

Rendering

This page provides general information about the Rendering rollout of Chaos Phoenix.

Overview

The rendering process in Chaos Phoenix is separated from the simulation process, because simulated caches contain only simulation data, and no render settings.

However, for ease of use, the simulator also contains a Rendering rollout, which provides flexible options for rendering the simulator grid's content, as well as sub-rollouts with shading controls for volumetrics.

The Rendering rollout offers multiple render modes, that can be divided into two types: **Volumetric** and **Surfaces**.

The volumetric modes are typically used for rendering fire and smoke, meanwhile, liquid simulations are typically rendered using one of the surface render modes.

Since **volumetric modes** do not have surfaces, their shading is described in the [Fire](#), [Smoke Color](#) and [Smoke Opacity](#) sub-rollouts, where you'll find all the parameters necessary for rendering fire and smoke as volumetrics.

The Rendering sub-rollouts enable you to shade the **Color** and **Opacity** of fire and smoke, using simple color gradients and graph diagrams that you can tweak with curves, to achieve a wide variety of different results.

Meanwhile, the **surface render modes** generate a mesh surface, which is based on the channel specified in the [Surface Channel](#) parameter.

Unlike volumetric modes, surface render modes cause the Simulator to behave just as any regular geometry. Thus, a Smoke simulation can be rendered as a mesh with a Maya or V-Ray material applied to it, and there is no need for a dedicated shader. This can be used to create more advanced effects, such as a freezing flame, cartoon-style smoke, or rendering fire as a liquid.

Note that the Fire/Smoke Simulator's Rendering rollout controls do not apply to any particles contained in the cache file.

If you want to render the **particle content** of the Simulator (e.g. **Foam**, **Splashes**, or **Drag particles**), create a Particle Shader object and add the Simulator to it, so the Particle Shader can read the cache data loaded by the Simulator. The Particle Shader can then shade the particles as either Points, Bubbles, Cellular, Splashes, or Fog, depending on the mode you select.

For even more advanced control over the shading, you can use the [Phoenix Grid Texture](#). It reads from the simulation's Grid Channels to generate a procedural texture, which can then be used to shade the simulation wherever colors are needed. The Grid Texture can be used with the volume shader to color or modulate the opacity of Fire and Smoke, using any of the supported Grid Channels (Smoke, Speed, RGB, etc.).

It can also be plugged into the texture slots of a material. For example, if you want to mix together liquids with multiple **RGB colors** emitted from different [Liquid Sources](#), the Grid Texture can be used to read and transfer the RGB colors to the Liquid mesh's material for shading.

