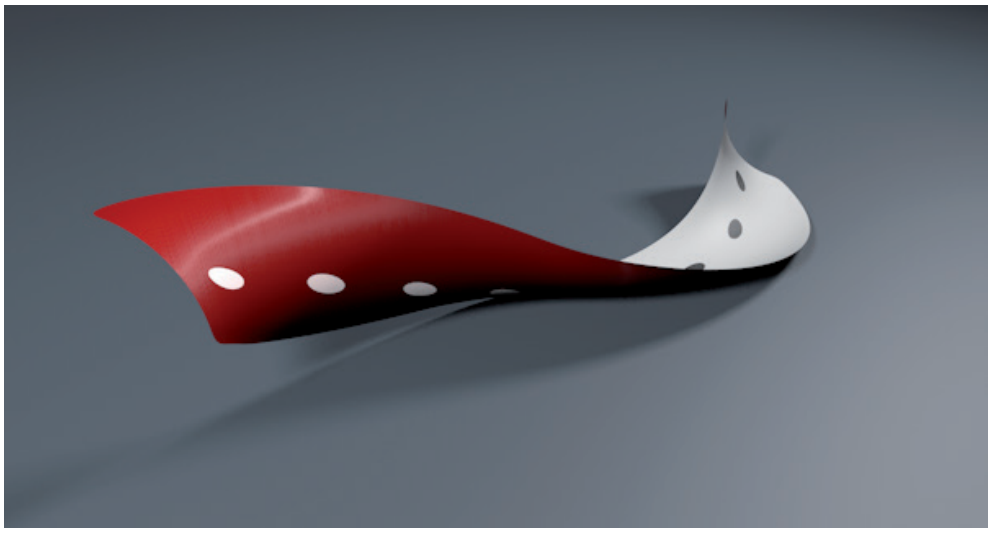


The **Add Material** option can be enabled if multiple materials are assigned to a single object. If this option is enabled for a texture to the right of a given material, its **Color**, **Luminance**, **Transparency**, **Reflection**, **Bump** and **Displacement** channels (if used) will be mixed with those of the material to its left. This also works with multiple tags simultaneously. This makes it possible, for example, to mix different projections without having to use the **Projection** shader.

Note that there is a restriction to using materials additively. Materials with different **Blur** settings and with different refraction indices cannot be mixed freely. A unique case is the use of displacement in a **Material** tag that lies deeper, i.e., arranged farther to the left. This will remain visible even if the added material does not have an active Displacement channel.

The **Offset U** and **Offset V** values can be used to move the material within the defined projection shape. If **Tile** is enabled, the number of repetitions for the material within the projection shape can be defined using the **Tiles U** and **Tiles V** values. The **Length U** and **Length V** values have the same effect. If **Seamless** is enabled, every other tile will be arranged mirrored, which can help hide the seams between material tiles. The **Use UVW for Bump** option should generally be left enabled because it improves the quality of the bump rendering on the surface.

Use the **Repetitions U** and **Repetitions V** values to define how often the tiles will be repeated for each direction. Note that very small values can result in only part of the tiles being visible. Otherwise, a value of 0 will cause the tiling to run indefinitely in all directions until the entire object is covered (see **"TextureTags"** Project).



9.7 Pin Material Tag

One advantage of using **UV** coordinates is that the material stays attached to the object's surface even if it's reshaped or deformed using **Deformers**. This is why the **UVW Mapping** method is a must, for example when texturing animated characters.

If, for example, you should for some reason have to use a **Flat** projection on an object that will be deformed, there is a solution for this as well: the **Pin Material** tag. This tag can be found in the *Object Manager's* **Tags / Material Tags** menu. Internally, this tag will create a copy of the position of each vertex on the object and will use this information to recalculate the texture position when the object is deformed based on the object's initial state. Texture pinning also works with multiple materials, which means that a **Pin Material** tag does not have to be created for each **Material** tag individually. This tag has two buttons in its *Attribute Manager* menu. The **Record** button saves the current point positions in the tag. This function was already executed when the tag was created. Hence, you only have to click this button if the object shape or number of vertices was modified after the tag was assigned to the object. Because this tag can only save vertex information it can, of course, only be used with polygon objects. Since parametric Primitives and objects that use splines do not contain any vertices, this tag cannot be used with such objects.

The **Reset** button sets the object back to the state that was saved by the tag. This can be helpful when working with **PL** animations (**Point Level Animation**). With PL animation, keyframes are set for each vertex' position of a given object. The **Active** option lets you enable or disable the tag itself. This has no affect on the vertex coordinates saved by the tag.

9.8 Editing and Converting Projection Types

We already discussed the main differences between the standard projection types such as **Flat** or **Spherical** and the **UVW Mapping** method. The latter uses information saved by the **UVW** tag that assigns a specific section of the material to each surface vertex. However, you can, for example, convert a texture projected via **Flat** mapping into one with **UVW Mapping**.

To do so, simply select the object and the corresponding **Texture** tag in the *Object Manager* and select the **Generate UVW Coordinates** command. The **Flat** projection will then be converted to UVW coordinates and a new **UVW** tag will be created. The texture's projection type will automatically be switched to **UVW Mapping**.

This makes it possible to create any number of **UVW** tags for a given object. If multiple **UVW** tags are assigned to an object, each **Material** tag will always use the **UVW** tag directly to its right. In combination with restricting textures to polygon selections, the entire object can be assigned several different projection types without having to edit UV coordinates, e.g., with BodyPaint 3D.

SUMMARY: MATERIAL TAG

- The connection between an object and a material is always made using a **Texture** tag.
- The **Material** tag is linked to the material. A different material can be assigned simply by replacing this link. The projection settings will be maintained.
- Multiple **Material** tags and therewith multiple materials can be assigned to a single object. If a material is not restricted to a polygon selection, tiling is not restricted and no alpha channel is used, a **Material** tag will cover all **Material** tags to its left in the ***Object Manager***.
- If a projection type other than **UVW**, **Frontal** or **Camera Mapping** is used, the material's position and size on the object can be modified when in **Enable Axis** mode. The object and **Material** tag must be selected. A preview of the texture projection will be displayed in the Viewport, which can be moved, scaled or rotated using the normal tools.
- If the object will be deformed, the material on the surface will shift unless **UVW Mapping** is used. The **Pin Material** tag can be used to add UVW coordinates.
- Alternatively, standard projection types can be converted to UVW projections.

10 Using Cameras

In the Perspective view we can adjust the view of the object to best fit our needs. However, this view cannot entirely replace a real camera. For example, the focal length cannot be defined and several lens effects such as depth of field cannot be simulated, as can be done with a **Camera** object. The Perspective view can be used to get a rough idea of what the angle of view should be and a **Camera** object should be used for rendering.

There are several types of cameras from which you can choose, for example, the normal **Camera** object, the **Target** camera and the **Stereo** camera. The **Target** camera is a normal camera that also has a **Target** expression, which can be used to automatically point the camera at a specific object in the scene. This works the same as the previously discussed **Target Light** object. The **Stereo** camera is designed to be used to create stereoscopic images. The remaining cameras are designed for use with animation and can, for example, be used to cut between camera positions or to more easily simulate a hand-held camera or a camera crane. These are all normal **Camera** objects with additional tags to enhance their function. Hence, these functions can always be subsequently added to other **Camera** objects.

10.1 Activating and Positioning Cameras

As soon as a **Camera** object is created, it will be positioned and oriented according to the Viewport that was active when it was created. Therefore, **Camera** objects should be created when the Perspective view is active. Activating the **Camera** object itself, however, requires an additional step. Next to the camera's name in the ***Object Manager*** you will see a small, black, square icon. If you click on this icon it will turn white and the camera will be activated and linked with the Perspective view. Alternatively, you can select the camera from the Viewport's **Cameras**/**Use Camera** menu. This additional activation step is required to you can use as many cameras as you want in the scene and switch between them freely.

If a camera is activated and, for example, linked with the Perspective view, it can be moved, rotated and zoomed using the normal icons at the top right of the Viewport window or by using the keyboard shortcuts. In the remaining views you will see that the **Camera** object is like any other object, which also has an axis and can be moved and rotated in **Use Model** mode. The camera's angle of view will be along the **Camera** object's Z axis. Therefore, if you move the active camera in the Side view, for example, the camera's angle of view will also be affected in the Perspective view automatically. Since the camera is itself a unique object, it can also be linked with polygon objects. For example, if you want to move with an animated car in the scene from the driver's point of view. All you have to do is make the **Camera** object a sub-object of the car model in the ***Object Manager*** and position it above the driver's seat accordingly. When the car moves, the camera will move with it and you can render an animation looking through the car's windshield. If the animation, or even a still image, rendered using this camera is subsequently edited using a compositing software, e.g., Adobe After Effects, the camera's 3D information can be used in conjunction with the **Export to After Effects** option. However, additional settings are required in the **Save** setting located in the Render **Settings** menu, which we will discuss later.

10.2 Defining Image Size and Focal Length

When viewing the **Camera** object in the Viewport you will see that it has a green pyramid shape, which defines the camera's angle of view. This is called the **viewing frustum (Cone)**, which can be adjusted in the Side or Top views to help position the camera.

Only those objects that lie within the **Cone** will be visible in the Perspective view and therewith in the rendered image. The **Cone** can be hidden by using either the Viewport's **Filter** settings to hide the camera itself or by disabling the **Show Cone** option in the **Camera** object's **Details** menu in the ***Attribute Manager***.

The **Cone's** angle – and therewith the size of the region of the scene that is visible – depends on several factors. The first one is the resolution that is defined in the **Render Settings** menu. This menu was already discussed in the material system section and during the explanation of the various test render methods. Click on the **Render Settings** icon in the top icon palette to open the dialog window or select **Edit Render Settings** from the main **Render** menu.

Click on **Output** in the left column to make its settings available. Here you will find the **Width** and **Height** settings, whose ratio is displayed in the **Film Aspect** setting. This aspect ratio will automatically be adapted by the **Camera** object's **Cone**. The aspect ratio can, for example, be modified to create a portrait or landscape layout. The remaining settings have already been explained. Instead, we will take another look at the **Camera** object's ***Attribute Manager*** settings. Here you will find the **Sensor Size** and **Focal Length** settings. The **Sensor Size** setting defines the film size for an analog camera or the size of a digital camera's CCD sensor. You can select from numerous pre-defined settings from the menu at the right.

The **Focal Length** value is calculated internally as millimeters and defines the distance from the film/sensor at which the lens' optical rays cross. The simple geometric relationship between the **Sensor Size** and the **Focal Length** is defined by the camera's horizontal aperture angle, the **Field of View (Horizontal)** setting. The aspect ratio defined in the **Render Settings** menu defines the vertical aperture angle, i.e., the **Field of View (Vertical)**. All these settings are interdependent.

Therefore, modifying the **Field of View (Horizontal)** value will also affect the **Focal Length** value since the Sensor Size will be maintained. In fact, you will only have to deal with the **Sensor Size** value if you want to create a composition made up of live footage or images and 3D objects, for example. Otherwise, this value can remain set to 36. The type of perspective distortion is defined exclusively by the **Focal Length** value and the image's aspect ratio.

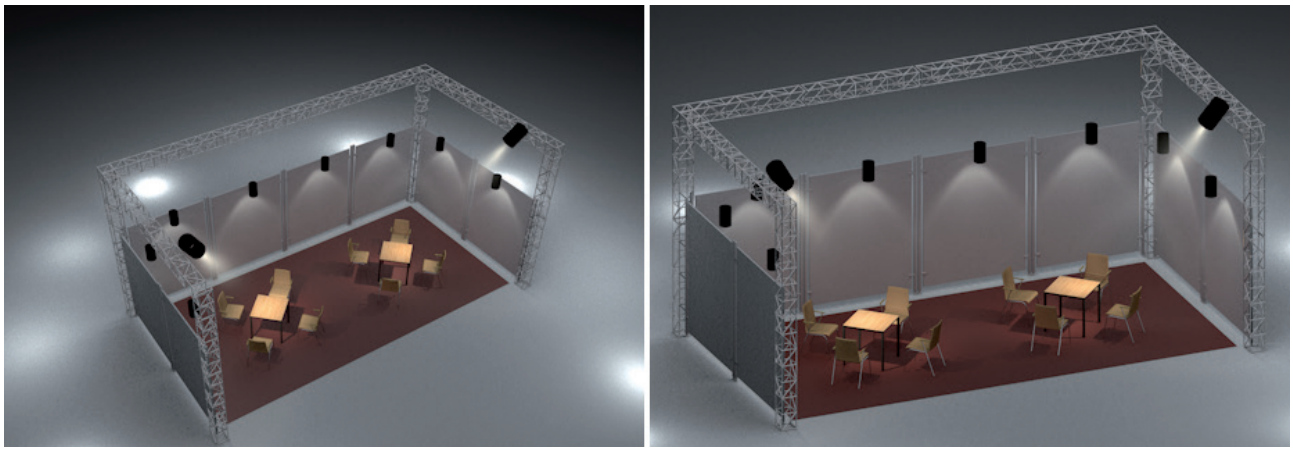
Larger **Focal Length** values simulate telephoto lenses and smaller values simulate wide-angle lenses. Hence, large values can be used to zoom in on objects and smaller values can be used to view a larger region of the scene, which is well suited for use in tight spaces.

The **Focal Length** also has an affect on the perspective of the scene. The greater the telescopic effect, the larger the **Focal Length** will be and the more illustrative and unnatural the objects will appear. Lines that lead away from the camera will appear to be parallel in extreme cases and will no longer converge at a common vanishing point. Very small **Focal Length** values will reverse this effect and can make objects look unnaturally distorted. Both effects can, of course, be used intentionally but in most cases you will be attempting to simulate natural-looking scenes anyway.

Common **Focal Lengths** based on a 36 mm images size therefore lie between 30 mm and 75 mm. The visual distortions are not very pronounced within this range.

10.3 Projection Types

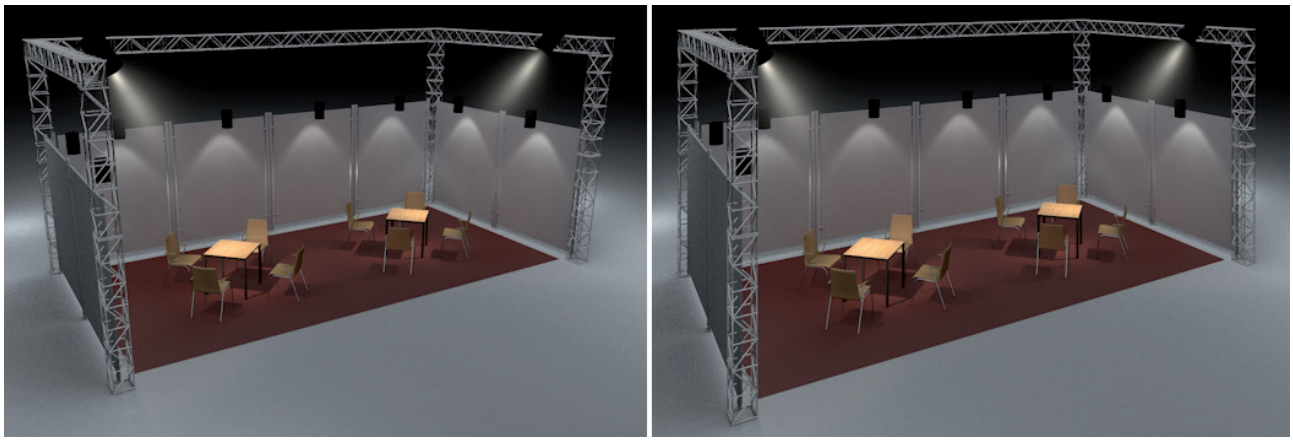
Until now we've always assumed that the camera works according to the Perspective view's principles. However, other standard perspectives are available for technical visualizations and other purposes. **Gentleman**, **Bird**, **Frog** or **Military** perspective views use standardized angles of view linked to the objects in the scene and can also be used for rendering.



The various projection types can be found in the **Camera** object's **Projection** menu in the **Object** tab. In addition to the aforementioned types, the standard view types can also be selected from this menu. A scene can be rendered frontally, from the top or from the side, if necessary. Note that the camera's movement can be greatly restricted, depending on the type of projection selected. Some projection types are clearly defined and can therefore not be varied individually.

However, the camera's zoom depth can be defined freely for all types with the exception of the Perspective view. The **Zoom** value can be used to adjust how far the camera is zoomed into or away from the scene without having to move the camera itself.

You can also offset the image using the **Film Offset X** and **Y** settings, which can also be used to create interesting effects in conjunction with the **Perspective** projection type. The **Film Offset Y** value in particular can be used to correct aberrant lines. These occur when a camera is not oriented precisely towards the horizon, causing vertical lines seeming to seeming to converge at a different vanishing point and thus not appearing to be parallel. Although this effect is visually correct, they are often considered to be bothersome for architectural renderings. To compensate, the camera's Z axis will be oriented parallel to the floor and the angle of view will be corrected using the **Film Offset Y** value. As a rule, a **Pitch** angle of 0° will be used for the camera. Vertical lines will then in fact appear to run parallel to one another.



Otherwise, the **Film Offset** values can be used whenever you want to move an element that lies too close to the image's edge towards the center of the image. The advantage this bears over moving or rotating the camera itself is that the perspective of the objects viewed will not be affected if the **Film Offset** values are used. The effect is as if the objects were lying on a 2D plane that is moved in front of the camera.

10.4 White Balance

If you were to hold a white piece of paper under a neon light and then in sunlight you would notice that the paper's color differs slightly in each setting. Neon light is a cool light, which gives the paper a very slight blue hue; sunlight is generally more reddish, which means that it will color the paper accordingly. To make white paper actually look white, photographers use what is called a 'white balance' effect, which corrects the image's color mood by attempting to compensate the coloring using existing light sources. The Cinema 4D **White Balance** setting works in a similar fashion.

Here you can select from various color temperatures for your scene. Smaller **Kelvin** values (**K**) generate a correspondingly more reddish light, i.e., warm light. The larger the values, the cooler and more bluish the light will be.

If you want to apply this function to your scene, select the **White Balance** option that best suits your scene with regard to color temperature. This can, of course, only be estimated and verified using test renderings. If a white object is actually rendered white then you know you have the right color temperature. Of course, this only applies assuming you want this correction to be made in the first place. Otherwise you can also use the **White Balance** to create a cooler or warmer color mood for your scene. In this case, the **White Balance** value will work inverted, i.e., larger values will make the scene look warmer and vice-versa.

In any case, the **White Balance** option **Custom** lets you define your own color temperature. Finally, you might remember the discussion about the **Light** object's color settings and how the color can be defined using the **Use Temperature** option. If **Affect Lights Only** is enabled, those lights configured via **Use Temperature** can be isolated and corrected via the **White Balance** setting.

10.5 Simulating a Focal Point

If you've ever photographed yourself, you are surely familiar with the camera's auto focus function, which automatically ascertains the distance of objects in the finder so it can focus on them. The aperture then defines the depth of field, i.e., those regions of the image that are seen in focus or less in focus. It's not always desirable to have the entire image completely in focus. A viewer's eye is always drawn to that part of the image that is most in focus, and the spatial relationship between objects is also more pronounced when more distant objects are rendered correspondingly more out of focus.

If you render an image in Cinema 4D you will see that the image is rendered with all elements in the scene equally in focus. Distant objects are rendered just as sharp as those that lie close to the camera. However, Cinema 4D cameras can also simulate depth of field. First, the **Focal Distance** has to be defined. This setting is located in the **Camera** object's **Object** menu and its value is automatically linked to the handle at the end of the camera's **Cone**. This makes it easy to adjust the **Focus Distance** interactively in the Viewport. This can be done even faster by clicking on the cursor icon next to the **Focus Distance** setting and clicking in the Viewport on the location at which the camera should focus.

Alternative to this manual method, you can also let the camera focus automatically. There are two options with which this can be done. You can assign a **Target** tag to a camera or if you can use a **Target** camera, which can be linked directly to a specific object (**Target** object). Enabling the **Use Target Object** option will use the distance between the camera and target object as the **Focus Distance**. This is good when focusing on a specific object during a camera animation. This is very practical but also bears the disadvantage that objects in focus will always lie at the center of the image. Using a **Focus Object** (e.g., a **Null** object) in the field of the same name will let you be more flexible.

The camera will then only measure the vertical distance to this object without pointing directly towards it. You can also use an additional **Target** object for orientation as long as the **Use Target Object** option is disabled.

Defining the focus plane/distance is only the first step. You can choose between two methods for depicting the amount of blur beyond this plane. We will begin with the manual setting via the camera's **Details** tab. The alternative would be to enable Physical rendering for the camera via the **Physical Renderer**. We will discuss this later.

10.5.1 Manually Defining Depth of Field

The **Details** tab offers two options with which you can define blurred regions within the scene separately or together. **DOF Map Front Blur** defines the blur between the camera and the defined **Focus Distance**; **DOF Map Rear Blur** defines the range beyond the defined **Focus Distance** within which objects will be blurred.

The respective **Start** and **End** values can be used to define a range, starting at the focal point, within which the blur should slowly increase. However, another step is required to actually be able to see this effect in the rendered image. Since this is a **Post Effect**, it must be activated in the **Render Settings** menu. You should already be familiar with this concept from the **Hair** and **Lens/Glow** sections. We will discuss this in detail in the render **Effects** and **Multi-Passes** sections.

10.5.2 Physically Correct Focus and Blur Rendering

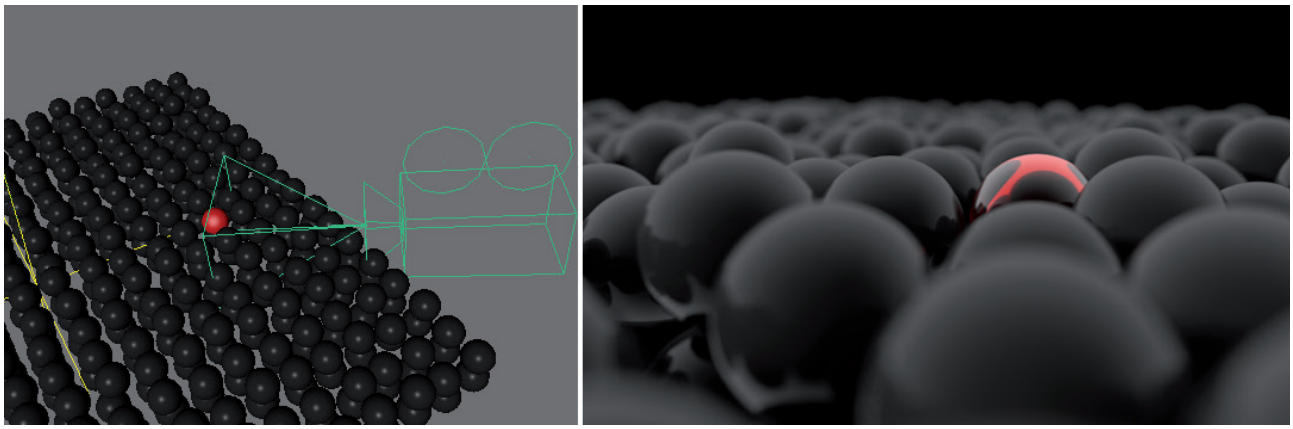
You just learned a simple method of defining different levels of focus for rendering. This is very practical but as a rule has nothing to do with depth of field that is created by a real camera's lens via its aperture. The lens uses small aperture values for large lens openings, which let correspondingly more – or less – light pass through. The lens system can be used to focus very precisely at certain distances, which produces a very narrow focal region. Objects that lie in front of or beyond this region will quickly become blurred. This effect is also dependent upon the focal distance. You might already be familiar with this concept from the field of macro-photography. A flower photographed up close is made much more interesting by the blurred background. The same aperture setting in conjunction with a very distant focus plane will not create any blurred regions.

In the following, note that the camera's **Physical** settings will for the most part only be evaluated if the **Physical Renderer** is used. Only the **F-Stop** setting will also be evaluated by **ProRender**. When rendering with the **Standard Renderer**, all inputs will be ignored for these settings.

When simulating a real-world camera you can choose between a photo camera and a movie camera. The primary difference between the two are the settings with regard to how motion blur is rendered. If **Movie Camera** is disabled, the image's brightness and the strength of the motion blur is controlled using the **Shutter Speed (s)** setting. If the **Movie Camera** option is enabled, you will be simulating a rotating shutter. In the end, either method can be used to achieve the same effect: fast-moving objects will be blurred and not rendered in focus. This can be used for animations or still images. An object can, for example, be animated using keyframes and a single frame from this animation can be rendered to realistically simulate motion blur in a still image. Additional options for creating these types of effects are available in the **Post Effects** menu in the **Render Settings** menu. Some of them can even be used without the **Physical Camera**.

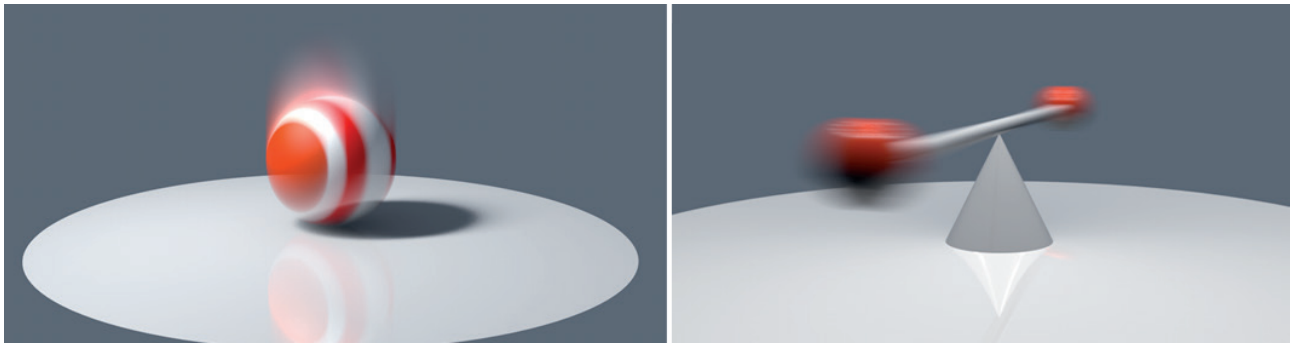
First we will take a look at the camera's **Physical** tab's options with the **Movie Camera** option disabled. The first setting is **F-Stop (f/#)**. Small values will let more light pass into the camera and can restrict the focus to small regions around the focus distance; larger values will allow correspondingly less light to pass into the camera an onto its sensor or the film. With regard to focus, this means that larger regions in front of and beyond the defined **Focus Distance** will be rendered in focus.

You have to make sure that your scene's scaling and the defined distances are correct to ensure that the scene is rendered as a real camera would render the scene. Use the **Units** settings in the **Preferences** menu to adjust the scene's scale, if necessary (see **"DOF"** Project).



Below the **F-Stop** setting is the **Exposure** option. If enabled, you can manually define the exposure for rendering using the **ISO** value. You might be familiar with this setting from your own 35mm camera. The **ISO** (or ASA) setting defines the sensitivity of the film to light. Film with high **ISO** values can therefore be used for filming with weak or artificial light, or at sunset, for example. However, this generally accompanied by a rougher film grain. Hence, images will look correspondingly more grainy and less sharp as the **ISO** value increases. In Cinema 4D, however, the images will always be rendered noise-free, regardless of how high the **ISO** value is set. If you want to manually adjust the exposure and **ISO** value, remember that small shutter values let more light into the camera, which means that they can be used in conjunction with smaller **ISO** values. Generally speaking, the **F-Stop**, **ISO** and **Shutter Speed** settings are a good starting point for making your image generally brighter or darker. However, several test renders will generally be required until you achieve the look you want. If you don't want to make these adjustments manually, simply leave **Exposure** disabled. Cinema 4D will then leave the brightness exactly as it is defined by your scene's lighting.

Another component of image brightness is the length of time the **Shutter** remains open during exposure. This is defined using the **Shutter Speed** setting. This setting is also important for defining image brightness if the **Exposure** option is enabled. Longer **Shutter Speed** times will basically result in more light entering the camera and thus brighter images. A second component is added if motion blur should be rendered. If the lens remains open longer, more of an object's movement per frame can be captured. The blurring of animated objects, therefore, is also increased accordingly with higher **Shutter Speed** values.



Shutter Efficiency also plays a role for rendering motion blur. It simulates a degree of inertia or temporal transition until the shutter is completely open or closes again.

Smaller values will prevent the shutter from quickly closing or opening. A degree of sluggishness will be added. During this time, light will fall onto the virtual image sensor but with less than full intensity. When rendering moving objects, this means that they will be rendered with blurred edges. Larger **Shutter Efficiency** values will produce a quick opening and closing of the shutter. This will produce more defined edges at the beginning and end of the movement captured by the camera during exposure.

The available options can generally be configured in the following order:

- First, define the desired resolution in the **Render Settings** menu and then point the camera towards the object you want to render.
- If you want to use depth of field, define the **Focus Distance** to the region that should be in focus.
- Activate the **Physical Renderer** in the **Render Settings** to make the camera's **Physical** tab's settings available. If only one simulation of depth of field is desired, **ProRender** can also be used. When rendering with the **Standard** renderer, blurriness can also be created using a Post Effect but these must be configured manually and don't make a physically correct simulation available.
- Select an aperture that reflects the desired intensity of the depth of field. The effect can be intensified by increasing the degree to which the object fills the screen and reducing the **Shutter Speed**.
- If you want to render motion blur simultaneously, animate the object using keyframes and select the frame during the object's motion that you want to render. Prolong the shutter speed to amplify the motion blur effect. Faster shutter speeds produce correspondingly less motion blur.
- The image brightness can be adjusted using the **ISO** setting. To do so, the **Exposure** option must first be enabled. Otherwise Cinema 4D will calculate the image brightness automatically and will keep it constant, independent of the **F-Stop** and **Shutter Speed** settings.

Next, we will discuss the settings that are made available when the **Movie Camera** option is enabled. The first difference affects the film's sensitivity to light. **ISO** will be disabled and replaced by **Grain (db)** (**Exposure** option enabled). This principally has the same effect, i.e., the camera sensor's sensitivity to light. Values less than 0 will darken the image and values larger than 0 will brighten it correspondingly. The second difference you will notice is with the **Shutter Speed** settings. Previously, only the **Shutter Speed** setting was available but now it is grayed out and the **Shutter Angle** and **Shutter Offset** are available instead.

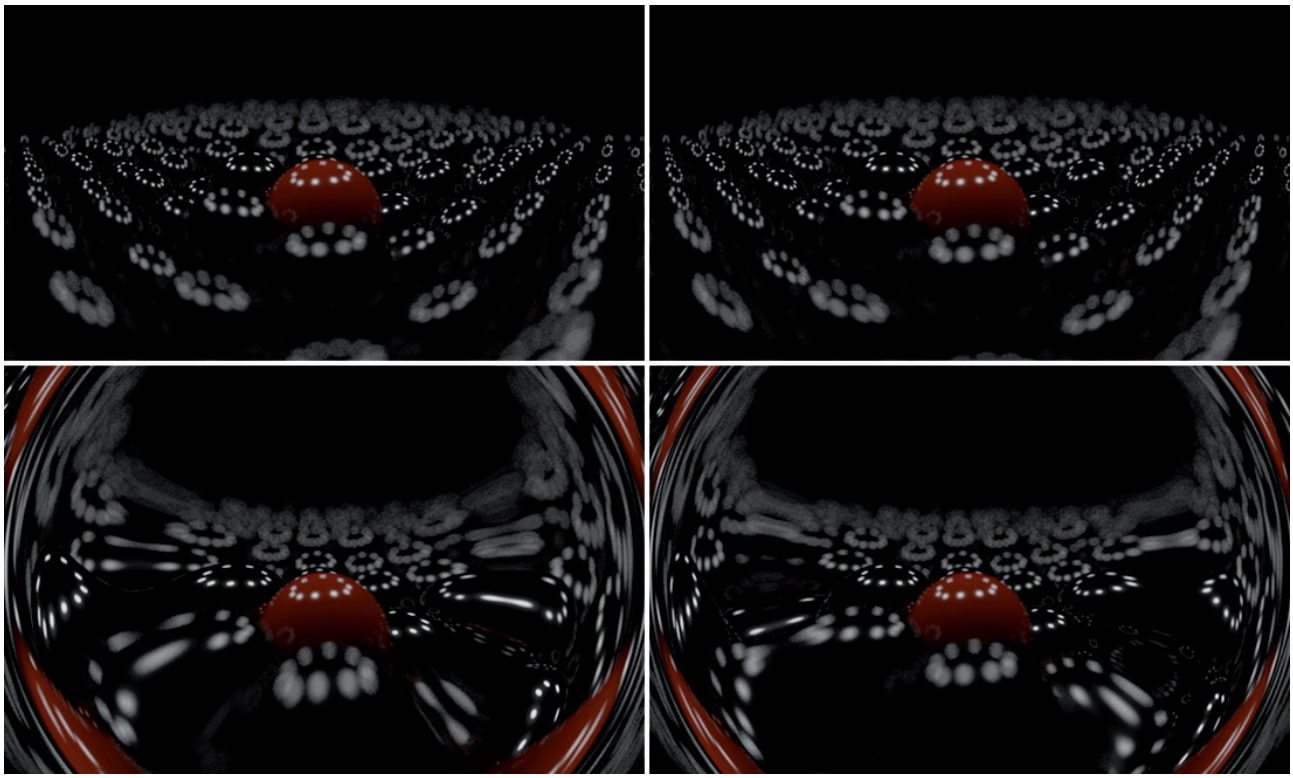
The **Shutter Angle** refers to the size of the rotating circular cutout in front of which the negative film to be exposed lies. This disc rotates in step with the animation's frame rate. A **Shutter Angle** of 360° will cause the exposure to run during the entire available time. For an animation with 25 fps (frames per second), this will mean a shutter speed of 1/25. If the **Shutter Angle** is set to 180°, a shutter speed of 1/50 will result for this frame rate.

As you already know, the exposure time as defined above affects more than just the image brightness. It also affects the intensity of motion blur. This is why Cinema 4D allows **Shutter Angles** greater than 360° to be used to achieve even longer exposure times when simulating extreme motion blur. The Shutter Offset value defines the (animation) time at which the shutter rotation should begin. Larger values will open the shutter correspondingly later. The rendered image's motion blur will show a correspondingly stronger weighting towards the object's position in the following animation frame.

When test rendering motion blur, note that motion blur will not be visible in Viewport renderings! You have to render to the Picture Viewer to see the effect. We will discuss how this is done later.

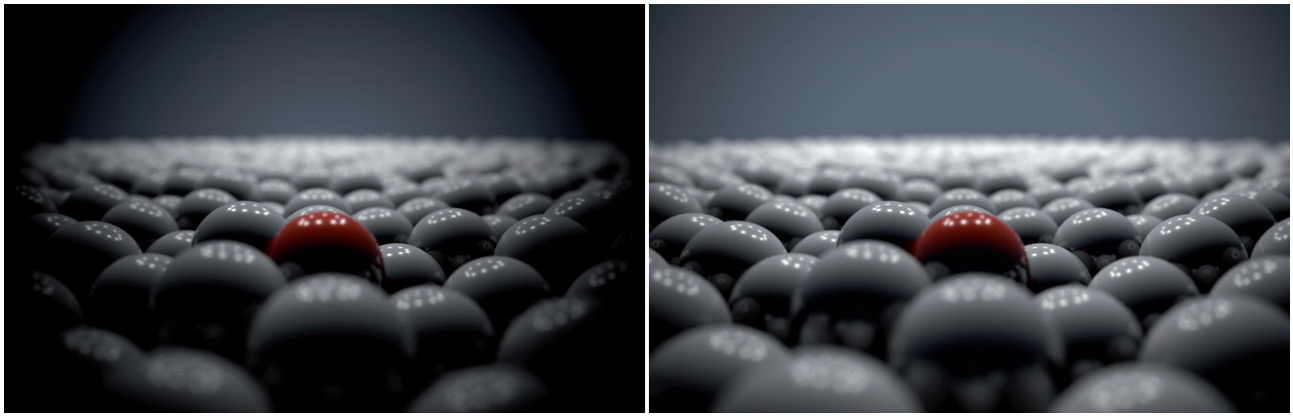
10.5.3 Lens Distortion

When using real lenses, aberrations can occur that lead to a curving of the motif. These phenomena most often occurs at the image's edge. Lenses with smaller focal lengths tend to cause barrel-shaped distortions. Telescope lenses with large focal lengths tend to cause circular distortions. **Lens Distortion** values in excess of 0% result in a barrel-shaped distortion; negative values produce circular distortions. It's up to you which value you decide to modify to achieve the distortion effect – **Lens Distortion – Quadratic** or **Lens Distortion – Cubic**. When using the **Cubic** method, the distortion begins at the image's center and only increases its intensity when it is very close to the edge of the image.



10.5.4 Vignetting Effect

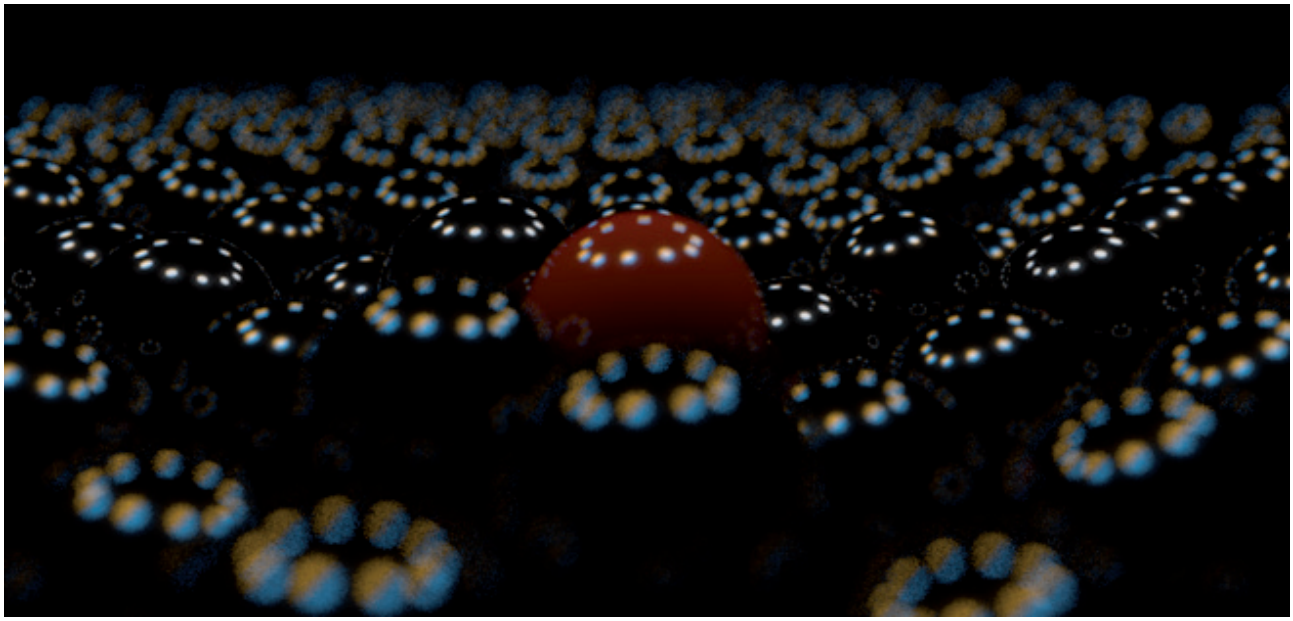
This effect is also used to simulate the shortcomings of a real-world lens system. Depending on the type and quality of the lens, the image's edge can end up being darkened.



The **Vignetting Intensity** value defines the brightness falloff between the center of the image and its edge. The **Vignetting Offset** value can be used to press this darkening effect from the image's center in the direction of its edges.

10.5.5 Chromatic Aberration and Bokeh Effect

Chromatic Aberration is the calculation of the different light wave lengths in the camera's lens system. This effect only occurs in regions that are not in focus, i.e., those that are blurred for rendering. Therefore, depth of field rendering must be used in conjunction with this effect. This effect causes the red and blue portions of high-contrast regions to be offset slightly in opposing directions – which can often be observed on specular highlights or at an object's outer edges.

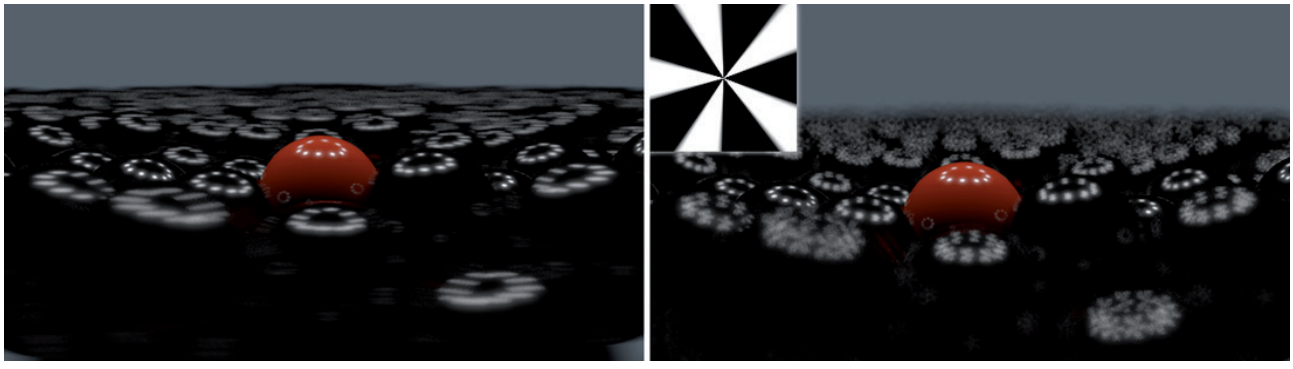


The direction of the color shift can be reversed by using negative values. The intensity of the color shift is also dependent upon the intensity of the blur.