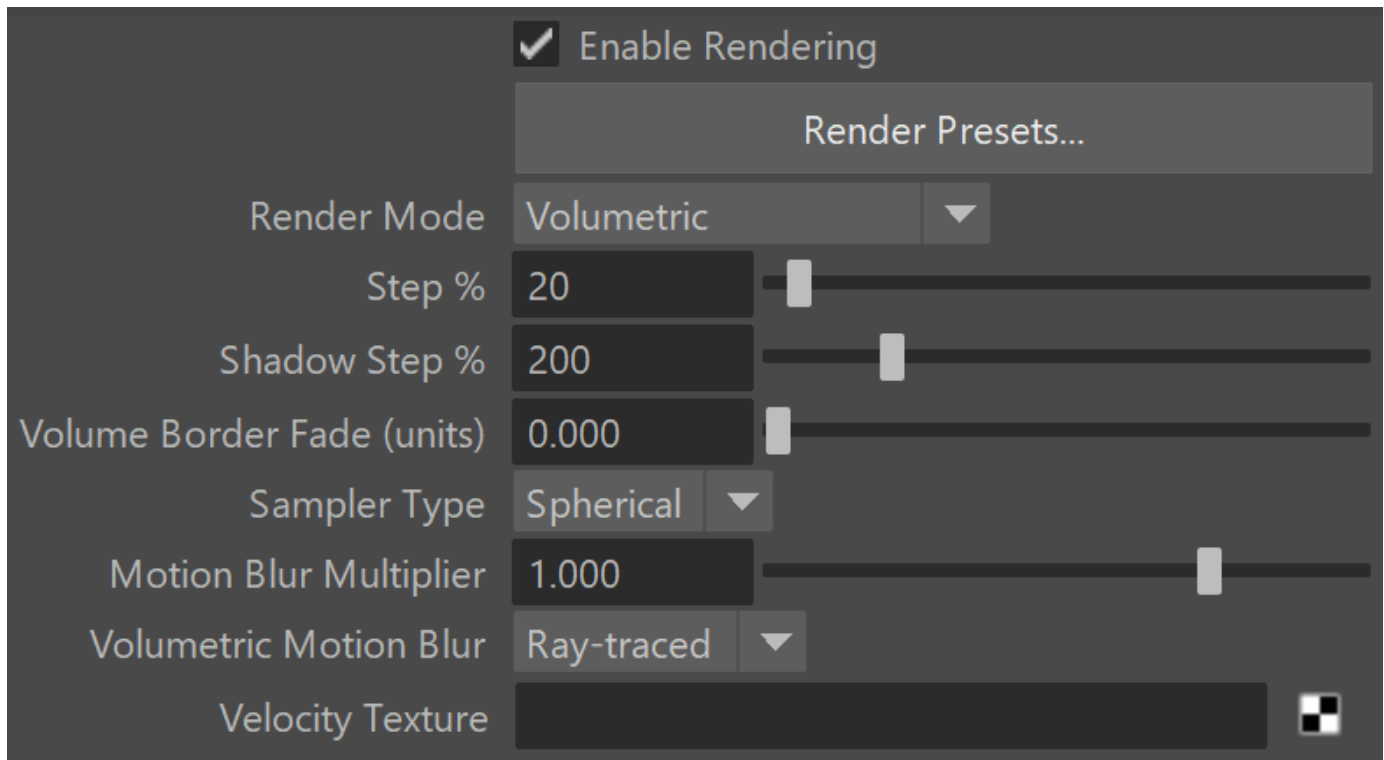
For a list of supported Render Elements, please check the [V-Ray Render Elements Support](#) page.

You may also check the Volumetric Rendering In-Depth guide for tips on speeding up the rendering of volumetric effects with V-Ray.

UI Path: `||Select PhoenixFDSim|| > Attribute Editor > Rendering rollout`

Parameters

Note that render settings are not stored within the caches themselves, so if you want to save your render settings to use with another simulator or project, you can do so from the Presets rollout.



Enable Rendering | *render* – Enables/disables the rendering of the simulator.

Render Presets... – Opens a menu for loading and saving different presets. The following options are available:

- Default Phoenix Render Settings;
- Fire/Smoke from FumeFX;
- Fire/Smoke .vdb from Houdini;
- Liquid .vdb from Houdini;
- Fire/Smoke .vdb from Maya Fluids.

Render Mode | *renderMode* – Specifies the method for visualizing the grid content. For fire/smoke, you will want to select a Volumetric method. For liquids, select a geometry method.

Volumetric – Visualizes the content similar to a [VRayEnvironmentFog](#). This method is used mostly for fire and smoke.

Volumetric Geometry – This method requires V-Ray. It produces the same result as the **Volumetric** option by using procedural geometry made up from multiple transparent layers. Used when rendering fire/smoke for exporting deep images and render elements such as normals, velocity, multi matte, etc. which would not be available in **Volumetric** mode. For more information on which render elements are supported in **Volumetric** and **Volumetric Geometry** mode, see [V-Ray Render Elements Support](#).

Approximate and **Approximate+Shadows** options for the **Scattering** parameter in the Smoke color window are not supported in **Volumetric Geometry** mode.

For a complete list of the supported **Render Elements** in both Volumetric and Volumetric Geometry mode, please check the [V-Ray Render Elements Support](#) page.