This takes us to the **Bokeh** effect, which also requires the use of depth of field. This effect enlarges and partially reshapes blurred specular highlights. This primarily affects smaller and high-contrast image parts that lie in the blurred regions of the rendered image. If these specular highlight enlargements do not appear perfectly round, the **Diaphragm Shape** option can be enabled and it settings used to define a specific shutter shape using the **Blades** value. The shape created can also be rotated using the **Angle** value and distorted using the **Anisotropy** value. You should already be familiar with this principle from the shader section.

The round Bokeh effect would then be converted to an elliptical shape. Positive **Anisotropy** values will squeeze the shapes vertically and values less than 0% will squeeze them horizontally. Since the use of this effect requires activating the Diaphragm **Shape** option, setting the **Blades** value to 32 will produce the best circular **Bokeh** shape. The **Bias** value can be used to define the brightness gradient within the **Bokeh** shape. Positive values will produce more brightness at the center; negative values will make the center correspondingly darker and the edges brighter.

You can also load a custom shape for the **Bokeh** effect using the **Shader** setting.



The best results are achieved using black-and-white images. The white regions will be used to create the **Bokeh** shape.

10.5.6 Details Tab

Most of the settings in this tab have already been discussed in other sections. The **Show Cone** option makes the camera's cone visible in the Viewport. The **DOF Front** and **Rear Blur** settings were already explained in the depth of field section. These are important when using **Multi-Pass** to render depth of field and when rendering with the **Standard Renderer**. All that remains are the **Clipping** settings.

You are already familiar with the term 'clipping' from the Light objects section, in which clipping was described as the restriction of a property to a specific region. Here, the Clipping function works similarly – you can have the camera ignore surfaces or entire objects completely. This can often be useful, for example, if a camera that is placed outside of a building is used to render the building's interior. Clipping can be used to simply clip out the building's outer wall. The physical properties of the elements that are clipped out will, however, be maintained, i.e., these surfaces will continue to cast shadows, for example, even though they won't appear in the rendered image.

10.5.7 Spherical Camera

360° scenes are gaining more and more in popularity with the advancements in interactive VR. Contrary to images for print or normal videos, a complete panorama is shown. Using special players, the viewer can define where they want to look in the scene. This gives the impression that the viewer is actually in the 3D scene itself and a high-quality rendering will always be displayed. To achieve this effect, a special **Spherical** mode is available that can be enabled in the menu of the same name. If the **Equirectangular** option is selected, a complete spherically-shaped surface with the resulting rendering can be defined. This is the preferred mode for creating scenes in which viewers should be able to choose freely the direction from which they want to view the scene. The **Long Min** and **Long Max** settings can be used to restrict the horizontal angle of view. The **Lat Min** and **Lat Max** settings can be used to restrict the vertical angle of view for rendering. If these options have been used to restrict the angle of view, the **Use Full Range** option can be enabled, which will cause the restrictions to be ignored and a complete sphere will be rendered.

If you want to render a horizontal panorama that is restricted in the vertical view, set **FOV Helper** to **Dome**. Then, only a **Latitude** value will be available, which is set to 0° by default and ensures that only the camera system's top hemisphere will be rendered. Larger values will expand this dome correspondingly, also beneath the horizon, so that a latitude of 90° will result in the scene's entire environment being rendered. Hence, the only difference to the **Equirectangular** mode is that the restriction of the horizontal field of view is missing and the **Latitude** setting can only be adjusted in the lower region.

The rendering itself can use different schemes for sectioning the bitmap. These are available in the **Mapping** menu. Which one should be chosen depends on the software with which the resulting rendering should be displayed or edited. The **Lat-Long** option is popular for use within Cinema 4D as well as for numerous players for full-spherical panoramas, e.g., on social platforms or YouTube. The **Cube** options, which assemble bitmaps using rectangular tiles that are displayed in the direction of the camera along one of the main axes, are often used for game engines. Note that parts of the bitmap will remain unused f the field of view is restricted using the **Lat**, **Long** and **Latitude** values. In such cases, enable the **Fit Frame** option so the rendering is automatically stretch to match fill out render resolution. Also, make sure that the camera has no **Pitch** or **Banking** values defined before rendering. Otherwise the navigation in the rendered panorama would produce wobbly movements. Since the rendering will initially be a normal image you should define a render resolution that matches the selected mapping method. The width to height ratio should be 2:1 for **Lat-Long**, 4:3 for **Cube (cross)**, 6:1 for **Cube (string)** and 3:2 for **Cube (3x2)**.

SUMMARY: CAMERAS

- A newly created camera is automatically set to the view of the currently active Viewport.
- A camera must first be activated before it can be used for rendering. To do so, either click on the black icon next to the camera in the ***Object Manager*** or select it from the Viewport's **Cameras** menu.
- An active camera can be controlled in the Viewport using the Viewport's navigation icons or using the corresponding keyboard shortcuts.
- Cameras can be moved or rotated using the standard tools.
- The camera cone's aspect ratio reflects the defined render resolution. Therefore, the **Resolution** should be defined as early as possible in the **Render Settings** menu.
- The focal length defines the camera's aperture angle. A small **Focal Length** value simulates a wide-angle lens and amplifies perspective distortion accordingly; a large **Focal Length** value simulates a telescopic lens and reduces perspective distortion.
- The distance at which the focal plane lies can be defined manually. Alternatively, objects can be linked to a camera on which the camera can be made to focus automatically, which is more practical, for example, if the camera itself is animated.
- The intensity of the depth of field can be controlled either manually via defined distances or physically correct using the F-Stop setting. The latter is only available if the **Physical Renderer** is used.
- The image brightness can be affected using the **ISO**, **Shutter Speed** and **F-Stop** settings.
- The **Shutter Speed** setting can also be used to affect the motion blur of animated objects.
- Aberrations that occur with real-world lenses can be simulated using the **Vignetting**, **Chromatic Aberration** and **Lens Distortion** settings. The **Physical Renderer** must be used for these effects to be rendered.

11 Render Settings

We already discussed how **Output** size, resolution **Antialiasing** quality and the **Renderer**, including its basic **Options**, are defined in the **Render Settings** menu. Once you have fine-tuned your scene and checked it with test renderings, you need to define a **Save** path for your renderings. This is where the image or film will be saved when it's rendered.

11.1 Save Menu

If all you want to do is save a rendered image or animation, the **Save** menu's default settings should be sufficient. Click on the button with the three dots to select a directory to which to save your file(s). Don't forget to name your file. The file format is defined using the **Format** setting. Some formats offer additional settings via the **Options** button at the right.

Note that not all formats offer loss-free compression and not all available color depths are compatible with every **Format**.

As you know, animations are made up of a series of still images, which can therefore be saved sequentially. In many cases, this is better than saving the animation in QuickTime or AVI format because you have much more control over image quality when individual images are output. An animation will be saved as individual images if an image format is selected in the **Format** setting, e.g., TIFF or JPEG. Clicking on the small triangle next to the **Format** setting will make additional settings available with which you can adjust the compression among other things.

Use the **Name** setting to define the type of sequential numbering these images will be assigned. Make sure the naming convention is compatible with your post-production software – as a rule, though, you should have no conflicts with the available options.

The **Image Color Profile** can be added separately. Otherwise, the standard RGB and linear color profiles are available.

If the **Alpha Channel** option is enabled, a mask can be automatically included for rendering. If you want to render individual masks for individual objects, you must use **Multi-Pass** rendering.

The **Alpha Channel** can also be rendered as **Straight Alpha**. This works best when rendering objects in front of a black background and will cleanly mask transparent objects.

Normally, alpha channels will be saved together with the rendered image, which is not a problem for image formats such as TIFF or Adobe Photoshop, for example. However, several formats do not support such channels, or you might want to save the alpha channel as a separate file. If so, enable the **Separate Alpha** option.

To prevent visible banding, i.e., visible color transitions within gradients, which can occur when using low color depths, leave the **8-Bit Dithering** option enabled. If you render an animation with sound tracks, these can be integrated directly into the QuickTime or AVI files by enabling the **Include Sound** option. This option does not work in conjunction with still images or sequential images.

If an animation is rendered, which will subsequently be edited in Adobe After Effects, for example, it should be saved as a **Composition Project File**. If its **Save** option is enabled, the composition file will be saved in addition to the animation or image sequence. You can also save the file manually by clicking on the **Save Project File** button. This project file does not contain the images files themselves. It simply helps the **Target Application** load the images and films correctly into its timeline or sequence list. This is particularly helpful when using **Multi-Pass** rendering, which can generate a great number of image sequences or films, which can only be combined correctly when they are in the right order and with the right layer modes.

If **Relative** is enabled, the sequences in the **Target Application** will be arranged as they are in the Cinema 4D **Timeline**. Otherwise each sequence will always begin at the start of the **Target Application's** timeline.

If you use markers on the Cinema 4D **Timeline**, their position will be assumed in the target application if the **Include Timeline Marker** option is enabled.

The **Export to After Effects** option was already discussed in the light sources section, and the **Camera** objects also offer this function. If the **Include 3D Data** option is enabled, the corresponding light sources and cameras will also be included. Other objects can also be included in the composition file, even if only as **Null** objects, i.e., coordinates, or as simple color planes. To do so, assign an **External Compositing** tag from the **Tags**/**Render Tags** menu to the objects in question in the ***Object Manager***.

If you select **Nuke** as the **Target Application**, the **Save FBX File** option must also be enabled to allow 3D files to be converted. A separate file in FBX file format will be output.

11.1.1 External Compositing Tag

This tag makes additional options available for external compositing. If the **Children** option is enabled, all **Child** objects (previously referred to in this curriculum as sub-objects) in the ***Object Manager*** will also be included in the compositing file.

The **Cache** option only applies to Generators, which, internally, generate a certain number of copies. If this option is enabled, tracks will also be created for these objects in the **Target Application**.

The **Anchor Point** setting defines at which location on the object, e.g., in After Effects, the anchor point should lie. Only the Center option will allow the corresponding **Plane**, for example, to be rotated around its center point.

If **Solid** is enabled, a 2D plane will be created around the anchor point whose **X** and **Y** size can be defined using the respective values. A custom **Color** can also be defined for the plane.

11.2 Multi-Pass Rendering

Whenever additional image layers or alpha masks are required in addition to the image itself, you should enable the **Multi-Pass** function in the **Render Settings** menu's left column. Note that this is not available when using the **OpenGL** renderer. ProRender also uses its own multi-pass system, which is configured via its own render settings menu. Enabling **Multi-Pass** will make additional settings available on the **Save** menu such as a **Save** path, image **Format** and color **Depth**. You can use the **Multi-Pass** function's **Save** path in addition to or in place of the **Save** menu's path. **Multi-Pass Files** will save the previously separately selected image layers to a single file. Of course this only works if an image **Format** is selected that supports layers and channels. If you would rather save the layers as separate files, enabling the **Layer Name as Suffix** option will name the files to reflect the content of the respective image layer. If a **Multi-Pass File** also contains an alpha channel, it can also be rendered as **Straight Alpha**, if this option is enabled. This is particularly beneficial when rendering transparent objects or effects such as fog or clouds. This option can only be enabled if the **Alpha** option in the **Save** menu's **Regular Image** menu is also enabled.

Note that all layers must be arranged correctly to recreate the original image. The layers will use additive and multiplying modes, and the **Linear Workflow** option, which is enabled by default in the **Project Settings** menu, will generate very slight brightness nuances. A color depth of at least 16 bits per channel (the default **Depth** value) should be defined for **Multi-Pass** images to prevent banding or imprecise results.

11.2.1 Selecting Multi-Pass Layers

After **Multi-Pass** has been enabled and the **Save** menu's settings have been defined, click on the **Multi-Pass** function in the **Render Settings'** left column. Here you can define how lights will be handled. You can select from **All**, **None** or **Selected** to be rendered as separate layers. If **Selected** is chosen, lights whose **Separate Pass** option is enabled in the **General** tab's settings will be rendered as separate layers. The **Mode** menu defines how many layers will be created per light source. **Diffuse** (material color), **Specular** and **Shadow** can be rendered separately or together. This makes it possible to correct the effect of individual lights or even disable lights completely in Photoshop, for example, without having to re-render the image(s).

When light and shadow are saved separately, seams can be generated at the edges of shadows during anti-aliasing calculation. This effect can be reduced by enabling the **Shadow Correction** option.

This is only a small portion of the number of layers that can be saved. The entire spectrum of **Multi-Pass** layers can be seen by clicking on the **Multi-Pass** button at the bottom of the **Render Settings'** left column.

The available options can be roughly divided into **image** channels, **material** channels and **effect** channels. **Image** channels affect individual properties of a rendered image such as visible specular highlights or shadows. The **material** channels only display the content of individual material channels such as the surface color or the **Texture** that is loaded into the **Reflectance** channel. This is not what is reflected in the object's surface but rather the texture or property that is defined in the material's **Reflectance** channel. The **effect** channels can, for example, contain the **Depth** as seen from the camera's angle of view or the **Motion Vector** of animated objects. In the case of the very useful **Depth Map**, this can be configured via the camera's **Details** menu. The blur effects have already been discussed in previous sections.

Each time you make a selection from the **Multi-Pass** list, a layer is added that is required in the saved file. Of course only those layers should be added that are actually needed so the file isn't larger than it has to be. This is particularly important when saving animations.

It's also very important that all required layers are added to make sure that no image property is missing in the saved image(s). This is what the **Blend Channel** is for. Here you can activate all image properties that are required for subsequent compositing. Those layers that are not activated must be created separately so the image is complete. For example, if **Specular** and **Shadow** are omitted in the **Blend Channel**, these must be added manually by selecting them from the **Multi-Pass** list.

Otherwise, the complete image, including alpha channel, can be selected by adding the **RGBA Image** option.

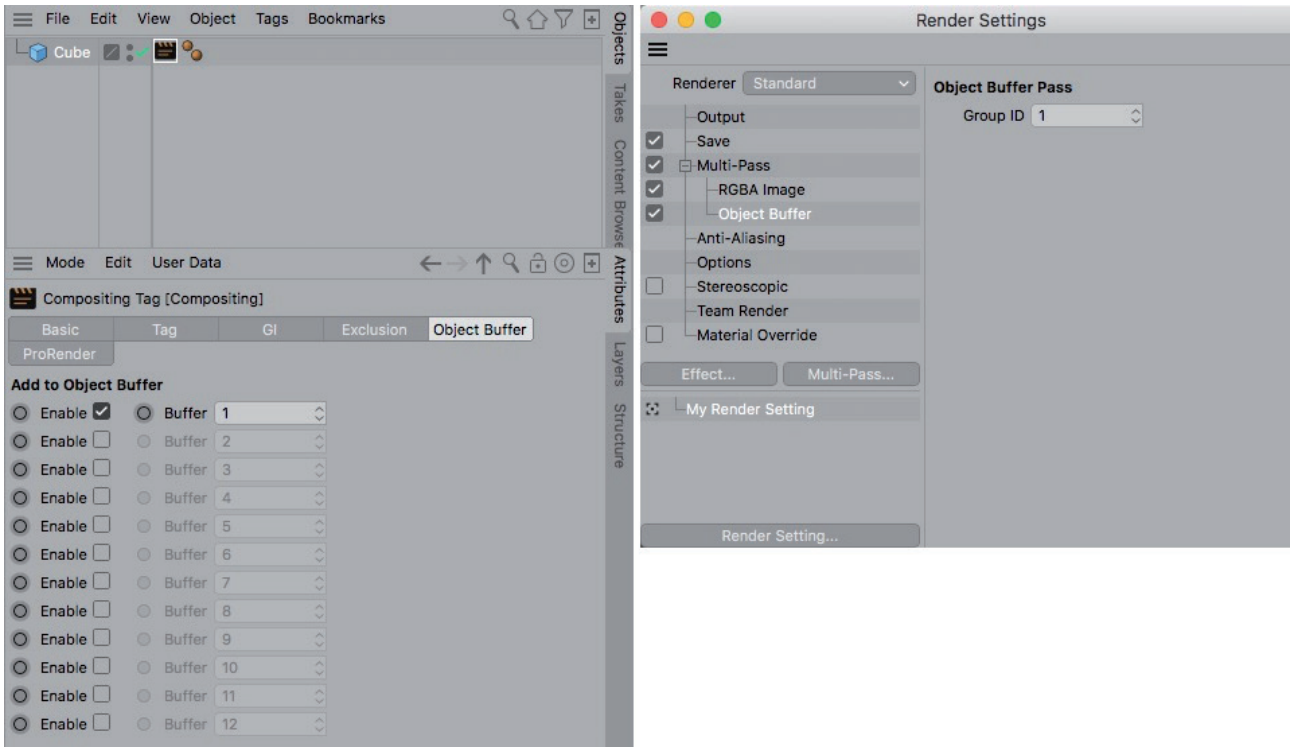
If you use physical materials for which the surface color and the specular and reflective properties are simulated vial the Reflectance channel, special multi-passes are available that allow these properties to be output separately as well. In this case it makes sense to set up all materials in the scene as physical materials, i.e., not in the **Color** channel but instead to use diffuse Reflectance layers for the color shading of surfaces. For this, the Reflectance layers **Lambert (Diffuse)** and **Oren-Nayer (Diffuse)** can be used. The results of these Reflectance layers can be output using multi-passes for **Diffuse Direct** and **Diffuse Indirect**. The effects of the **Beckmann**, **GGX**, **Phong** and **Ward** Reflectance channels are on the other hand output via the multi-passes for **Direct Reflection** and **Indirect Reflection**.

The **direct** multi-passes each show the direct effect of the HDRI environement, physical sky and the surfaces visible in reflective surfaces. These are also objects that are responsible for the direct illumination of objects. The **indirect** multi-passes contain the reflections of the remaining scene objects.

Object Buffers must be added if individual objects or object groups should be assigned an alpha mask. An **Object Buffer** can also be added from the list of **Multi-Pass** options. Any number of object buffers can be added and assigned different **Group IDs**. These **Group IDs**, which can consist of any unique digit, must then be assigned to the respective objects. This is done using **Compositing** tags, which can be added from the **Cinema 4D Tags** menu in the *Object Manager*.

11.3 Compositing Tag

In this tag's **Object Buffer** tab you will find a list of **Buffers** that you can simply enable or disable, as needed. As you probably already assumed, these numbers correspond to the **Group ID** numbers in the **Multi-Pass** layer. You are not restricted to using the default numbers (1 – 12) – you can enter higher numbers, if necessary.



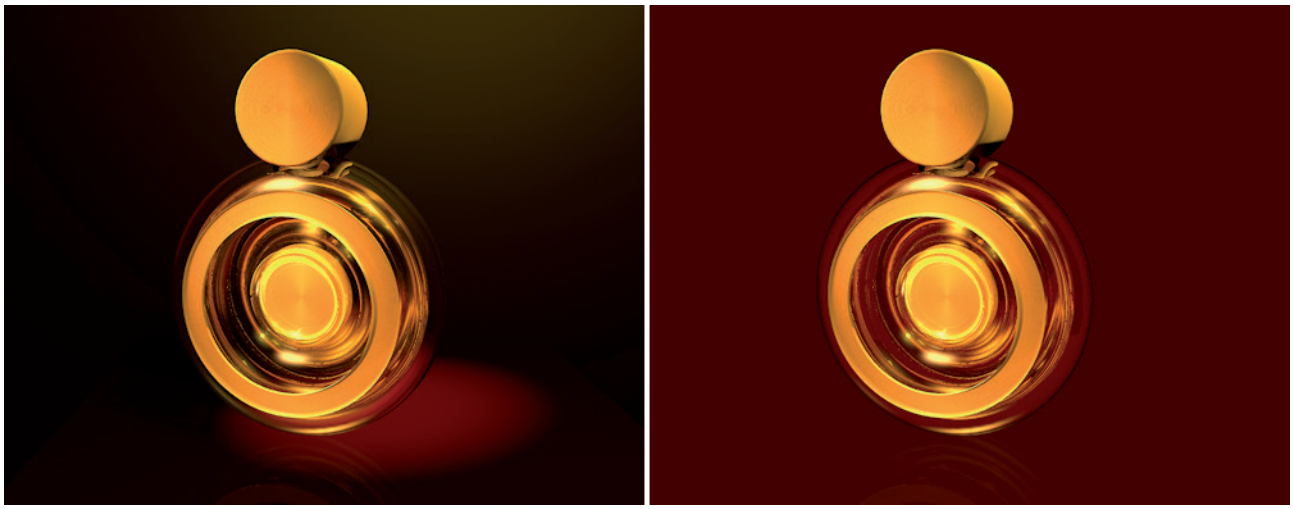
Multiple channels can also be enabled simultaneously. The respective object can then, for example, be made to appear through several different alpha masks. Also note that the **Compositing** tag's **Buffer** settings will automatically affect all of a given object's **Child** objects in its hierarchy. This is only one way in which this very useful tag can be implemented. In the following we will take a look at the tag's other settings.

11.3.1 Tag Tab

Here you will find options that let you fundamentally define the object's properties for rendering. Options such as **Cast Shadows**, **Receive Shadow** and **Seen by Camera** should be self-explanatory. If **Seen by Transparency** is enabled, the object will be visible behind alpha materials and transparent materials. If a transparent also has a **Refraction** value not equal to 1.0 in the **Transparency** channel, **Visible for Refraction** must be enabled to the object is visible behind a refractive sphere, for example. This object can only be seen in a reflective surface if **Seen by Reflection** is enabled. **Seen by AO** refers to the rendering of **Ambient Occlusion**. The object will only be seen by **Ambient Occlusion** if this option is enabled, which can result in a material being darkened.

The **Seen by Refraction**, **Reflection** and **AO** options can all be enabled or disabled simultaneously using the **Seen by Rays** option. The **Self Shadowing** option is only available if both **Cast Shadows** and **Receive Shadows** are enabled. If **Self Shadowing** is disabled, the object will not be able to cast shadows onto itself. Other objects will not be affected, and this object will still cast shadows onto other surfaces.

The **Compositing Background** option is also very interesting. It prevents any light shading on the surface. The color or texture defined in the material's **Color** channel will be rendered in its original color and will not be affected by the light. This can, for example, be used to render an element on a neutral background that can still receive shadows.



A variation of this is the **Compositing Background for HDR Maps** option. This is designed for scenes that are surrounded by a HDR texture and illuminated with **Global Illumination**. Often, this requires a floor plane on which shadows can be cast. If the Floor object hast a **Compositing** tag assigned to with **Compositing Background for HDR Maps** enabled, the floor will remain hidden for rendering but will still show the cast shadows. This option has no effect if the image is rendered without **GI**.

Visible for GI only plays a role if the image is rendered with **Global Illumination**. If enabled, this object can be illuminated by **Global Illumination** or itself forward diffused light in accordance with its **Illumination** settings.

If **Antialiasing** is set to **Best** in the **Render Settings**, the anti-aliasing options at the bottom of the **Tag** tab will be made available, which can be used to define **Min**, **Max** and **Threshold** values individually for the anti-aliasing function. This is very useful for optimizing render time because only the edges of correspondingly complex objects will be smoothed. All other objects will continue to be rendered using the **Anti-Aliasing** setting defined in the **Render Settings** menu. Note that the **Anti-Aliasing** defined in the **Render Settings** will be the minimum that will be used. The **Compositing** tag's settings will only be used if they exceed those of the **Render Settings** menu. The **Matte Object's** color is only designed for use with extreme compositing tasks. The object will be displayed with the **Color** defined. The object will no longer be able to receive shadows but it will still be reflected in reflective surfaces. This can be useful if you want to position an object with the same shape at exactly this position in the final rendering.

11.3.2 GI Tab

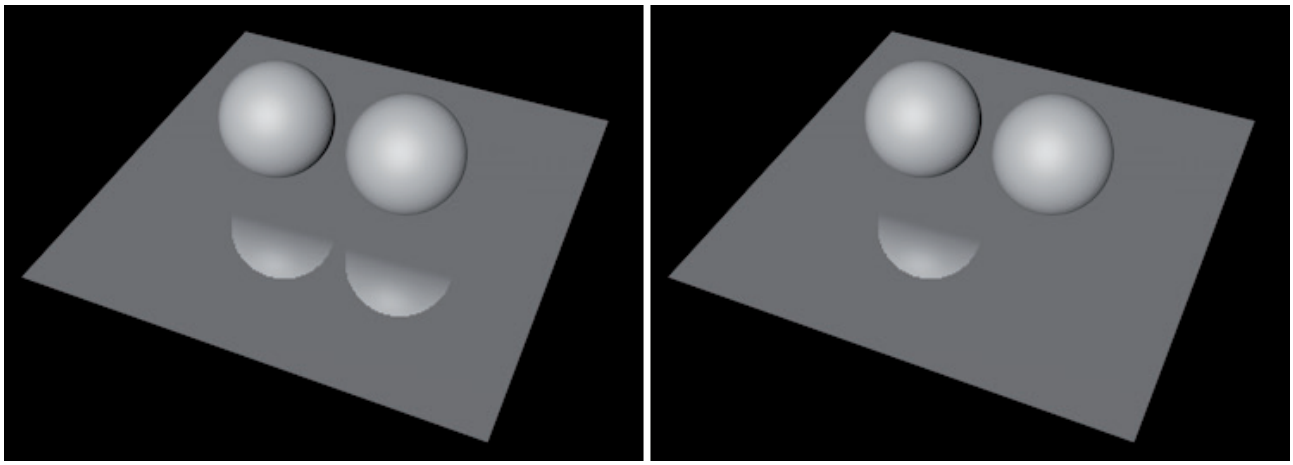
This menu only affects the object's **GI** properties. If the **Enable GI Parameters** option is enabled, the **Intensity** of the object's generating as well as its receiving properties can be defined in percent. If **Irradiance Cache** is used for rendering, the **Record Density** and **Stochastic Samples Ratio** settings will be decisive for the quality of the diffused lighting. We will discuss this in detail later. The **Record Density** value multiplies the number of shading points on the object. The samples per shading point will in turn be multiplied with the **Stochastic Samples Ratio** value. Since all values can be set to exceed 100%, the **GI** render quality can be defined selectively per object.

Forced QMC Sampling will use the **QMC** method to precisely sample an object if **Irradiance Cache** is used for rendering, which makes it possible to render highly detailed **GI** caustics and shadows on this object. A different algorithm is used for objects rendered with GI that have transparencies. Generally speaking, objects with transparencies benefit less from GI than other objects. This is why their accuracy is automatically reduced. A Compositing tag with Enable Transparency Overlook enabled can be assigned to objects if the difference between shadows of neighboring objects with little and with a lot of transparencies is too stark. The transparent object will then be sampled with the same precision as if it were opaque.

Interpolation Grouping works similarly but is used to separate the GI sampling of neighboring surfaces, which prevents overlapping GI sampling from occurring. As a rule, however, this option can be left disabled because a more homogenous shading, even across object borders, will be generated without it.

11.3.3 Exclusion Tab

If an object should be reflected in all other objects except for the **Floor** object, this might be a problem. However, if the **Floor** object is dragged into the **Exclude** tab's list and the list is set to **Exclude**, the object would be excluded from being reflected in the floor. The **Compositing** tag only lets objects be generally excluded from reflections and transparencies, for example. This is what the **Exclusion** tab is for. If the Floor object is dragged into this tab's field, it can be included or excluded from the reflection.



This is not restricted to reflections. The icons next to each item can be enabled or disabled individually. The left icon affects **Transparency**. If the floor were transparent, we would no longer be able to see our object if it were behind it. The middle icon affects **Refraction**. If the floor were transparent with a **Refraction** value not equal to 1.0 our object would not be visible here, either. The icon on the right affects the object hierarchy. If enabled, the exclusion would be applied to all of the **Floor** object's **Child** objects in the ***Object Manager's*** hierarchy.

SUMMARY: RENDER TAG

- If the rendered image or animation should be saved, the corresponding **Save** path must be defined in the **Render Settings** menu.
- Internally, Cinema 4D renders all images at 32-bit color depth. The color depth of a rendered image can be defined individually.
- The file format can be selected from an available list. However, not all available formats support higher color depths or additional layers or channels.
- Animations can also be rendered as sequential images instead of QuickTime or AVI movies.
- If an image should be saved with different layers, e.g., for materials or effects, the **Multi-Pass** function should be used. The **Save** path must then be defined separately in the **Multi-Pass** function's settings.
- **Multi-Passes** are also available when rendering with ProRender. These can be found in ProRender's own multi-pass menu in its render settings.
- The Buffers selected must be enough to properly recreate the image when rendered. The **Blend Channel** is recommended for ensuring that this is done. The Buffers omitted here must be added manually.
- If individual alpha channels should be rendered for specific objects or object groups, these must first be assigned a **Compositing** tag, which lets you enter specific Buffer numbers.
- These numbers can be integrated into **Multi-Pass** rendering using the **Object Buffers**.
- **Compositing** tags can also be used to exclude or include objects from shadows, reflections or transparencies, for example.
- An **Include/Exclude** list also makes it possible to define in which surface an object should be reflected or behind which transparency it should be visible.
- The intensity of anti-aliasing and **GI** can also be defined individually.
- Composition files can be created for use in external compositing applications with which an animation and its assets, including 3D objects, lights and cameras, can be loaded automatically.

11.4 Special Render Effects

Additional effects such as **Hair**/**Grass**, lens effects or **Global Illumination** and **Caustics** can be added by clicking on the **Effects** button and selecting them from the menu that appears. Several of these effects will be activated automatically, like the previously discussed Hair or Lens effects. Others can be added manually as needed. Which effects are made available also depends on the selected renderer. The largest number of effects is available with the **Standard** or **Physical** renderers. With ProRender, similar effects can be configured directly in its render settings, such as Global Illumination or Ambient Occlusion. An effect can be easily removed by right-clicking on it and selecting **Remove**. In the following we will only discuss some of the most important effects.

11.4.1 Ambient Occlusion

This effect offers basically the same settings as the shader of the same name, which is why we will not rehash these here. The primary differences to the shader are:

- The **Ambient Occlusion** effect automatically influences all objects in the scene if the **Apply to Project** option is enabled.
- If **Apply to Project** is disabled, **AO** will be calculated but will not be visible in the image. This bears the advantage that this effect can be saved separately on an **Ambient Occlusion Multi-Pass layer**.
- The Multi-Pass effect can be rendered as a Cache file. This will often speed up the calculation of the AO effect, which is useful if several test renderings have to be made. The cache must only be calculated once if no objects or camera angles are subsequently modified. The AO effect is not dependent on the scene's lighting and can therefore be created prior to setting up lighting (**"Room_AO"** Project).

11.4.1.1 Ambient Occlusion Cache Tab

A cache is a file that contains the results of a given calculation. If no properties are modified in a Project that would require the cache to be recalculated, this file can be used over and over to create a given effect, which can dramatically reduce render times. If the **Enable Cache** option is enabled, the **Ambient Occlusion Cache** will be calculated as a **Pre-Pass**, which essentially determines where throughout the scene measuring points can be placed.

This method is less precise compared to calculating the effect for each pixel individually but this does not play a role for **AO**. To the contrary: The effect contains slightly more noise and the number of calculation steps must often be increased to remove this added noise. Of course this will also result in longer render times. Using a **Cache** can avoid this because the locations between the measuring points will be interpolated and thus blurred. The result not only looks nice and soft but can also be rendered faster.

The precision of the **Cache** calculation is affected by two settings: **Record Density** and **Samples**.

The **Record Density** setting defines the density and dispersion of the measuring points, which are used as a reference for the **AO** effect. The **Samples** value defines the number of measurements that will be made from each **Record Density** measuring point. The more samples that are made, the more precise a measuring point's location can be determined. The **Record Density** setting offers **Custom**, **Preview**, **Low**, **Medium** and High options, which all have numeric values that can be viewed and edited by clicking on the black triangle next to the setting to make these settings available.

Min Rate and **Max Rate** define the number of passes during which the measuring points throughout the scene are distributed. The measuring points will not be distributed evenly. Fewer will be distributed across simple surfaces than in corners of a room or where surfaces lie close together, for example. This procedure ensures that more samples are calculated at locations at which a visible **AO** effect can be expected.

The **Min** and **Max Rate** numeric values represent pixel sizes that will be used for a pass. A value of -3, for example, means that the image pixels will be 8 times larger than normal and the distribution of measuring points will be correspondingly loose. If set to -2, the pixels will be 4 times as large. If set to -1, the pixels will be twice as large as normal. A value of 0 represents the pixels' normal size. Values larger than 0 will invert the effect and the pixels will be subdivided into sub-pixels, as is the case with the anti-aliasing effect.

The difference between the **Min** and **Max Rate** values defines the number of passes that will be used to distribute the measuring points. Passes with larger image pixels will be faster but less precise as those with smaller pixels. These values give you indirect control over the precision with which the effect will be rendered as well as how long it will take to render.

The density of the measuring points can generally be defined using the **Density** value, and even more precisely using the **Minimum** and **Maximum Spacing** values. The larger the percentage values for **Minimum** and **Maximum Spacing**, the larger the spaces between the measuring points can be – and the less precise and softer the result will be. **Minimum Spacing** is applied to regions with a high density of measuring points, e.g., room corners, etc. **Maximum Spacing** is applied to simple surfaces or layers without detail.

The **Density** value acts counteractively. Larger values will produce more overall measuring points in relation to what is defined by the **Minimum** and **Maximum Spacing** values.

The **Smoothing** value affects the interpolation between the measured values. Since the measuring points only ascertained data selectively, the regions between the measuring points must be interpolated. The larger the **Smoothing** value, the larger the regions will be and the more measuring points that will be included in the process. This will, of course, cause some details to get lost. Therefore, the **Smoothing** value should not be set too high to ensure that enough detail can be measured from the measuring points.

If **Screen Scale** is enabled, the calculation will be made according to the rendering's actual screen resolution. This is the recommended method, which is why this option is enabled by default. If this option is disabled, the same number of measuring points will always be calculated, regardless of the screen's resolution. This is generally not recommended because the settings cannot be scaled together with the screen resolution.

The **Cache File** menu is used to define what should be done with the calculated results. If **Auto Save** is enabled, the **Cache** file will automatically be saved to your scene's project directory after the image has been rendered. The file will normally be saved to a separate folder named '**illum'**. If you want to save the file to a different location enable **Custom Location** in the **Cache File Location** menu and select a **Location**.

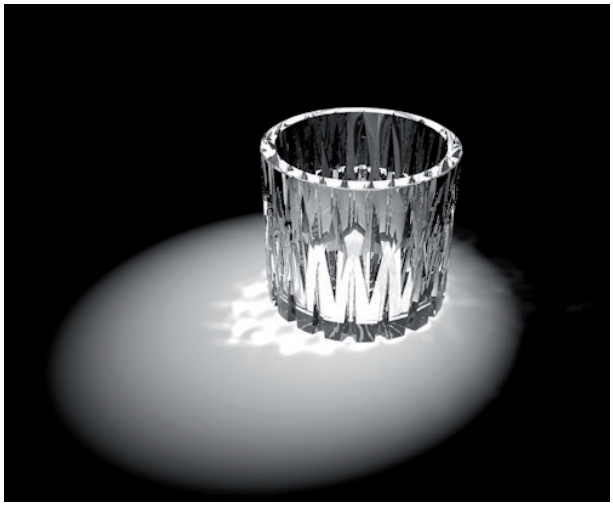
If you want to use an existing **Cache** file, enable the **Auto Load** option. If no matching file is found, a new **Cache** file will automatically be created. In any case, Cinema 4D will check if there is an available **Cache** file for the scene or the view that can be used. This procedure will generally require some time, which is why the **Skip Prepass (if present)** option can be enabled to skip this process. You should, however, be sure that the information saved in the **Cache** file is valid for the current scene – it should not contain any objects that have since been modified, moved or deleted, including cameras. Lights and materials on the other hand have no effect on the **AO** calculation and modifying them will not affect the usability of the Cache for the scene. The only exception are transparent materials, which can, of course, affect the intensity of the **Ambient Occlusion** effect.

When rendering animations, a separate **Cache** file will be created for each frame if objects or the camera move during the animation. If the **Full Animation Mode** option is enabled, sequentially numbered **Cache** files will automatically be created for each frame of animation. If **Team Render** is used to render, each active client computer will create its own **Cache** file(s). We will discuss this in detail later.

Click on the **Flush Cache(s)** button to delete any saved **Cache** files from the target directory. This means that the **Cache** will definitely have to be recalculated.

11.4.2 Caustics

You already know how **Caustics** is activated in the **Light** objects' settings and how it is evaluated via the material's **Illumination** channel's settings. However, **Caustics** will only be rendered if it is also added as an effect in the **Render Settings** menu (**"CausticsExample2"** Project).



This effect's settings largely reflect those of the material and the **Light** object. Enabling the **Volume Caustics** option will make the following three settings available. **Step Size** defines the distance between samples. Shorter distances will increase render times accordingly and produce more detailed results. The **Sample Radius** value defines the distance within which neighboring samples will be interpolated. Larger distances will produce correspondingly more blurred results but can also result in details being lost. The number of **Samples** is the maximum number of samples that can be allocated within the **Sample Radius**. Higher **Samples** values will increase quality and render time.

Several test renderings are generally necessary to determine the correct Caustic effect. If **Save Solution** is enabled, the current **Caustics** calculation will be saved to a separate file. Cinema 4D will create a folder named '**illum'** in the project directory – as was already mentioned for the **AO Cache**.

If such a file already exists, you can use the **Recompute** setting to define how these files should be handled. If **Frist Time** is selected, Cinema 4D will automatically search for an existing **Solution** that it can use. However, any modifications made to your scene will not be taken into consideration. Only the saved Solution will be used. If this Solution is not available, a completely new calculation will be made. This is also the case if Always is selected, which is why these modes are well suited for making test renders – Caustics will always be recomputed. If **Never** is selected, a **Caustics** file must be available. If no file is found, the rendering will be terminated with an error message. Enabling the **Single Animation Solution** option can save a lot of render time. A complete Caustics file will only be created for the first frame of animation. This option will only work if the camera is the only element animated in the scene.

11.4.3 Global Illumination

We've already discussed this effect quite often, so you already know that it creates a natural-looking reflection of light onto surfaces or it can be used to create light sources using luminous material properties. There are several methods for rendering **GI**, which can also be combined. This depends primarily on the **Diffuse Depth** that you want to simulate. The **Diffuse Depth** is the maximum number of light reflections that you want rendered. The more you render, the longer the process will take and the softer, more natural and brighter the rendering will be. Even if your scene is lit well enough using normal light sources, you might still want to improve the result using **Global Illumination**. In such cases, a **Sample Accuracy** value of 1 using the **Primary Method** will be sufficient. The **Secondary Method**, which is responsible for all secondary reflections, can then be set to **None** (**"Room_GI"** Project).

Note that the calculation of Global Illumination when using Physical materials is not necessary. The exchange of light between surfaces will be handled automatically by the material's reflectivity.

11.4.3.1 Primary Method

Even using only a single method, i.e., the **Primary Method**, can improve the result. Basically, there are two methods available: **Irradiance Cache** (short: **IR**) and **Quasi Monte Carlo** (short: **QMC**). As the name suggests, the **IR** method is an adaptive process, which generates a **Cache** file. The principle is very similar to the previously discussed **Ambient Occlusion** effect, only that light is measured and not distances. The **QMC** method on the other hand is a brute force process that calculates each image pixel with equal precision without using a pre-pass or an approximation about where and how intensively sampling should take place.

Mathematically speaking, the **QMC** method is the most precise method offered by Cinema 4D. It also takes the longest to render. However, it can help avoid fluctuations in brightness that can occur if the adaptive **IR** method is used, which is caused by the varying distribution of measuring points from frame to frame.

The **IR** method's **Record Density** settings can be found in the **Irradiance Cache** tab and are almost identical to the **Ambient Occlusion** settings. This is why we won't bother with another explanation here. The only additional setting is the **Color Refinement** value, which can be used to add measuring points at locations at which abrupt brightness or color transitions occur, or where stark contrasts can be found. Cast shadows would be a good example.