Volumetric Heat Haze – This method requires V-Ray. It produces the same result as the **Volumetric Geometry** option, and also adds a heat haze effect when used with the **Heat Haze** parameter. Note that you might need to increase the **Max depth** of a V-RayMtl with refraction in case it intersects with the Heat Haze shader.

Isosurface – This method requires V-Ray. It produces a procedural [isosurface](#) without polygons at render time using the [Surface](#) section options. Compared to the **Mesh** mode, the result is always smooth but will take longer to render. In case in **Mesh** mode your mesh is too jagged and edgy, and smoothing it out is too slow or impossible, this means you should switch to **Isosurface** mode instead.

Mesh – The content is converted into a mesh using the Surface section options. This mode is mostly used for liquids but can also be applied to thick smoke using a scatter material or to plumes of smoke to create effects such as large underwater bubbles. Depending on the renderer you are using or if you want to export the mesh to Alembic, you can change the **Mesh Type** under the Mesh rollout.

Ocean Mesh – The grid content is extended to a flat area, fitting the camera's view. In most cases, this mode is used with a displacement texture such as the [Phoenix Ocean Texture](#).

Cap Mesh – Only the upper liquid surface is rendered. This mode can be used for swimming pools and other placid liquid surfaces.

The ocean surface can be generated only when the liquid touches the sides and the bottom of the grid, which act as a container for the liquid. The detail of the mesh extension around the simulator depends on the camera resolution - for each pixel of the viewport or the rendered image, one or several polygons are generated, depending on the **Ocean Subdivisions** option.

Also note that if you use a material with fog color for the ocean mesh, and you have particles submerged below the ocean surface which you render using a Particle Shader, you need to also place a geometry that would serve as a bottom, or you could get flickering and darker rendering of the particles. For more info, see the **Render as Geometry** option of the Particle Shader.

Step (%) | *rendStep* – Specifies the ray marching step of the camera rays as a percentage of the cell size. As the renderer traces rays through the Simulator, this value controls how often to read information from the grid. If this value is more than 100, some cells will start getting skipped and artifacts may appear. Usually you don't need to lower this below 90%, unless you use render curves or texture maps for the Fire or Smoke opacity or color - in these cases you might need to lower the step in order to capture the details which are smaller than a voxel, otherwise these details will either be skipped or will render very noisy. However, decreasing the step will make the render slower. This parameter is used when **Render Mode** is set to **Volumetric**, **Volumetric Geometry**, **Volumetric Heat Haze** or **Isosurface**. See the [Step % example](#) below.

Shadow Step (%) | *shadowStep* – Specifies the ray marching step of the rays used to evaluate the lighting (shadow rays) as a percentage of the cell size. Usually, this value can be higher than **Step %**, as generally shadows will not need so much detail. Increasing the **Shadow Step %** will also speed up rendering performance, particularly with dome and area lights.

Border Fade (units) | *borderOpacityFade* – Makes the content near the grid's boundaries more transparent to prevent sharp edges from being rendered. This parameter controls how far from the boundaries the transparent effect should start, in scene units.

Sampler Type | *rendSamplerType* – Determines the blending method between adjacent grid voxels. Used to balance between rendering speed and quality. This parameter is used when **Mode** is set to **Volumetric**, **Volumetric Geometry**, **Volumetric Heat Haze** and **Isosurface** modes.

Box – Displays voxels as cubes. There is no blending between neighbor voxels. This is the fastest mode.

Linear – Linear blending occurs between neighbor voxels to smooth out the fluid's look. Sometimes this mode may unveil the grid-like structure of the fluid. Up to 20-30% faster than the **Spherical** option.

Spherical – Uses special weight-based sampling for the smoothest looking fluid. With increasing resolution, the visual advantage of this method over the **Linear** method becomes less noticeable.

Motion Blur Multiplier | *rendVelMult* – Specifies a multiplier that affects the strength of the motion blur. This value can be a negative number.

To render your Simulation with Motion Blur, you need to enable Velocity channel export from the Output tab of your simulator.

When rendering liquids, the Motion blur of the mesh is obtained by shifting each vertex along the velocity by the shutter time. If rendering a Liquid simulation with secondary particle effects such as Foam, Splashes or Mist, you would also have to enable Velocity export for each particle system under the Output rollout Output Particles tab.