The **Samples** settings, which should also be familiar to you, are located in the **General** tab and are used to locate light in the scene. Here, you can also select from **Low**, **Medium** and **High** options, which define the **Accuracy** value – which is made visible by clicking on the small black triangle next to the **Samples** setting. The **Sample** count is not fixed but is ascertained during the pre-pass. The **Accuracy** value refers to sample count that is ascertained. The pre-calculation of samples can be sped up by defining a custom **Sample Count**. The scene should be rendered in order to ascertain a realistic number of samples. If the **Show Global Illumination Info in Console** option is enabled in the Cinema 4D **Preferences** menu's **Renderer** settings, you can use the estimated sample count displayed in the **Console** window. Of course the render quality should be in line with what you want. The **Console** window can be opened by selecting it from Cinema 4D's main **Script** menu.

Because each sample ray costs valuable render time, you should try to make sure that only those regions of the image in which light appears are sampled. This can be ascertained in some cases, e.g., if a **Physical Sky** or **Sky** object with an HDR image in the **Luminance** channel are used, or if luminous materials are used on object surfaces. Both sky types will be sampled if the **Discrete Sky Sampling** option is enabled. **Discreet Area Sampling** affects luminous materials that function as **Area** lights using **Global Illumination**, hence the name **Area Sampling**.

Clicking on the black triangles next to these options will make additional settings available that can be used to adjust the sampling accuracy. Enabling the **Force Per Pixel** option will create samples for each image pixel that emit in the direction of the **Sky** or **Area** light. For example, this is the only way that a comparatively very small **Area** light can be accurately sampled – such as a luminous sphere that lies inside a cube that only has a small crack opening on one side.

Force Per Pixel must be enabled to ensure that the sphere's light can be seen through the crack and not the luminous sphere itself. The render time will be correspondingly longer.

Enable the **Custom Count** option if you want to define a **Sample Count** other than the default values for the skies or the **Area** light.

Of course, not only light sources should be searched for. Surfaces can also reflect light. Such random reflections of light can only be captured if the **Hemispherical Sampling** option is enabled, which is why it is enabled by default.

The **Cache Files** tab's settings should also look familiar to you. The only option that wasn't previously discussed is the **Prepass Only** option. If enabled, the rendering of the image will be suppressed and only the **Cache** file will be saved (if **Auto Save** is enabled). This can, for example, be used to create a **Cache** file for an animation and subsequently use a smaller image resolution for the final rendering. This will speed up rendering accordingly and the quality will be good enough, even if the resolution is subsequently increased and the **Auto Load** function is used to apply the "small" **Cache**.

The brightness, color saturation and contrast of the **GI** effect can also be adjusted using the **Primary Method**'s **Intensity**, **Saturation** and **Gamma** values, respectively, without affecting render time. The **Saturation** value affects the light reflected by surfaces and the light created directly by luminous materials. The **Gamma** value affects the contrast between bright and dark regions. **Gamma** values greater then 1 will brighten the overall **GI** light effect. The **Primary** and **Secondary Methods'** Intensity value work similarly but only work as a multiplier for the brightness of the respective method's light reflection.

The **Options** tab contains general **GI** settings that, for the most part, work independently of the selected **GI** method. The **Debug information Level** is information about the course of the **GI** rendering and can be saved in an abbreviated (**Minimal**) or comprehensive (**Complete**) format. A corresponding text file will be saved in the scene's project directory. This file can, for example, be used when making support inquiries with Maxon. Otherwise, these files will not be of any importance to you.

The **Glass/Mirror Optimization** setting only applies to surfaces with reflective or transparent properties. As was already mentioned in the **Render** tag section, materials with strong reflective or transparent properties do not benefit additionally from **Global Illumination** because their actual surface color is so minimal. Surfaces whose reflection or transparency are stronger than the defined **Glass/Mirror Optimization** value will not be affected by **GI**, which will speed up rendering of these objects.

We also already discussed **Caustics**, albeit the type that is created directly by light sources. **Global Illumination** can also be used to create **Caustics** for diffused light. However, very intensely luminous materials or HDR images with stark contrast are needed. The **QMC** method generally produces more precise **Caustics** but the **IR** method can also be used. The **Refractive** and **Reflective Caustics** options define which material properties will be used to generate **GI Caustics**.

To check the global illumination quality you can omit the effect of real light sources in the rendered image. If **Diffuse Illumination Only** is enabled, only the light emitted by surfaces and generated by reflections will be rendered. Make sure this setting is disabled before creating your final render. If **Hide Prepass** is enabled, the depiction of the render steps prior to the actual rendering will be suppressed. This only saves a minimal amount of time and can therefore stay enabled since it gives you the opportunity to observe the render process and terminate it if you notice any errors during the process. If Show Samples is enabled, the colored pixels that indicate the number and distribution of measuring points during pre-pass will be displayed. If you don't want to display these pixels, simply disable this option. This option has no affect on render time or on the image quality.

This completes the description of the **Irradiance Cache** settings. But what if you want to use the more precise but slower **Quasi Monte Carlo** method? If this method is selected, most of the previously described settings will not be made available because the **QMC** method requires no **Prepass** and will thoroughly render each individual pixel. This is why only the **Samples** setting can be used to define the number of samples per pixel. The remaining settings are the same as with the **IR** method.

The **Irradiance Cache (Legacy)** method is also available and it is only designed to ensure compatibility with Cinema 4D versions prior to R15 and should not be used when creating a new scene in R15 (or above).

The new **IR** method is not only faster but also approaches the quality of the **QMC** method and is compatible with **Team Render**. We will discuss this in detail later.

11.4.3.2 Secondary Method

Whenever you have a scene that is illuminated by luminous materials or, for example, if a complex arrangement of objects prevents the scene from being properly illuminated using **Light** objects, you will have to increase the **Diffuse Depth**, i.e., activate the **Secondary Method**. This method also offers the **IR** and **QMC** options, which use exactly the same settings, which instead affect the second light reflections on a surface and beyond. The **Diffuse Depth** value defines the number of reflections that the **Secondary Method** should calculate. The higher the value, the farther the light will be dispersed throughout the complex environment, the more precise the light simulation will be, the brighter the image will be rendered – and the longer it will take to render.

The **Intensity** and **Saturation** values can be defined separately.

The disadvantages of longer render times that result when higher **Diffuse Depth** values are used can be compensated for by setting **Secondary Method** to **Light Mapping**. This process calculates the dispersion of light from the observer's angle of view based on interlinked samples. This is less precise but very fast, even with a high **Diffuse Depth** value. Often, the lower precision hardly plays a role because the initial light reflections already depict the most intense light and are handled by the separate **Primary Method**.

The **Light Mapping** mode's diffuse depth is called **Maximum Depth**. As you can see, the default value of 16 allows for many more light reflections to be captured. This is why **Light Mapping** is often much brighter than the other methods, which can be compensated for using the **Intensity** value.

11.4.3.2.1 Light Mapping

As the term **Mapping** suggests, we are dealing with a fixed structure into which the Light **Mapping's** render results are fed. This map can in turn be saved as a **Cache** file and re-used to save time for subsequent renderings. The corresponding settings can be found in the **Cache Files** tab.

The **Light Mapping** structure has cells that can be compared to the pixels of a bitmap.

The precision of the **Light Mapping** varies depending on the number and distribution of these cells. In the end, the final result is produced by a blurring of neighboring cells, which will, as a rule, cause details to be lost. So-called 'light leaks' can also occur. These are regions through which light seeps through geometry. However, this method does offer incredibly fast rendering with high **Maximum Depth** values.

To determine the right settings for the **Light Mapping** precision you should first set Mode to **Visualize** in the **Light Mapping** tab. When the image is rendered, the **Light Mapping** cells will be displayed, which is a good way to analyze the size and distribution of the cells. The final rendering, however, should always be done in **Normal** mode.

As we already mentioned, **Light Mapping** first sends several samples into the scene from the observer's angle of view that are then reflected from objects' surfaces. The number of samples that are sent out is defined using the **Path Count (1000s)** value. As the extension suggests, this value is always multiplied by a factor of 1000.

The results of several of these samples are then summarized in cells. The size of such cells determines how many samples are interpolated within it – each cell can only save one color or brightness value. You can enable the **Show Preview Paths** option if you want to display this process during rendering.

If the cells are too large, details will be lost or the probability of light leaks will increase. If the cells are too small, render times will increase sharply. The cell size is defined using the **Sample Size** value, which can be applied with one of two available methods. If **Scale** is set to **Screen**, the **Sample Size** value correlates to a fraction of the image resolution. A **Sample Size** value of 0.01 will correlate to 1/100th of the width and height of the image. This bears the advantage that the cells adapt to the defined render resolution and produce consistent results. In addition, large objects will have more cells than small objects, which are often located farther away from the camera.

If **Scale** is set to **World**, the **Sample Size** will work with real units of measure e.g., centimeters. If your scene contains two cubes of the same size, and one lies near the horizon and one near the camera, both will have the same number of cells. This might be more precise but will be unnecessary in most cases because small or distant objects will not require the same precision as large objects or those that lie closer to the camera.

If **Direct Lights** is enabled, the surfaces that are illuminated by normal light sources will be assumed in the **Light Mapping**, which can speed up rendering and also make it more precise. This primarily applies to scenes with actual **Light** objects.

Enable the **Use Camera Path** option if you want to animate a camera movement. This will take into consideration the various camera positions and angles during the animation so the **Light Mapping** is only activated at those locations at which new shapes enter the image. This ensures a more stable calculation and can help reduce fluctuations in brightness during the animation. If you want to be absolutely sure, though, you should use the **QMC** method, especially for complex animations in which not only the camera but also objects are animated as well.

We already explained that the results of the samples are gathered in cells where a median value is ascertained. This often creates stark color contrasts and brightness in the **Light Mapping**. **Light Mapping** works best with more blurred transitions and the following options describe how this can be done. If **Prefilter** is enabled, a radius can be defined using **Prefilter Samples** within which neighboring cells are calculated together. If the radius defined is too large, a corresponding amount of detail and important differences between cells will be lost. The probability of light leaks will also increase.

In addition to the **Prefilter** function, the **Interpolation Method** affects the smoothing between cells when the image is rendered. If set to **Nearest**, the cells that lie nearest the pixels to be rendered will be interpolated. The **Sample Count** value defines the number of cells. This means that regions with many small cells will only blur correspondingly small sections of the image. Regions with large cells will blur correspondingly large sections. This is also a good method of maintaining the **Light Mapping's** existing details while also smoothing the cell's edges. If **Fixed** is selected, the calculation will take place within a fixed radius, which is defined by the **Size Ratio** value, around the image pixel, independent of the cell size. All cells within this radius will be interpolated and blurred. This method offers the strongest smoothing effect but also reduces the Light Mapping details accordingly.

You can also create a **Radiosity Map**. **Radiosity Maps** use a similar cell structure to **Light Mapping**. The cells are called **Texels** and are inseparably linked with the objects' surfaces. The **Map Density** value defines the size of the **Texels**. Larger values will create smaller Texels. The blurring between **Texels** is done using sample subdivision and works similarly to antialiasing.

Larger values will subdivide the **Texels** accordingly and can produce better results for interpolated values. However, render times and required memory will also increase. **Radiosity Maps** can also be saved as a Cache file and re-used.

This is also the primary advantage of using **Radiosity Maps**. Their structure can be evaluated very quickly during **GI** renderings, which can greatly speed up overall **GI** rendering. The disadvantage is that **Radiosity Maps** can require a lot of memory. This can be an issue when using **Team Render** when caches have to be sent across a network to other computers.

If these disadvantages are irrelevant, **Radiosity Maps** are a good way of speeding up **GI** renderings.

This is also why the **Radiosity Map** method is also offered for the **Secondary Method**. The **Cache** settings for **Radiosity Maps** can be found in the **Cache Files** tab.

11.4.3.2.2 Radiosity Maps

Radiosity Maps can also be defined for the **Secondary Method** for **GI** rendering. However, this option can generally only calculate and save the **Record Density**. Using **Radiosity Maps** as the **Secondary Method** is primarily designed to speed up the **Primary Method** and even improve its quality. This is especially the case when using **QMC** in combination with **Radiosity Maps** because the **QMC** process tends to produce noise if the **Samples** settings are too low. The interpolation and blurring of **Texels** that is integrated in **Radiosity Maps** can reduce noise without having to increase the **Samples** settings. However, the memory required for the **Radiosity Map** will increase accordingly.

The number and size of **Texels**, in which the **Radiosity Map** saves its results, can be defined using the **Map Density** setting in the **Radiosity Maps** tab. If **Mode** is set to **Normal**, the image will be rendered normally. If **Visualize Texels** is selected, the normal rendering will not be made and the entire **GI** pre-calculation will be skipped. Instead, you can have the **Texels** displayed on your object as a type of checkerboard pattern. Larger **Map Density** values will reduce

the size of the **Texels** accordingly. This makes the process more accurate but will also require correspondingly more memory. The **Sampling Subdivisions** value was also previously discussed in the **Light Mapping** section. This value represents the subdivision of **Texels** in even smaller regions, between which blurring will take place.

The following modes enhance the **Visualize** modes and are only meant for analytical purposes, not for final rendering. If **Mode** is set to **Visualize Shading**, the **Texel** squares will also be overlain with the color and brightness values saved for them. **Visualize Shading (Front)** and **(Back)** restrict the shaded overlay to the front and back of the polygons, respectively.

The previous description of using **Radiosity Maps** as the **Secondary Method** does not result in an increase of the **Record Density**. Rather the **Texel** structure will be used to speed up the **Primary Method**. The image brightness remains the same, regardless of whether or not **Radiosity Maps** is used as the Secondary Method.

Since **Radiosity Maps** only saves the **Primary Method** in this mode, it doesn't matter if, for example, the **Primary** or **Secondary Method's Intensity** is used to affect the **GI's** brightness. The result will be the same.

The increase in render speed can be quite remarkable if **QMC** is used in conjunction with the **Primary Method**. A reduction of 50% can even be achieved.

Additional **Record Density** for the **Primary Method** will be calculated if **Area** or Sky **Sampling** are enabled in the **Radiosity Maps** tab.

You should already be familiar with these options from the **Global Illumination** effect's **General** tab settings. Here, these options work identically and their accuracy is defined using the **Sample Count** value.

In any case, it makes sense to know how to implement the aforementioned processes and options. If you want to save time you can use the **Preset** menu from which you can choose from common presets, e.g., for interiors or outdoor scenes. These presets can, of course, be subsequently modified manually.

11.4.4 Denoiser

The **Denioser** effect is used to help reduce noise in renderings whose sample count is too low. This effect helps improve image quality in particular in conjuction with Physical materials, area shadows and Global Illumination, for example. **Denoiser** can be used in conjunction with the Standard or Physical renderer but is particulary effective when used with ProRender where it can achieve high-quality renering with low sample counts and short render times.

The **Denoiser** effect's settings are very clear and concise. In principle, only the type of material configuration used has to be selected via the **Albedo pass**. When using Physical materials, i.e., the surface color is simulated using diffuse reflectance, select Albedo. If your materials use the **Color** channel or if you use Node Materials and you have enabled the **Color** Channel for Node Material option in the **Preferences** menu, select the **Albedo pass material color**. **Denoiser** will then know which material property is primarily responsible for image noise.

When rendering with ProRender, this setting is not needed becaue a physical material setup will always be assumed.

Since the original rendering is changed when Denoiser is applied, you can enable the **Save the Raw Image** on a **Separate Layer option** to also keep the original render result as well as a multi-pass layer. The **Multi-Pass post effect** must be enabled.

As previously mentioned in the **Denoisers** settings section, adding multi-passes can improve results for **Albedo** or **material color** (depending on the **Albedo pass** selection) as well as for **material normal**.

11.5 Physical Renderer

We already mentioned the **Physical Renderer**, which is an alternative render method that can be selected in the **Renderer** menu at the top right of the **Render Settings** menu. This renderer can also be used in conjunction with effects such as Global Illumination. The **Physical Renderer** should always be used if you want to make use of the **Camera** object's **Physical** tab settings.

Even if you don't include depth of field or motion blur in your rendering, the Ph**ysical Renderer** offers several advantages compared to the **Standard** renderer. For example, in the **Advanced** tab you will find a **Raytracing Engine** selection menu. The default **Embree (Faster)** setting uses Intel CPU's SSE3 execution resources. These resources are executed directly from the CPU and speed up vector calculations and the conversion from float comma and whole numbers, for example. This can result in a substantial increase in render speed – but also requires additional memory. The **Embree (Smaller)** option will reduce the amount of memory required but will also be slower. The **Physical** raytracing engine from previous Cinema 4D versions is also available if your computer does not support **Embree**.

The **Quick Preview** setting affects the way in which the Cinema 4D render steps are updated. If **All Modes** is selected, a preview of the rendered image will be displayed after the pre-render or pre-pass. Its resolution will be very low but it will generally be good enough to judge whether or not there is anything that must be corrected before continuing the render process. The rendering can be aborted, if necessary. If **Progressive Mode** is selected, the low-resolution preview will only be displayed when the special **Progressive Mode** is used – we will discuss this in detail later.

The Debug Information Level option was also mentioned in the Global Illumination section. This setting lets you write Regular or Detailed information about internal processes to a text file. This information can then be provided to the Maxon support team in case you should have problems using the **Physical Renderer**.

11.5.1 Basic Tab

This tab offers numerous **Sampling** settings that you would otherwise have to define at various other locations if you used the **Standard Renderer**. At the bottom of the window, for example, you will find settings for **Blurriness**, **Shadow**, **Ambient Occlusion** and **Subsurface Scattering**. The numeric values will be used to the power of two to define the sample count. For example, a value of 0 will be converted to $2^0 = 1$; a value of 1 will be converted to $2^1 = 2$; 2 will be converted to $2^2 = 4$ and so on.

The corresponding settings for example for **Area** shadows for light sources or the **Ambient Occlusion** sample value will be grayed out and replaced by these settings. Individual deviations are not possible. However, Cinema 4D will also vary the number of samples itself. This is why many settings include the term **(Max)** because they only define the maximum possible value.

The **Physical Renderer's Sampler** setting replaces the **Anti-Aliasing** setting in the **Render Settings** menu. In the **Anti-Aliasing** menu, only the **Filter** options can be selected, i.e., to define whether the rendering should be sharper or more blurred. The **Sampler** setting basically subdivides all image pixels into even smaller sub-pixels and interpolates them to create smooth edges and to improve the quality of other effects such as depth of field or motion blur. Various methods are available.

The **Adaptive** mode works like the **Best** mode in the **Anti-Aliasing** settings and determines specifically where and to what degree sampling and subdivision should occur. The **Sampling Quality** options let you select between **Low**, **Medium** and **High** and the **Custom** option can be selected if you want to use custom values. Selecting **Automatic** will leave the decision making to Cinema 4D, which will then approximate the degree to which the image should be sampled.

The **Sampling Subdivisions** value in **Adaptive** mode defines the number of samples per pixel that will be made. Based on these samples, a decision will be made how subsequent **Shading** settings will be implemented, e.g., if they should lean more toward using the minimum or the maximum values. The **Shading Subdivisions** values define die minimum and maximum subdivision per pixel. These values will in turn also be applied to the power of two, which makes it possible to use float comma numbers for in-between values.

The **Shading Error Threshold** value works like the **Accuracy** setting in the Standard **Renderer**, as it is used for **Area** shadows or **Blur** effects, for example. Here, the percent value is applied inverted, i.e., small percentages will cause the Raytracer to lean correspondingly more towards the **Shading Subdivisions (Max)** value.

If **Sampler** is set to **Fixed**, the variation in sample density will be omitted entirely. Only the **Sampling Subdivisions** value will remain, which will be applied to each pixel. This means that you have fewer settings to deal with but you have to concentrate on the critical regions in the image. To produce a clean depth of field, the entire image must be rendered using a higher subdivision, even though far fewer samples would have been necessary in those regions of the image that are in focus. This variance is only possible in **Adaptive** mode, which is also faster, as a rule, than the **Fixed** adaptive mode.

The **Progressive** mode lives up to its name because it re-samples the scene, improving the render quality each time. No **Sampling Subdivision** values will be predetermined.

This mode can be configured differently using the settings made available when you click on the small black triangle next to the setting. If **Progressive Mode** is set to **Infinite**, the image will be rendered until you abort it manually. Each pass will improve the image quality. This mode can, for example, be used to fairly quickly create a preview that includes all effects. Simply abort the rendering as soon as you have a proper impression of the image.

Since a never-ending rendering via **Team Render** would not be very practical, the **Progressive Team Render Pass Count** value can be used to define a limit to the number of passes that will be rendered. We will discuss how to use **Team Render** later.

The **Progressive Pass Count value** works similarly. If **Progressive Mode** is set to **Pass Count**, this value can be used to also define the maximum number of passes that will be rendered on your local computer. After the maximum number of passes has been rendered, the render process will end automatically. If a **Save** path has been defined in the **Render Settings' Save** menu, the image will automatically be saved as well.

If **Progressive Mode** is set to **Time Limit**, the number of passes will be replaced by a time limit. Once the **Progressive Time Limit (minutes)** value is reached, the rendering will be ended automatically and saved if a **Save** path was defined. This cannot be used in conjunction with **Team Render** because multiple computers will render a single image. An image should, of course, be rendered in its entirety using a defined number of passes. This cannot be guaranteed if a time limit is used.

The **HDR Threshold** value can only be used if you define a 32-bit color depth in the **Render Settings' Save** menu. The extreme brightness that can be produced by HDR renderings can also lead to stark contrasts, e.g., in blurred regions. This value can be used to reduce the rendering's dynamic range. Smaller values will reduce the brightness of overexposed regions.

11.5.1.1 Blurriness

We already discussed how a camera's physical characteristics can be used to control depth of field. In order for these to be rendered, the **Depth of Field** and **Motion Blur** options must be enabled. The quality of both effects depends on the **Sampler** settings.

Motion blur offers additional settings for fine-tuning its precision. The **Motion Subdivisions** value is applied temporally. We've already defined the shutter speed in the camera's settings but the number of render passes within this time is defined by the **Motion Subdivision** value. When rendering rotating objects or objects that move along a swaying path, it's important to increase this value so the object's motion is actually "seen" for rendering.

The **Deformation Subdivisions** value has the same effect but only affects objects that are animated using deformations, e.g., a character.

The **Hair Subdivisions** value in turn only affects animated hair.

Note that **Motion Blur** in combination with **GI** works best when in **QMC** mode.

11.6 ProRender

The **ProRender** settings are clearly defined and mainly focus on the desired number of light reflections and mirroring between objects and the number of calculation steps per pixel. A large share of the settings appear twice and are arranged in the **Preview** and **Offline** menus. The **Preview** menu offers fewer options that are mainly designed for fast renderings in the Viewport. The **Offline** menu defines higher quality levels and offers additional options such as **motion blur**.

The **Render Mode** setting is set to **Global Illumination** by default, which simulates dispersed light, in addition to the normal surface shading. Hence, a **Global Illumination** post effect does not have to be added.

Various settings are available to limit and control the render precision. These can be made available by clicking on the small triangle next to the **Max Ray Depth** setting. The **Max Ray Depth** setting defines the maximum number of rays that can be used for a certain effect. This value also defines the limit for the subsequent settings. Even if, for example, the **Diffuse Depth** has a higher value, only the value defined for **Max Ray Depth** will be applied. In general, the ray depth stands for the reflections on the surface. If the ray deph is to low, several panes of glass standing behind one another will not be rendered transparent since no rays will be calculated for the objects that lie behind the panes of glass. The same applies to reflections. Imagine two mirrors facing one another, between which the rays would actually have to permanently be sent back-and-forth. The ray depth limits this calculation so that only a reasonable calculation depth can pass to the next pixel.

The following settings make it possible to restrict rays even more. The **Diffuse Depth** is for the rays used by **Global Illumination** that are used in the Reflectance channels **Lambert (Diffuse)** and **Oren-Nayer (Diffuse)** or within the BSDF node.

The matte depth will be used to calculate the reflections on rough surfaces using **Beckmann**, **CGX**, **Phong** or **Ward** reflectivity. These are not the samples used for roughness but only the number of reflection rays. Again, think of the example with the facing mirrors.

The **Ray Depth** relates to the refracting and transparent materials and the number of penetrations of transparent surfaces should subsequently take place. **Refraction Depth** is similar only that only rays will be controlled for matte, transparent materials. The **Shadow Depth** defines the number of overlying transparent or **alpha** masked regions should cast individual shadows.

The **Depth of Field** and **Motion Blur** settings define if these properties should also affect the rendering. Two options are available for **Motion Blur**. If **Linear** is selected, simple object animations for position, scale and rotation can be rendered quickly becaue their motion will be blurred linearly. For faster movements, this can however lead to unrealistic-looking results. Furthermore, many types of movements such as camera movements or deformations cannot be rendered. **Sub-Frame Motion Blur** is different. A specific number of frames defined by the **Samples** setting will be rendered and a motion blur will be calculated for each frame. Any type of motion blur can be depicted using this method but the render times will be correspondingly longer. The **Dithering** setting can be used to add random noise to the motion blur, which can soften hard edges in particular if lower **Samples** values are used.

To reduce things even further, **Render Mode** can be set to **Direct Illumination**, which is basically only good for test renderings because no shadows and no reflections will be rendered.

If Ambient Occlusion is selected, the result will resemble that of the Ambient Occlusion shader or post effect. A normal image will not be rendered but a grayscale image that is darker where surfaces lie in close approximation of one another. The search radius for this measure of distance can be defined using the Ambien Occlusion Ray Length setting.

The **Wireframe render mode** renders polygon edges as dark lines. The rest of the scene will be white. In this mode, all quads and n-gons will be converted to triangles for display. The Diffuse render mode offers a neutrally textured result in which all material properties are displayed in a neutral gray. This mode can be useful when evaluating light shading and shadow casting and is similar to the Override Material option available for the Standard and Physical renderers, which lets you use a **single material** to replace all materials in the scene for rendering.

When rendering with ProRender, the scene will be sampled pilxel-by-pixel from the angle of view of the camera. A continuous caluculation of color values within each pixel will take place, which will make the rendering more precise with each render pass. The number of of calculations or samples that are made per pass is defined by the **Samples per Iteration** seting. This setting does not play a role in the length of the rendering and the final quality because a low sample count can be offset by a higher number of iterations. More decisive is the point at which the rendering is stopped, which is defined farther below. We will discuss this later in the **Progressive Rendering** section.

The **Filter** settings work in principle as with the Physical or Standard renders and their antialiasing. They define how sharp or blurred the rendering is, which takes place during the per frame pixel sampling calculation.

The following settings only pertain to correcting errors in rendering. For unbiased rendering, as is the case with **ProRender**, such errors can be in the form of single, very bright pixels. These are generated when relatively few samples per pixel are used. If a sample ray happens to hit a very bright surface, e.g., a light source, a luminous polygon or the sun in an HDRI environment, this pixel will be made extremely bright. Normally, this is corrected automatically as the rendering progresses when an increasing number of samples is gathered and mixed – but you don't always want to wait that long for an error-free image. This is why the brightness ascertained by the samples can be restricted using the **Radiance Clamp** option. This value can be used to restrict the number of reflected or refracted rays. Direct reflections, i.e., the reflection of a light source or a sky on an object, are not affected. Note, however, that a very low **Radiance Clamp Value** can cause a loss of variations in brightness or details, e.g., those created by caustics.

The **Firefly Filter** principally has the same job as the **Radiance Clamp** setting. It is also used to correct overly bright pixels, also referred to as fireflies. This option, however, does not restrict the brightness during sample calculation but compares the brightness of each pixel with its neighboring pixel after the image has been rendered. If an individual pixel is brighter as its neighboring pixel, its brightness will be reduced. The **Firefly Threshold** value defines the point at which this function should take effect. The smaller the value is, the more the brightness of a contrasting pixel will be reduced. This can, however, result in a loss of detail in the rendered image and should therefore not be made too strong.

Finally, the **Reload Scene per Frame** option can be used to optimize the memory requirements of your scene duing rendering. Normally, the entire scene has to be loaded by the graphics card for each frame of animation. This can lead to very long render times for complex or long animations. If this option is disabled, an optimized algorithm will be used which will prevent the entire scene from being loaded to the graphics card each time. If display problems occur, this option should be disabled. This can, for example, be the case for shaders that are animated but have no keyframes.

This concludes the quality settings for **ProRender**.

The **Level of Detail** settings and options primarily affect parametric primitives, splines and generators whose number of objects or polygon density can be increased or decreased as a percent for rendering via the level of detail. Enable the **Render HUD** option if HUD elements should also be displayed in the rendering. The **Render Doodle** option works in a similar fashion and lets doodles or other elements created using the **Doodle** tool me made visible for rendering.

11.6.1 Progressive Rendering

ProRender is an unbiased renderer. This means that rendering is not sped up if information is suppressed or pre-calculations are assumed, as can happen with global illumination for the **Standard** or **Physical** renderers. The advantage is that much fewer test renderings and parameter definitions have to be made while still achieving realistic-looking results. The rendering takes place in several steps, called iterations, in order to also quickly supply a fast feedback. An iteration represents a complete render process of all image pixels. The entire rendered image will be visible at the end of an iteration. All **Samples** for each pixel will be used, as defined at the top of the menu under Samples per **Iteration**. The higher the image resolution and the more samples per pixel that have to be rendered, the longer each iteration will take to complete. Since this rendering uses a lot of the graphics card's processing power, the render times can be sped up using a fast GPU. Since multiple graphics cards can be used for rendering, this process can as a rule be sped up more efficiently – and within a single workstation – compared to CPU rendering. Since a single iteration generally uses far too few samples per pixel to produce a noise-free rendering, this process is repeated consecutively several times. The location of the samples in the pixels is different with each repetition, which results in a better and more

precise rendering with each iteration. The **Iteration Count** is therefore the main quality criteria next to **Samples**. In any case, the number of iterations improves the quality of rendering accordingly, at least when more than one **Sample** per pixel is defined. This process can be infinite but at some point, the rendering will reach its zenith at which point additional iterations will not do anything more to improve the quality. This is why the render time per image should be restricted. There are various modes in the **Stop Condition** menu from which you can choose.

After an **Iteration count** has been defined, you simply select a condition under which rendering will stop. Since this value can vary depending on the complexity of materials and scene environment, you should perform test renderings to find out which **Iteration Count** produces the best results for your needs.

You can also define a **Time Limit** that restricts the render time for the iterations. Each iteration will be rendered in its entirety. A **Threshold** value can also be defined, which performs a running check of an image's noise. If a pixel's noise is less than the defined **Threshold** value, the rendering will end after the iteration has been completed. This method can be used to carry over characteristics from one rendering to another since image noise is a typical characteristic of unbiased rendering, regardless of which motif is rendered. The smaller the **Threshold** value, the more samples and iterations that will be calculated per pixel. The change in pixel color after each iteration will be compared with the previous result.

You can also set **Stop Condition** to **Never**, which would normally mean that the rendering will be performed infinitely. However, since the renderer also has a Command Line, for example, which means that it can be started outside of the Cinema 4D interface, an iteration limit can also be defined here, which makes this comparable to the **Iteration Count** option.

11.6.2 Refresh Rendering Interval

Normally, the display of the rendering in the Viewport or in the Picture Viewer will be refreshed after each iteration so you can assess the rendering as quickly as possible. This update requires some time, which extends the overall render time. Therefore, it's better if the completed iteration is refreshed for the final rendering. This can be defined using the **Refresh Rendering Interval** settings. You can select either **Time Interval(s)** or **Iteration Interval**. With the former, the image will be refreshed after a defined period of time (each iteration will be completed entirely). The latter will refresh after an iteration has been completed.

11.6.3 Options and Technical Requirements

ProRender is so fast because it uses one – or more – graphics cards to render. This also brings restrictions with it because graphics cards generally don't have the amount of memory we are used to having when rendering with the CPU and the RAM can't be used. Therefore, make sure the graphics card you buy has enough memory. The entire Cinema 4D project, including textures, must fit within the graphic card's memory. The graphics card also needs to process other programming languages than when it renders Cinema 4D via the CPU. This is why a great number of shaders can't be rendered using the graphics card and have to first be converted to bitmaps. The following two settings are designed to deal with this in **ProRender**.

If **ProRender** is not able to render a scene due to a lack of sufficient memory on the graphics card, try enabling the **Bucket Rendering** option. This will render only one section of an image at a time on the GPU. These sections are called Buckets. The **Bucket Width** and **Bucket Height** settings are used to define the resolution of the Buckets. The order in which Buckets are rendered is defined using the **Bucket Sequence** setting.

The previously mentioned baking of shaders is defined using the **Default Texture Resolution** settings. These are global settings that can be overwritten in the shader's Base settings, for example if you need a higher resolution for an individual shader.

11.6.4 Render Multi-Passes with ProRender

ProRender has its own multi-pass system for rendering. The following channels can be enabled for output:

- The Direct Illumination channel generates shading that is created by light that is cast onto a surface directly from a light source or from a luminous surface.
- **Direct Illumination** is made up of three additional passes that can also be output individually. **Direct Diffuse** defines the shading light effect of the light that hits the surface directly. **Direct Reflection** is the first reflection on the surface and **Emission** is the self-illuminative property of the surfaces.
- The Indirect Illumination channel generates light that is passed on by reflections in the scene.
- **Indirect Illumination** can also be broken down into individual passes for **Indirect Diffuse**, **Indirect Reflection** and **Refraction**. A fourth pass for **Volume** can be used in conjunction with volumetric effects when they are, for example, displayed using a Volume Node.
- Adding all passes for **Direct Illumination** and **Inidrect Illumination** produces the final rendering.
- The Environment channel displays the sky or physical sky as the camera sees it.
- **Ambient Occlusion** outputs the Ambient Occlusion calculation separately. The ray length used can be defined in the **Offline** tab's settings.
- **Albedo** is the diffuse surface color of all objects. No illumination will be taken into consideration.

The following multi-passes make structural information available that does not directly belong to the rendered image.

- **World Coordinate** colors all surfaces with respect to their location in the room in relation to the world coordinate systsm
- **Texture Coordinate** also colors surfaces but in relation to the texture coordinates on the surface
- **Geometry Normal** converts the orientation of the surface normal to RGB color values. The non-smoothed Normals will be used as they are positioned by the respective polygons
- For **Shading Normals**, the direction of the shading Normals on the surface will be converted to RGB color values. In the process, the changes made to the Normal orientation caused by Phong and Normal tags, as well as by bump or Normal properties will be passed to the material
- **Depth** outputs the distance of the geometry from the camera in grayscales. The farther away objects lie the brighter they will be. Since this pass is output in 32 bits, manual lighting adjustments can be helpful to modify the pass to make the visible brightness range lie within a specific distance from the camera
- **Object ID** is already known from the other multi-pass system. All objects will be displayed with a random color, which can, for example, be loaded as a selection during editing
- **Object Group ID** works similarly to the **Object ID** function but you can also assign unique colors to individual objects. To do so, apply a **Render** tag to the respective object and select the desired color in the **ProRender** menu.
- **Material ID** colors all objects that have the same material with the same color. Otherwise, objects will be assigned a random color

In the **Anti-Aliasing** menu at the bottom you will find an additional list of options that can be used for rendering anti-aliasing for individual passes. An overall anti-aliasing will be applied to multi-passes not listed here.

SUMMARY: RENDERER

- **Ambient Occlusion** is not only available as a shader but as a render effect in the **Render Settings** menu as well. It bears the advantage that the effect can be rendered separately as a **Multi-Pass**.
- The **Ambient Occlusion** effect can be saved to a separate **Cache** and re-used. This speeds up rendering if the same motif is rendered several times and can also increase the effect's quality while simultaneously reducing render times.
- **ProRender** has its own **Ambient Occlusion** mode.
- **Caustics** can also be added as an effect and will automatically be calculated for **ProRender**.
- **Global Illumination** enhances direct illumination by light sources with diffused light. It can also be used to create light sources using luminous materials. Global Illumination is no longer necessary when working with Physical materials.
- Various **GI** render methods are available, depending on the desired quality and the complexity of a scene when the Standard or Physical Renderers are used.
- **Irradiance Cache** can be saved to a re-usable **Cache**.
- The scene will be sampled by a **Pre-Pass** prior to actually being rendered.
- The automatically distributed measuring points use Samples to gather information about lighting conditions. The results are then interpolated and blurred accordingly.
- **Light Mapping** uses a fixed structure in which the calculated results are saved. This structure can also be saved separately as a re-usable **Cache** file.
- **Light Mapping** makes a much deeper sample depth possible compared to the **IR** method but generally produces less detail.
- **Radiosity Maps** are primarily used to store brightness and color information and do not constitute an actual process as such.
- Using **Radiosity Maps** can greatly speed up **GI** rendering, which will, however, also increase render times. The disadvantages can be compensated for using the **QMC** method.
- **QMC** stands for **Quasi Monte Carlo** and is a brute force **GI** render process. An adaptive **Prepass** calculation will be made because each pixel will be rendered with equal precision. This is the most precise but also the slowest method.
- **QMC** is especially well suited for very precise rendering or for rendering animations with **GI**. If other methods are used, re-arranged measuring points can produce variations in brightness in animations.
- The **Physical Renderer** not only enables the camera's physical characteristics but also make the **Raytracing Engine** available.
- If your computer's CPU supports **SSE3**, **Embree** can be used, which can greatly speed up mathematical calculations during rendering.
- The **Physikalische Renderer** also includes the Cinema 4D **Sampling** and offers additional anti-aliasing options.

12 Team Render

Team Render lets you simultaneously render images or animations on external computers (clients) that are part of your network environment. This can, of course, dramatically reduce render time. The more computers that are in your network, the faster images or animations can be rendered. Cinema 4D itself acts like a server for the render clients. The **Client** version of **Team Render** must be installed on each client.

Team Render is enabled in the **Preferences** menu's **Renderer** menu. A separate sever application is available for Team Render, which makes it possible to render and manage jobs even if Cinema 4D itself is not running. When you enable the **Enable Team Render** option and subsequently the **Share Machine over Network** option, **Team Render** will be activated. If other computers in your network are running and have the **Team Render Client** installed and running, a list of active computers will be displayed in the **Team Render Machines** window, which is located in the main **Render** menu. New clients can, for example, be accessed by entering their respective I.P. addresses. To do so, select **Machine/Add Machine** in the **Team Render** computer's **Team Render Machines** window. The rendering can also be started by selecting **Render/Render to Picture Viewer**, but this will only use your own computer for rendering.

If the **Show Bucket Color** option is enabled in the **Team Render Machine** window's **View** menu, you can see if and where each client is rendering based on the color of the **Buckets**.

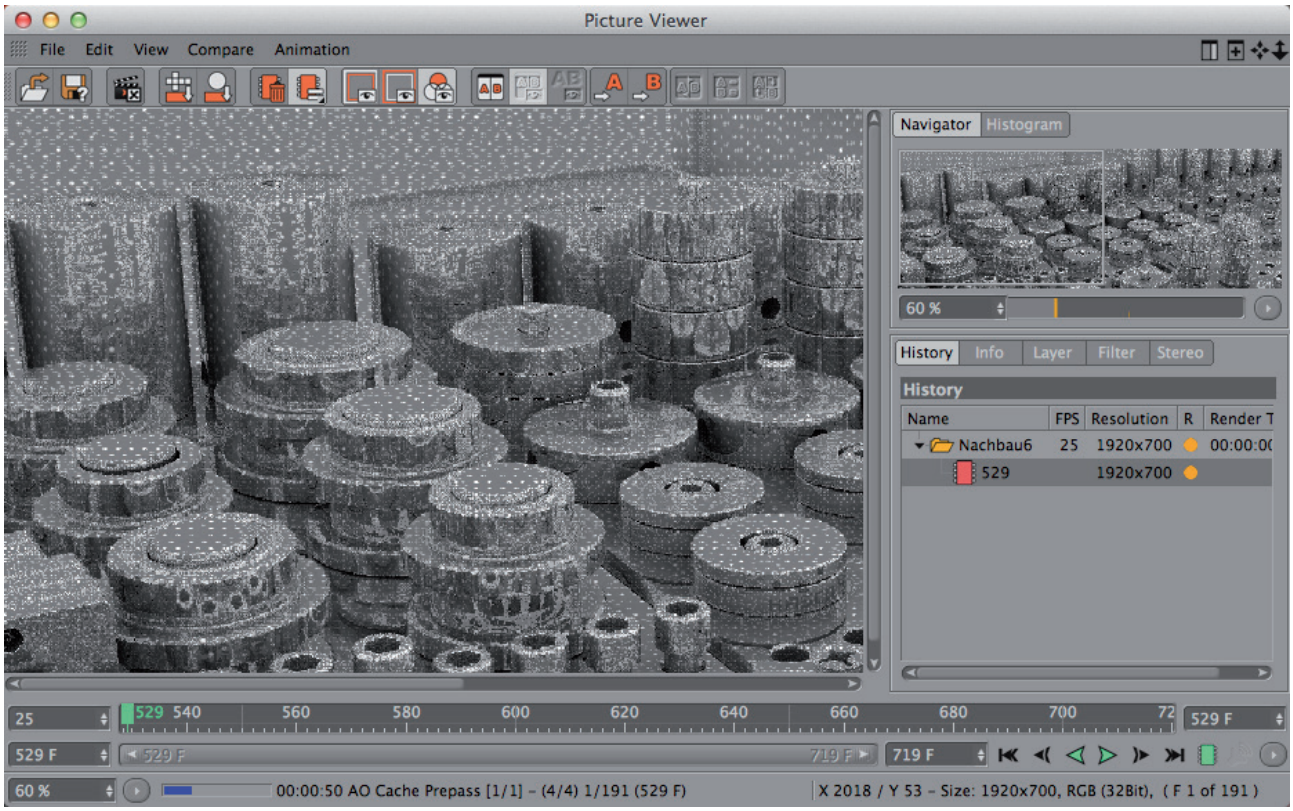
Since scene files and caches are exchanged between the server and clients, you should configure any existing firewall to allow this exchange. Furthermore, using a wireless network should be avoided. Computers that are physically connected via network cables generally offer a faster exchange of data.