Volumetric Motion Blur | *volMoblurMethod* – Specifies the type of Motion Blur that will be used.

Ray-traced – The default Volumetric Motion Blur method.

Grid-based – This method could be used instead of the default Ray-traced method in cases when you need more visible motion blur streaks, especially with faster moving fluids. The method requires a pre-pass and uses more memory.

Note that the Grid-based method only affects the Volumetric, Volumetric Geometry, and Volumetric Heat Haze Modes and it doesn't apply to Mesh, Ocean Mesh, Cap Mesh, or Isosurface Modes.

Phoenix meshes are motion blurred in a different way than regular transforming and deforming geometries. When rendering regular meshes with motion blur, the entire mesh is moved along its transformation path back and forward in time, and so each individual vertex of the mesh follows this path. However, for each rendered frame, a new Phoenix mesh must be built from the voxel grid, and so it usually has a different number of vertices than the previous and the next frame. Because of this, individual vertices can not be traced back or forward in time between frames. Instead, motion blur of fluid meshes uses the velocity of vertices which is recorded by the simulation, and moves each vertex back and forward in time along the vertex velocity. This is why the generated liquid mesh does **not** support frame sub-sampling for motion blur. This may cause a mismatch between the liquid and transforming /deforming objects in your scene that interact with it. The fluid mesh is generated from data at the exact rendered frame and fluid data for the preceding or following frames is not used, unlike regular deforming meshes. As a consequence, the liquid and the objects in your scene would synchronize best if those objects do not use additional geometry samples for motion blur.

The Grid Channel Smoothing controls allow you to smooth the Grid Channels loaded from cache files for preview and rendering.

You can smooth the Velocity Grid Channel stored in the simulated caches in case the motion blurred edges are looking jagged.

The recommended values for the smoothest result are all zeroes for the Threshold, Similarity and Random Variation options - this will produce strongest smoothing, evenly applied over the entire Grid, without adding any random variation.

Velocity Texture | *rendVelTexture* – Specifies a custom texture used for motion blur when rendering Fire/Smoke simulations. The colors of the texture represent the velocity direction for each cell, similar to how Vector Displacement textures are evaluated. Every simulator cell receives a certain color from the texture, treats it as a direction vector and stretches the contents of the simulator along to produce motion blur. This parameter can be used with a V-Ray User Color texture for motion blur in a single direction(e.g. by setting the color to (0, 1, 0) for motion blur in the +Z axis), or any other Maya texture applied as a 3D projection. The texture overrides the Velocity grid channel even if available in the cache files. Note that the Y and Z components of the velocity vectors are flipped - they are sampled respectively from the Blue and Green color values of the texture.