Since some caches can be very large, e.g., **Radiosity Maps**, the **Team Render** settings in the **Render Settings** offers additional options with which this can be optimized.

If an option is enabled, the clients will work on the rendering of the respective cache together and will then split the file amongst each other across the network. Since **Radiosity Maps** are often very large, the respective option is disabled by default. Each client will render its own cache, which avoids transferring unnecessarily large amounts of data across the network.

13 Picture Viewer

We already mentioned that a rendering can be started in the **Picture Viewer** according to the settings defined in the **Render Settings** menu. It doesn't matter if **Team Render** is enabled or not. The **Picture Viewer** bears the advantage that it is completely independent of the loaded scene. As soon as a rendering is started in the **Picture Viewer**, you can even close your Project and, for example, start working on another Project. The rendering will continue on its own. However, closing the **Picture Viewer** will end the rendering process. You can simply minimize this window to get it out of the way and the rendering will continue. Below the preview window in the **Picture Viewer**, in which you can observer your rendering, you will see a progress bar next to which the elapsed render time is displayed. You can zoom in or out of the rendering by clicking on the small button next to the percentage field and selecting a corresponding option. You can also enter a percentage value manually.



Selecting **Fit to Screen** will automatically fit a high-resolution rendering into the **Picture Viewer's** preview window. Alternatively you can simply double-click on the rendering in the preview window. This has nothing to do with the actual image size with which the image will be saved, which is defined exclusively in the **Render Settings' Output** menu.

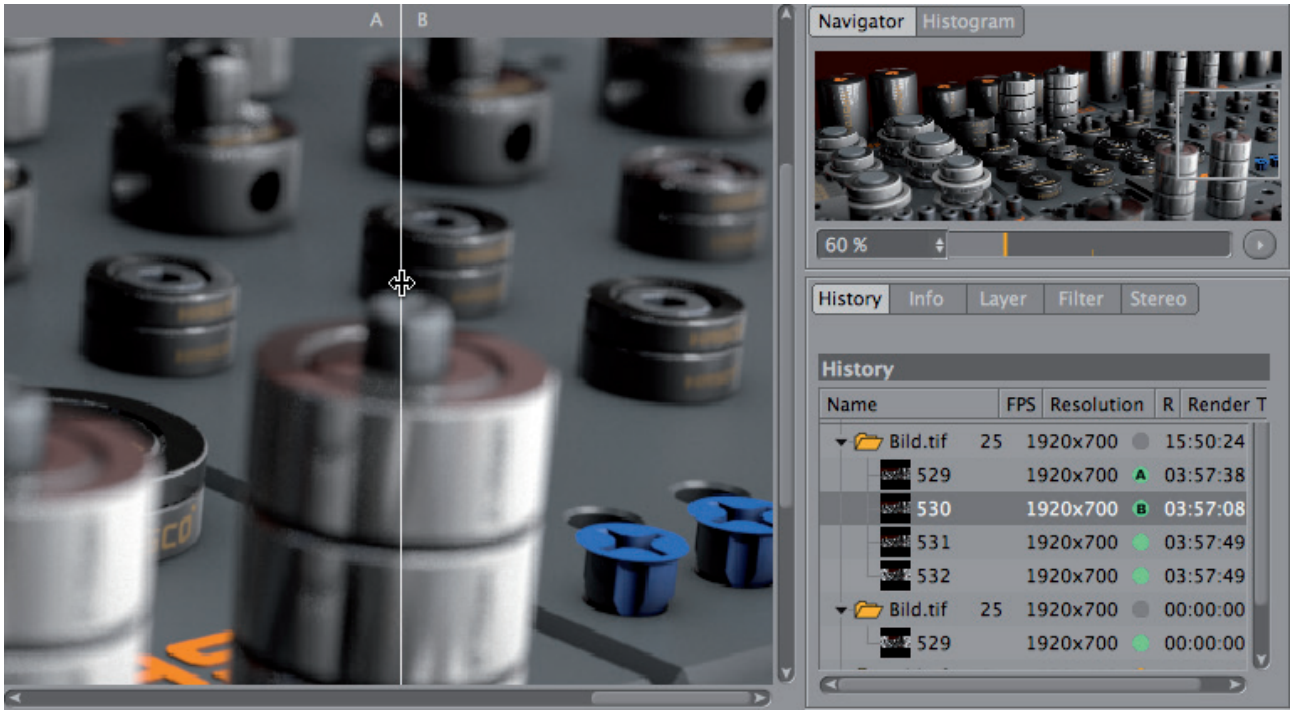
At the top right of the **Picture Viewer** you will see the **Navigator** tab, which also displays a small preview of the last frame rendered. Here you will also find a zoom value with a slider and a button from which a zoom option can be selected. Next to the **Navigator** tab is the **Histogram** tab, which contains information about your rendering's **Luminance** and **RGB** values, for example. The **Display Filter** option is only enabled if the **Enable Filter** option is enabled in the **Filter** sub-menu. More about this later. Speaking of color, it might be of interest to you that pressing the **Cmd/Ctrl** key will display a color picker when you hold the cursor over the **Picture Viewer's** main preview window. The RGB color values of the color over which the color picker is placed will be displayed at the bottom of the **Picture Viewer**.

Below the **Navigator** and **Histogram** tabs you will find several additional tabs that each contain specific information or settings. We will take a look at the **History** tab first. Here, a list of all rendered images is displayed. In addition to a tiny preview, you will also see the frame rate (**FPS** (frames per second)) displayed for animations, the **Resolution** and the **Render Time** that was required. The **R** column displays a colored dot. If this dot is gray, the image was rendered but no longer lies in the **Picture Viewer's** cache and can therefore no longer be displayed as a preview. This often

happens if you render a great number of images or if an image's resolution is extremely high. The **Picture Viewer** only has a limited amount of memory available for displaying rendered images. The **Memory** setting for the **Picture Viewer** can be found in the **Preferences** menu's Memory menu.

Images that are in the cache will have a green dot. You can click on these to display them in the main preview window. This can also be done while another image is rendering. The image that is currently being rendered will have a yellow dot and its tiny preview image will also be missing. An important advantage of being able to store rendered images is that it is very easy to compare test renderings. You can easily see if the modification of a given parameter produces the desired effect. The **Picture Viewer** has a special mode that can be used to simultaneously display two images. Right-click on an item in the **History** list that you want to compare. A context menu will appear from which you can, for example, delete images from the cache (**Remove Image**). To compare an image, select **Set as A**. A black **A** will appear in the item's green dot. Right-click on the second image you want to compare and select **Set as B**. A black **B** will appear in the item's green dot. This lets you quickly see which images are being compared. If the cursor lies over a rendered image in the list, the **a** or **b** keys, respectively, on your keyboard can be used to quickly assign the **A** or **B** status to the image.

The same commands can also be accessed via the **Compare** menu at the top of the **Picture Viewer**.



This menu offers additional options. If **AB Compare** is enabled, a white line should appear in the main preview window, which can be clicked on and dragged. The **Show Line** option must be enabled in the **Compare** menu. The **Show Text** option is also useful because it displays the images' respective **A** or **B** letters in the preview window. If **Swap AB** is enabled, the images will be swapped. **Swap Vert./Horiz.** will switch from a horizontal split to a vertical split and vice-versa. You can even compare two image sequences, i.e., animations. To do so, select the images as before and enable the **Animate Second** option in the **Compare** menu. If the selected image is part of a sequence, the **Picture Viewer** will also make icons available with which the sequence can be played, etc. However, all images of the given sequence must be present in the **Picture Viewer's** cache. Images can be loaded from the hard drive but then the sequence will not play back smoothly.

The **Compare** menu also contains the **Difference** option. The dividing line will be removed and both images will be overlain. Only the differences between the two images will be visible as a pixel pattern. Identical regions will be displayed in black.

13.1 Info Tab

This tab displays information relevant to the image selected in the **History** list, including the **Name**, **Directory**, color **Depth**, date of rendering, file size (**Memory**), **Color Profile** and **Render Time**. Below, the **Title Safe**, **Action Safe** and **Pixel Aspect** values are displayed.

13.2 Layer Tab

As you know, images can be saved as multiple layers using the **Multi-Pass** function. A very simply variation of this would be a simple alpha channel or grayscale gradient. All of a rendering's elements are displayed in the Layer tab's list. If the **Image** option is enabled, the **Picture Viewer** will only display the rendered image. If you want to display the individual image layers, enable the **Single-Pass** option and click on the corresponding layer in the list below. If the **Multi-Pass** option is enabled, you can use the eye icon for each layer to turn it on or off, or you can use the mix modes and **Strength** slider to modify the layers.

This can be done because the **Picture Viewer's File** menu lets you save renderings. This can also be done if you did not define a **Save** path in the **Render Settings** menu. In the File menu you will find the **Stop Rendering** command, which can, for example, be useful when using the **Physical Renderer** in conjunction with the **Progressive** render method. Otherwise the rendering can be aborted at any time by simply closing the **Picture Viewer** window. The **History** list will not be affected by this. The next time you open the **Picture Viewer** window, all images contained in the cache will be displayed.

13.3 Filter Tab

This tab contains numerous sliders that let you adjust **Saturation**, **Contrast**, **Gamma** and other properties. Curves are also available with which individual color values can be easily adjusted. These settings are all made available when the **Enable Filter** option is enabled. The changes made here will be updated automatically in the preview window. This is very useful for animations in particular because you can simply render the first image and use the Filter tab's settings to adjust the look for all subsequent images. Once you have made all adjustments, click on the **Create Post Effect** button at the bottom of the menu. This will add a corresponding **Color Correction** post effect to the Render **Settings** menu. This means that all subsequent renderings can be rendered with these exact same settings if this post effect is added. However, **Enable Filter** should be disabled again prior to rendering in the **Picture Viewer** so the images are displayed the same way that they will be saved.

Enabling the **Enable Filter** option will apply the **Filter** settings directly to the **Histogram** preview. This is useful, for example, when correcting the **Filter** settings with regard to overlapping brightness. Once you've found the right settings you can click on **Save Preset** to save these settings as a preset in the *Content Browser*, which can be loaded at any time by clicking on **Load Preset**. This is very useful if you want to apply a specific look to another Project. If you click on **Reset Filter**, all sliders will be returned to their original positions and the preview image will not be modified. Note that the **Filter** settings will only be applied to the image when you use the **Color Correction** post effect for rendering. When the images are saved via the **Picture Viewer** they will generally be saved without the modified **Filter** tab's settings!

14 Render Queue

You now know how to render an image or an animation and get the most out of the available resolution, optimize quality and save the file automatically after it's been rendered (if a **Save** path was defined in the **Render Settings** menu). But what happens if you want to automatically render several still images or different Projects overnight, for example? This is what the **Render Queue** is for, which can be found in the main **Render** menu. If the current Project should appear in the list, simply select **Render**/Add to **Render Queue**.

At the top of the dialog window you will see a list of the Projects that have been added, which should be rendered one after the other. Before a Project is added to the list you should make sure that all necessary quality settings, render processes, effects, resolutions and of course **Save** paths have been correctly defined. You can add any number of Projects to the list by using the **File**/**Open** command. Each item on the list will automatically be numbered. The items can be re-sorted by simply dragging and dropping them into their new position. Disabling an item's **Render** option will omit it from being rendered. If you want to use **Team Render** to render an item on the list, simply enable the item's **Team** option.

The **Status** column shows the current status of the item's rendering. **In Queue** means that the item has not yet been rendered; **In Progress** means that the item is currently being rendered; **Finished** means that the rendering was completed successfully; **Stopped** will be displayed if the rendering was aborted and **Error** will be displayed if the rendering was aborted because a material was missing, for example. The Render Settings column displays the name of the Render Settings used for the respective Project. If a Project has multiple **Render Settings**, you can also subsequently define them to be used as the preset for the rendering.

To do so, use the **Render Settings** button at the bottom of the **Render Queue** window. Next to this setting you can define which **Camera** should be used for rendering. Of course, this only works if more than one **Camera** object is present in the scene. This is a very useful function because you can add a single Project to the **Render Queue** multiple times and if the scene contains several cameras, the animation can be rendered using each camera, e.g., to show a product or building from several angles. This means that you don't have to have a separate scene for each camera!

The Logfile column displays information regarding any problems that occurred. This means that no **Save** path was defined for the image or animation in the **Render Settings**. The **Render Queue** will generate its own name for the file so rendering can continue. This can, however, result in images with the same name being overwritten. In such cases you should either define a **Save** path in the Project itself or use the **Output Path** setting in the **Render Queue** to define a path. This setting can be used to subsequently define a save path for the image or for a **Multi-Pass** image. The **Multi-Pass Image** setting will be grayed out if the **Multi-Pass** effect is not active in the Project.

A **Log** file can also be created for each render job. The path to which this file is saved can be defined at the bottom of the **Render Queue** window. If you don't want to save a log file, simply leave this field empty or disable the function in the **Preferences** menu's **Renderer** menu. The content of the log file can also be viewed without saving the file by switching to the **Logfile** tab. The information that is displayed here is defined via the **Show**/**Filter**/**Filter Log** command.

System refers to your computer's identification information and can therefore be left disabled in most cases. **Render** and **Project** will list the most important information regarding the **Render Settings** for the saved image and the scene, respectively. Selecting **Show**/**Filter**/**Open in Picture Viewer** will open the image in the **Picture Viewer**. The other commands can be used to open the image in your computer's **Explorer** (Window) or **Finder** (Mac).

If you want to make sure that no errors will occur during rendering, you can check the Project's materials and textures when it's loaded. If, for example, a bitmap is missing, you will be notified and can add the bitmap before you start rendering. To check your Project, select **Check Textures** from the **Jobs** menu. You can then open the corresponding Project by selecting **Jobs**/**Edit Project**, which will automatically open the Project in Cinema 4D. After you have made the corrections to the Project simply save it and you can start rendering. If only **Texture Errors** is enabled in the **Jobs** menu, the rendering will only be stopped if a texture is missing. In this case, the **Render Queue** will continue rendering with the next item on the list. If **Texture Errors** is disabled, Projects in which textures are missing will also be rendered. This will, of course, also be visible in the rendered images.

Projects can be removed from the list by right-clicking on them and selecting **Delete** from the context menu that appears. You can also select **Jobs**/**Delete** from the menu or click on the trash icon. The rendering can be started by selecting **Jobs**/**Start Rendering**. Select **Jobs**/**Stop Rendering** to stop the rendering. If using the **Physical Renderer**, make sure that **Sampler** is not set to **Progressive** in conjunction with **Progressive Mode** set to **Infinite**. Otherwise the **Render Queue** will render infinitely unless it's stopped manually! This will, of course, prevent all subsequent Projects on the list from being rendered. After all, the Render Queue is designed to let you render while you are absent so you don't have to constantly monitor your computer during rendering.

▶ *See: Exercises for Rendering and Various Render Techniques*

SUMMARY: RENDER MANAGER

- **Team Render** can be used to render images or animations over a local network.
- The **Team Render Client** must be installed on all client computers.
- Each client can be assigned its own unique password.
- Cinema 4D itself serves as the server and can verify the **Team Render** clients via its own interface and define which clients will be used for rendering.
- The render speed is also dependent on the speed with which the network is able to transfer files such as caches, textures and the scenes themselves. This is why you are able to define which caches should be rendered by all clients and subsequently be distributed across the network, and which caches should be rendered locally by the clients.
- The **Render Queue** can be used to successively render multiple Projects by adding each Project to its list. Individual Projects can also be rendered using **Team Render** by enabling the corresponding option in the **Render Queue**.
- A final rendering – with or without **Team Render** – is started in the **Picture Viewer**. It will copy the scene to a separate memory cache before rendering. The scene can therefore be modified while the **Picture Viewer** renders without affecting the rendering.
- The **Render Queue** displays the progress of the rendering but can also be used to manage older renderings in the list as long as they are still in the cache. This makes it possible to compare different renderings.
- Individual **Multi-Pass** layers can be displayed in the **Render Queue** even during rendering. The layers can also be mixed so post-production can be completed directly in Cinema 4D.
- The **Filter** tab's settings can be used to adjust the color, brightness and contrast of the image(s). Once the fine-tuning is complete, the settings can be saved as a separate render effect and applied to subsequent renderings.

15 Managing Projects and Versions

There are several good reasons for saving multiple versions of a given scene. A client can, for example, request a new version of the scene but then change his mind and revert back to the previous version; or you want to use multiple cameras and various render settings within a single scene. If you render certain passes or layers, it can be useful to create several variations of a scene if objects for certain passes should be hidden or rendered with different render settings.

One solution would be to save different versions of the Project under different names. However, if an animation, lighting or material is changed in one file, the same changes have to be made to all other files. A much easier solution is to use the Take System, which has its own tab in the ***Object Manager***.

15.1 Main Take

The **Main** Take is always present and contains the entire scene by default. As a rule, new objects, materials and tags will automatically be added to the **Main** Take. A new Take is created by selecting **File/New Take** in the **Take Manager**. Double-clicking on a new Take will let you rename it. New Takes are automatically grouped below the **Main** Take by default, which means that they contain all elements of the **Main** Take. Changes to new Takes can be made using various methods.

15.2 Switching Cameras

To use a different camera from the one that is active in the **Main** Take, use the camera icons to the right of the Take in the Take Manager. By clicking on the rectangular symbol next to each Take you can, for example, switch between the **Main** Take and a newly added Take thereby automatically switching the camera. This eliminates the need for creating a **Stage** object and setting keys for the camera if the Project should be visualized from various cameras' angles of view. If no selection is made, Takes will automatically use the camera from a Take higher up in the hierarchy (this will usually be the camera in the **Main** Take).

15.3 Switching Render Settings

If you have saved multiple presets in the **Render Settings**, you can select which preset should be used for each individual Take using the clapboard icon at the right of the respective Take in the **Take Manager**. This means that each Take can have its own render settings for rendering. If no preset is defined for a given Take, the active render settings of a take higher up in the hierarchy will be used, which is generally the **Main** Take. If you're working with very many Takes, a useful feature is the **Token** function, which lets the name of the camera, render settings or even the Take itself automatically be carried over to the save path of the rendered image or animation. The **Token** function can be used to automatically create a folder at the save location.

15.4 Switching Visibility and Tags

A very practical feature of the Take System is the fact that the visibility of objects and/or lights can be easily switched, e.g., from a daytime scene to a nighttime scene. To do so, create two new Takes below the **Main** Take and name them, for example, 'Day' and 'Night', respectively. Add an Override Group to each Take using the **Add Override Group** command. A folder will be created at the right in the Take Manager into which objects can be dragged from the ***Object Manager***.

Drag elements into the respective folder whose properties should be different from those of other Takes. For example, all light sources for the 'Night' scene can be dragged into the 'Day' Take's **Override** folder and then hidden by setting the dots to the right (V column) to red. Of course the same can be done with the 'Day' Take's lights – drag them into the 'Night' Take's **Override** folder and hide them for rendering and the Viewport. When you switch between Takes, the lighting situations will be switched accordingly.

Render settings and tags, for example, can be switched in a similar way. To the right of the Override Group you will see a plus symbol that can be used to add various tags to the objects. This is particularly helpful with regard to **Compositing** tags, e.g., to control the visibility of cameras or shadows.

15.5 Overriding Settings

Settings can be overridden when switching between Takes, e.g., to change the radius of a sphere primitive or the color of a material. This can be done in one of two ways. After activating the respective Take you can, for example, right-click on the name of a setting in the ***Attribute Manager*** and select **Override** from the context menu that appears. The value will be made available for editing and can be modified. Values that were not overridden will automatically maintain their original values used in the parent take (which is most often the **Main** Take).

If you want to modify several settings it can be quite laborious clicking on each one individually. The **Take Manager** offers the **Auto Take** function in the **File** menu, which is a special mode that lets you modify all settings of a selected Take at once.