Example: Volumetric Motion Blur for Fire



Grid-based Volumetric Motion Blur

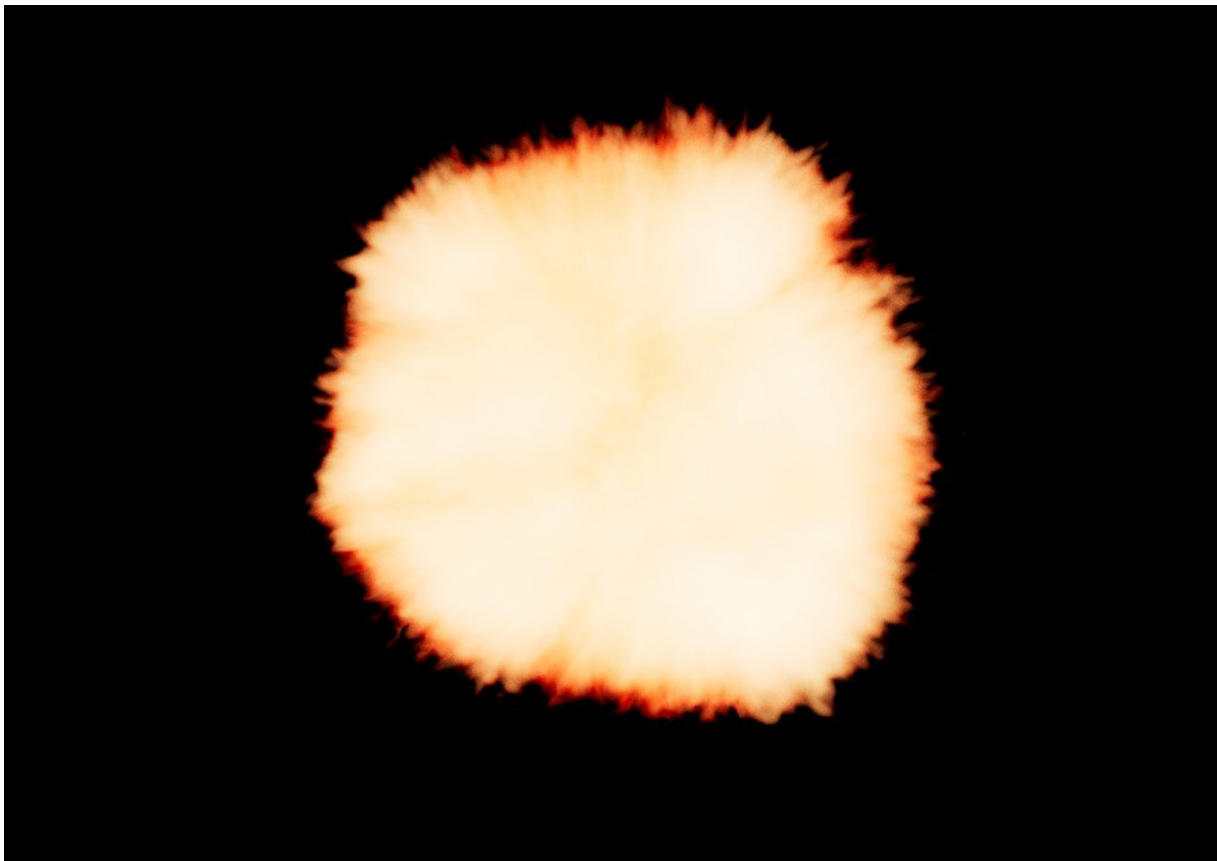


Ray-traced Volumetric Motion Blur



No Volumetric Motion Blur

Example: Volumetric Motion Blur for Explosions



Grid-based Volumetric Motion Blur



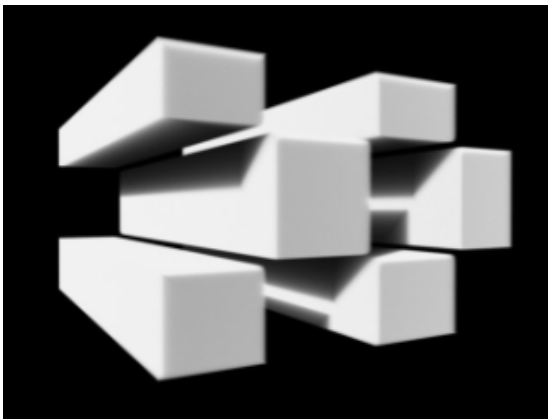
Ray-traced Volumetric Motion Blur



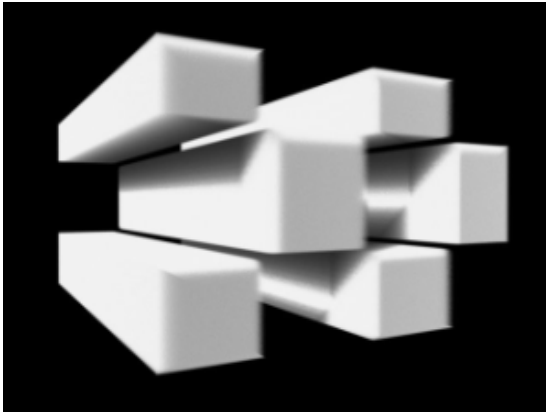
No Volumetric Motion Blur

Example: Step %

This example shows how the **Step %** value can be used to improve the quality of the ray-marching.