15.6 Render Take

After all Takes have been set up correctly, each one can be rendered individually or all Takes can be rendered successively using an automated procedure. To do so either click on the gray dots at the right of the Take in the Take Manager and then select **Render Marked Takes to PV** or **Team Render Marked Takes to PV** from the Take Manager's **Render** menu. The commands **Render All Marked Takes to PV** and **Team Render All Marked Takes to PV** will automatically render all takes successively.

15.7 Organizing and Switching Takes

The Viewport **HUD** can be used order to always maintain a good overview of which Take is currently active since the ***Take Manager*** is hidden by the ***Object Manager*** as a rule.

In the Viewport you will find the **Current Take** option in the **Options/Configure** menu's **HUD** tab. If this option is enabled, the name of the currently selected Take will be displayed at the top center of the Viewport and clicking on the name will let you switch to other Takes without having to open the ***Take Manager*** to do so.

15.7.1 Exercise for Creating Takes and Rendering Different Takes

Exercise: Use an existing scene with lighting and create a variation using the Take System so you can switch between the two versions. Also let a different camera perspective be rendered automatically.

16 Animation Basics

An animation is a sequence of individual images. Modeling, lighting and rendering principles remain the same. However, there are several settings that must be made prior to animating.

16.1 Project Settings

These settings can be accessed by selecting **Edit**/**Project Settings** from the main Cinema 4D menu. As you already know, these settings include the Project's scale and **Default Object Color**, etc. In the following we will take a look at the settings that are relevant for animation.

The units with which time is measured and the frame rate are important when creating animations. You might already be aware of the fact that different presentation devices use different frame rates. For example, European television uses 25 **FPS** (frames per second) and US television uses 30 **FPS**.

Films for cinema are played at 24 fps. Other countries (regions) use even other frame rates for their video systems, not to mention online films with their almost random frame rates. When setting up your animation, one of the first things you should do is define the frame rate with which it will be rendered. Generally speaking, the more frames per second of film, the longer the rendering will take, the smoother the movement will be and the larger the file(s) will be. The technical requirements for playing the animation smoothly on a PC will also increase accordingly.

The **Project Time** value defines scene's current time within the animation. This has more of an informational purpose because this information can be accessed more easily at other locations in the Cinema 4D interface. We will discuss this in more detail later when we discuss the **Timeline**, which is located below the Viewport. The temporal range that the **Timeline** displays is defined by the **Preview Min** and **Max Time** values. This is in turn dependent on the animation's overall length, which is defined by the **Minimum** and **Maximum Time** values.

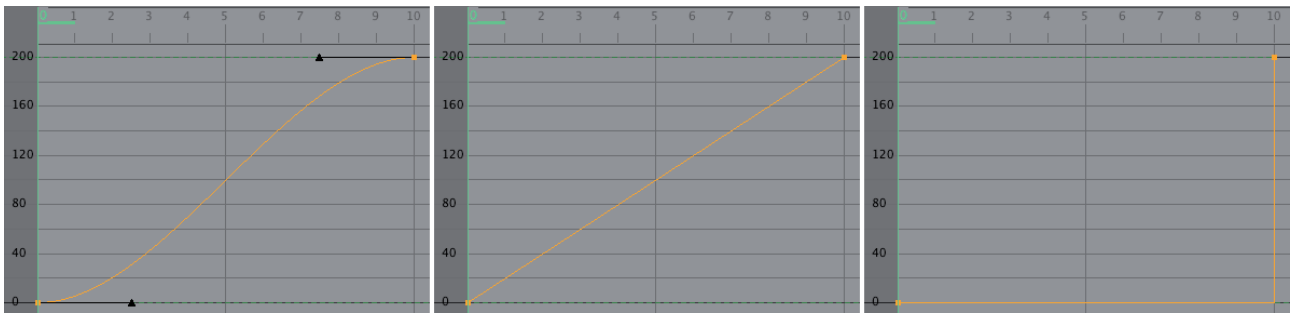
For example: Your animation will be played at 25 fps so you set the **FPS** (frame rate) value to 25. Your animation should run for a total of 10 seconds, which means it will have a total of 250 frames (25 * 10). To define this range in Cinema 4D, set **Minimum Time** to 0 and **Maximum Time** to 249. The **F** stands for 'frame'. This can be changed in the **Preferences** menu if, for example, you prefer to use **Seconds** instead.

The **Preview Min** and **Max Time** values can now be set to any range within the defined **Minimum** and **Maximum Time** values. These values only affect the range to which the **Timeline** will be zoomed in the Viewport and do not affect the animation length in any way. You can, for example, set **Preview Min Time** to 10 and **Preview Max Time** to 30 if you only want to work on the animation within this range. The animation length can be modified at any time, if necessary. To lengthen the animation, simply increase the **Maximum Time** value. The frame rate, however, should be defined very early to avoid conflicts with timing once keyframes have been set.

16.1.1 Keyframe Interpolation

We already mentioned that keyframes are tiny data packs that can be positioned at specific locations within the animation to precisely control an object's movement, for example, during the animation. During the time between keyframes, the object's motion will be interpolated, which means that a keyframe's settings will be transitioned to the next keyframe's settings to create a cohesive movement. This keyframe interpolation can be defined in advanced but can also be subsequently modified for each keyframe individually. These settings can be found in the **Project Settings** menu's **Key Interpolation** tab.

This tab's settings can be used to define how the interpolation should take place.



If **Interpolation** is set to **Spline**, an ease-in and ease-out behavior will be activated. This means that a movement will start slowly, i.e., ease in, and slow again shortly before it ends, i.e., ease out. This is very similar to real-world motion because it simulates a degree of sluggishness. Technical animations on the other hand would benefit more from **Linear** interpolation, which produces an uniform transition of keyframe values. The movement will have the same velocity from beginning to end and will also begin and end correspondingly abruptly. The **Step** interpolation method is even more extreme. It suppresses any interpolation of values between frames and produces very abrupt jumps on motion, etc. as soon as the next keyframe is reached. A typical example of how this method can be used is the second hand of a clock that jumps from one second to the next without any type of smooth transition.

Since each keyframe's interpolation can be defined individually, the **Overdub** option can be enabled, which will maintain a keyframe's interpolation method if it's overwritten. Otherwise the newly created keyframe will automatically use the method defined by the **Interpolation** setting.

16.1.1.1 Locking Keyframes

Keyframes consist of saved values that are positioned as 'keys' at specific times in the animation at which these values are applied. Each of these properties can be locked separately. If **Lock Value** is enabled, only the time at which the key is positioned can be modified. For example, the position information for a given object that is defined in this keyframe will be locked. If **Lock Time** is enabled, the position information can be modified but not the key's position in the animation.

If part of the animation has already been defined using keyframes and the timing has to be subsequently adjusted by moving a key, then the **Breakdown** option can be helpful. Keyframes that are marked with the **Breakdown** option will move as a group if a neighboring 'normal' keyframe is moved. The **Breakdown** keyframes can still be edited individually and can be adjusted along the timeline like normal keyframes. Keyframes with an enabled **Breakdown** option will be displayed with a different color in the Viewport (for position animations) and **Timeline**. If you want to apply this coloring for normal keyframes as well, enable the **Breakdown Color** option for them. This will not affect their behavior.

Selected keyframes can also be muted using the **Mute** option in the ***Attribute Manager***. This will cause the animation at this point to behave as if the keyframe does not exist. This can be helpful if you want to deactivate the animation along a certain section and, for example, calculate movements solely via Expressions.

16.1.1.2 Affecting Spline Interpolation

The following settings are only relevant if **Interpolation** is set to **Spline**. As already mentioned, a soft transition will be created between keyframe values that simulates real-world inertia. This is done by using Tangents, like the ones used for splines. A Tangent's length and pitch can be adjusted to adjust the interpolation between keyframes. Tangents have two ends that represent the time prior to and after the keyframe within the animation. If **Auto Tangents** is enabled, the Tangents will be automatically scaled and angled to create a harmonious interpolation curve. The only disadvantage to this is that the resulting curve can easily 'overshoot', i.e., the curve can be interpolated beyond the defined values of two keyframes with the same value. This behavior can be avoided by enabling the **Clamp** option. This will automatically create a linear interpolation between neighboring keyframes if their values are identical.

If the **Remove Overshooting** option is enabled, the incline of neighboring tangents will be decreased if the keyframe values are approximately the same, which in turn prevents overshooting.

The **Weighted Tangent** function automatically adjusts the length of both tangent arms in consideration of the temporal distance to the next keyframe, which produces more natural interpolation curves.

Automatic Weighting locks the tangent lengths along the X axis (in the direction of the temporal axis) and maintains this even if the tangent is subsequently rotated. This behavior simulates the calculation of tangents in other 3D applications. Automatic Weighting can, however, not be applied if Auto Tangent is enabled.

If **Auto Tangent** is disabled, the interpolation Tangents can be modified manually. If you only want to modify the Tangents' length, enable **Lock Tangent Angles**; if you only want to modify the Tangents' angles, enable the **Lock Tangent Lengths** option.

Normally, both Tangents will lie on the same line. If **Break Tangents** is enabled you can scale and angle each Tangent separately. However, this will often result in a jump or abrupt change in interpolation. Enable the **Keep Visual Angle** option to lock the angle between broken Tangents. As already mentioned, each of these options can also be modified individually for each keyframe.

16.2 The Simple Timeline

The Timeline is located directly below the main Viewport. Here you will find a slider, numeric values and icons whose functions we will describe briefly below.

As you already know from the **Project Settings** section, animations are made up of a sequence of images. The numeric values at the ends of the Timline define the temporal range of the animation in frames/images. If your animation is longer than 101 images (frame 0 is the first image), simply increase the value at the right accordingly. You can slide the green Timeslider to move to a specific location in the animation, or you can enter the frame number in the numeric field at the right end of the Timeline. The bar between the numeric values does not have to stretch over the full length between the values. You can also zoom into a section of the animation by adjusting this bar accordingly. To do so, either click and drag on one end of the bar or change one the numeric values at one or both ends.

The double arrow icons at the right of the bar can be used to jump to the beginning or to the end of the animation. The arrow icons to the left and right of the green **Play** icon will jump to the previous and next frame, respectfully. The green arrow will **Play** the animation. While the animation is playing, this icon will switch to a **Pause** icon with which the animation can be halted.

Note that the animation will generally not play as smoothly in the Viewport as the rendered animation will play. This is especially true if the scene is very complex. In such cases, it's often easier to advance the animation frame-by-frame to check the motion. Click on the red key icon to create a keyframe for the active element at the current location in the animation. Generally, an object's starting position is saved as a keyframe, a new position in the animation is selected, the object is moved and a new keyframe is set. The information that is saved in the keyframe reflects the changes that were made to the object, i.e., move, rotate, scale, etc.

If the **Move** tool is selected, a keyframe will be saved for the object's current position. If the **Scale** tool is selected, the object's size information will be saved. The scaling is done along the object's axes, which is why scaling for animation must be done in **Use Model** mode. The same principle applies to the **Rotate** tool.

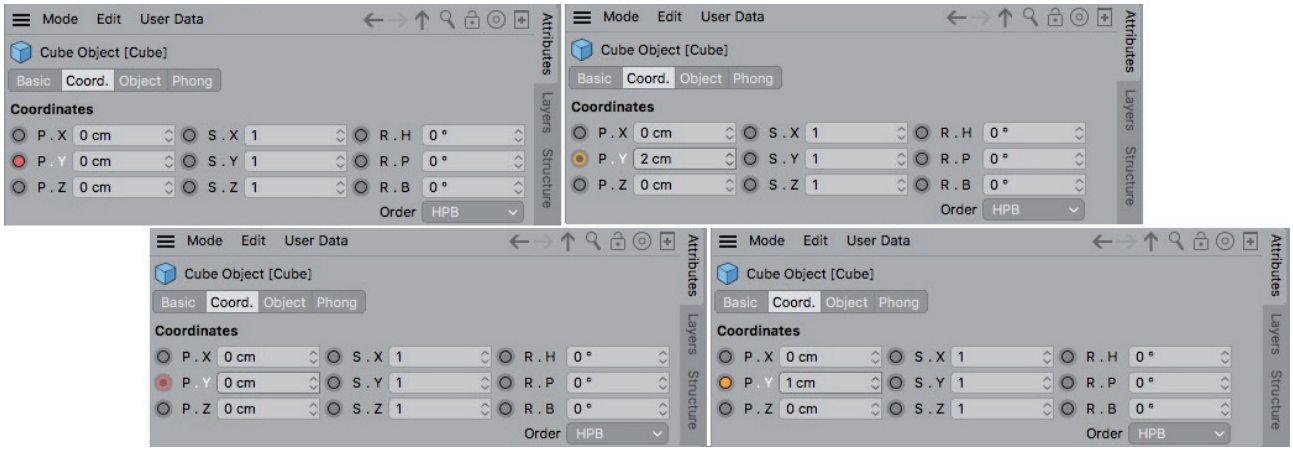
If a keyframe was set, it will appear as a small blue rectangle on the Timeline when the animated object is selected. The arrow icons with black circle next to the **Play** icon can be used to move to the neighboring keyframe.

16.2.1 Play Mode

As you already know, animations can be played by clicking on the green **Play** button. The range and sequence that will be played can be defined in the main **Animate** menu. The **Play Mode** includes an option that lets you play the **Preview Range**. Only the range defined by the **Preview Min** and **Max Time** will be played (these values can also be seed on the bar in the Timeline that lies between the **Minimum** and **Maximum Time** value fields. This lets you play only that part of the animation that you want to fine-tune. How the animation is played is defined by selecting the **Simple** (animation will be played once, then it will stop), **Cycle** (animation will automatically start again each time it reaches the end – until you stop it manually), and **Ping-Pong**, which will play the animation in reverse when it reaches the end and forward again – until you stop it manually).

16.3 Animating Settings

You already know that an object's position, scale and rotation can be animated. Keyframes can also be set for colors, materials, deformations and much more. This is what the black circles next to the respective settings are for. These can be seen next to all settings that can be animated, for example in the ***Attribute Manager***. Click on such a circle to set a keyframe for a given setting. The circle will turn into a red dot. If you move to a different location in the animation, the dot will turn into a red circle, which signals that this setting is animated but does not have a keyframe at the current location. A yellow dot or circle will be displayed if the setting's numeric value deviates from the keyframe animation.



A keyframe can be removed from a setting by simply **Shift** + clicking on it. Other keyframes will not be affected (of course the interpolation of color, etc. will be adapted accordingly). **Shift** + **Cmd/Ctrl** + click on the dot to remove all keyframes for a given setting.

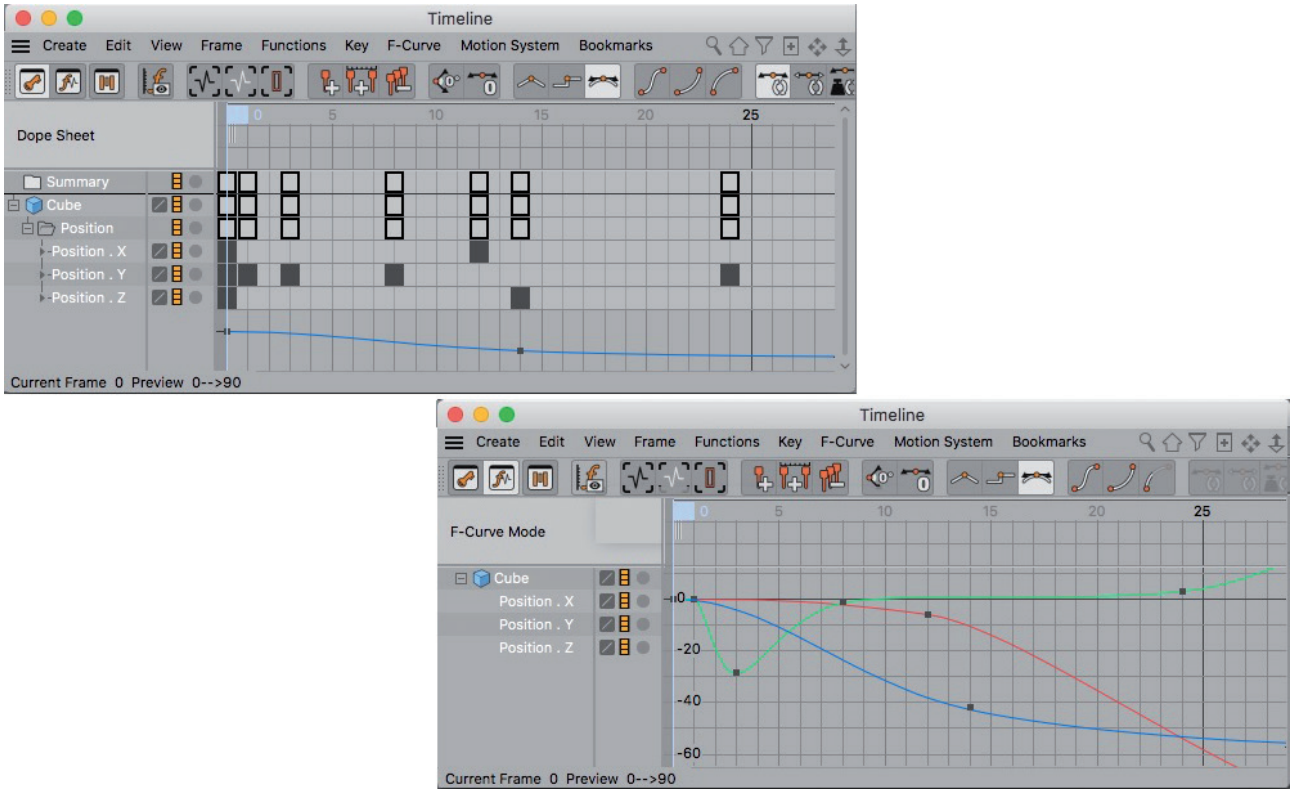
A keyframe can be corrected/replaced by simply overwriting it. Change the parameter's setting and then click again on the parameter's yellow circle.

16.4 Timeline

Working effectively with interpolation curves between splines or with keyframe Tangents is made easier with a graphic interface. Such an interface can be called up by selecting **Timeline** from the main **Window** menu. When the Timeline (Dope Sheet) is called up, the Timeline will be opened in the **Key Mode**, which offers an overview of all keyframes, including the possibility of editing them.

The animated objects are displayed in the column on the left. Clicking on the plus symbols will unfold the object's animation track hierarchy, including individual tracks for **X**, **Y** and **Z Position**, **Scale** and **Rotation**, for example. Click on the triangle next to each track to display its interpolation curve. When in **Key Mode**, individual keyframes can be selected, moved, duplicated via **Cmd/Ctrl** + drag & drop, or even deleted.

The **F-Curve Mode** offers an even easier way of editing interpolation curves.



It can be opened from the **Window/Timeline (F-Curve)** menu or by pressing the **Tab** key in the Timeline, which will switch you between Key and F-Curve modes.

Clicking on a track in the left column in F-Curve mode will display its curve in the main graph window for editing. The **1** and **2** keyboard keys or the icons at the top right of the **Timeline** window can be used to navigate in the main graph window.

▶ *See: Animation Exercises*

SUMMARY: TIMELINE

- Animations can also be defined using keyframes. Keyframes contain transformation information about a specific object and are saved at a given time during the animation.
- The time between keyframes is interpolated automatically.
- The **Spline** interpolation method simulates real-world inertia and causes objects to start and end their movements slowly.
- The **Spline** interpolation method is defined by Tangents, which can be edited using a curve in the **Timeline** window or adapted automatically to the temporal intervals or difference in values.
- The **Linear** interpolation method simulates mechanical movement and can be used to create uniform or jerky movement.
- The **Step** interpolation method applies a keyframe's settings immediately, without any interpolation at all.
- The interpolation type can be defined in the **Project Settings'** menu. You can also define the animation's frame rate using this menu.
- By clicking on the circle next to the setting's name in the ***Attribute Manager*** you can animate any setting using keyframes.

About the author:

This curriculum was written by Arndt von Koenigsmarck. Arndt is a Maxon Certified Lead Instructor and has authored books about Cinema 4D since Release 5. Since this time he has also worked with Cinema 4D as a trainer and 3D visualization specialist, and founded his own publishing house, Rodenburg Verlag (www.rodenburg-verlag.de), in 2011. Rodenburg Verlag also offers Cinema 4D plugins developed by Arndt. His German-language workshops are featured regularly at LinkedIn Learning and online sources.

Arndt has been active as an instructor at various colleges since 2010 where he manages numerous interdisciplinary projects. Arndt's wide-ranging professional experience is reflected in the structure and content of the Cinema 4D Curriculum.

Max-Planck-Str. 20 • 61381 Friedrichsdorf • Germany • Tel. +49-(0)6172-5906-0 • Fax +49-(0)6172-5906-30
maxon.net