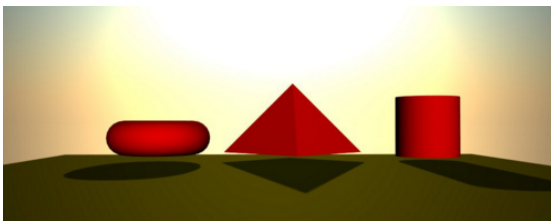
Step %: 50



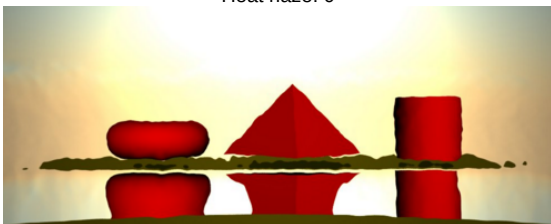
Step %: 150

Example: Heat Haze

Heat haze adds refraction at each ray-marching step through the volume. This only affects the camera's view. Heat haze will not affect shadows cast through a volume.



Heat haze: 0



Heat haze: 1

Surface

This section controls the conversion of the grid content into geometry. If **Render Mode** is set to **Mesh**, **Ocean Mesh**, **Cap Mesh** or **Isosurface**, the **Surface** channel needs to be set and an appropriate **Isosurface Level** must be chosen from this section.

The controls in this section also denote the surface used in **Gradient** and **Surface-driven** displacement. This way it can affect all render modes, not just the surface modes.

The technique for generating the surface is based on the isosurface concept. The resulting surfaces are mostly used to render liquids, but can be used for smoke and fire as well to create effects like underwater bubbles, freezing fire, etc.

Surface Channel | *sarg* – Specifies the channel that will define the surface of the fluid. It is used for solid rendering and displacement.

Texture - the values of a custom texture will define the liquid surface.

Liquid/Temperature - the Liquid/Temperature channel will define the liquid surface. Temperature is typically in the range 0-1 for Liquid simulations and 600-2000 for Fire / Smoke simulations.

Smoke - the Smoke channel will define the liquid surface. Smoke is typically in the range of 0-1 for Fire / Smoke simulations.

Speed - the Speed channel will define the liquid surface. Speed channel output has to be enabled for this to work. Speed is calculated as the length of the velocity vector for each voxel.

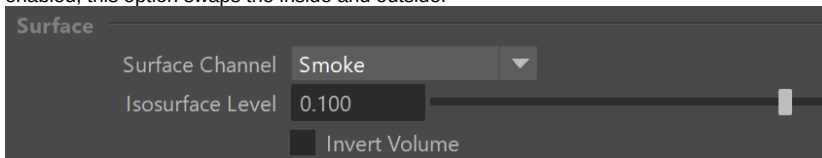
Fuel - the Fuel channel will define the liquid surface. Fuel channel output has to be enabled for this to work.

Viscosity - the Viscosity channel will define the liquid surface. Viscosity channel output has to be enabled for this to work.

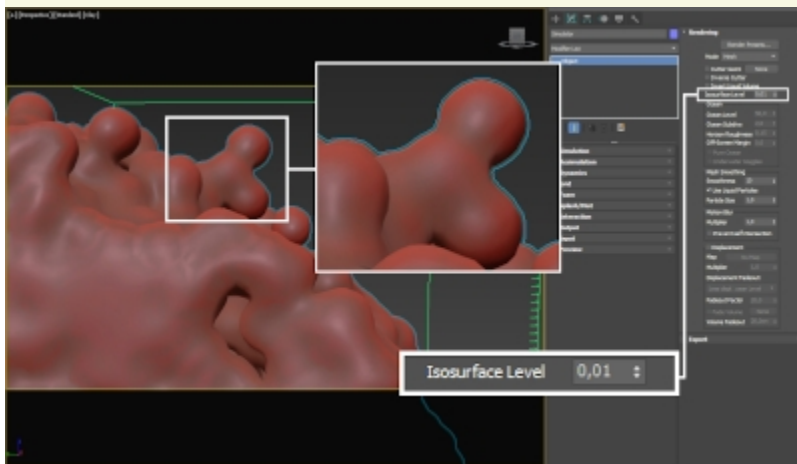
Texture | *rendEffectTex* – If the **Surface Channel** is set to **Texture**, this slot specifies the texture. In Mesh, Ocean Mesh, Cap Mesh and Isosurface render modes, the selected map will completely replace the cache files that have been loaded, if any. For more information on texture mapping in Phoenix, please check the [Texture mapping, moving textures with fire/smoke/liquid, and TexUVW](#) page.

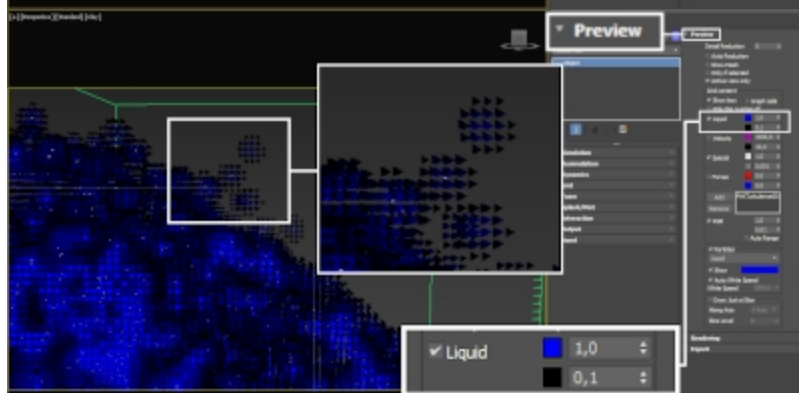
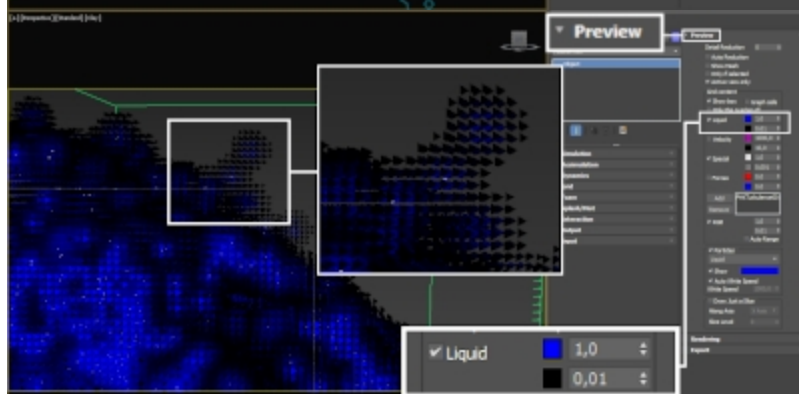
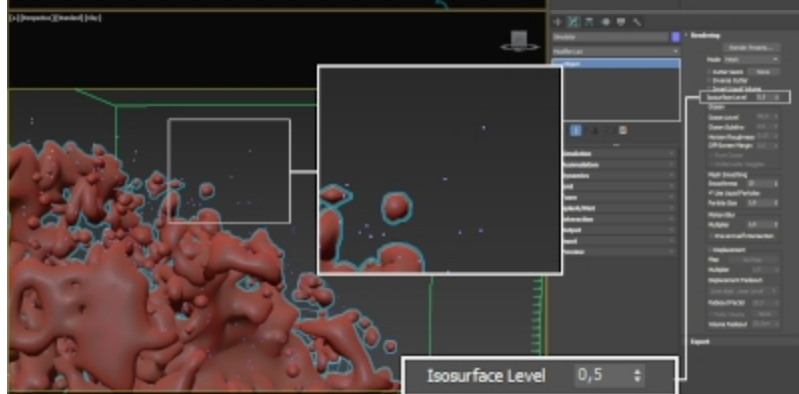
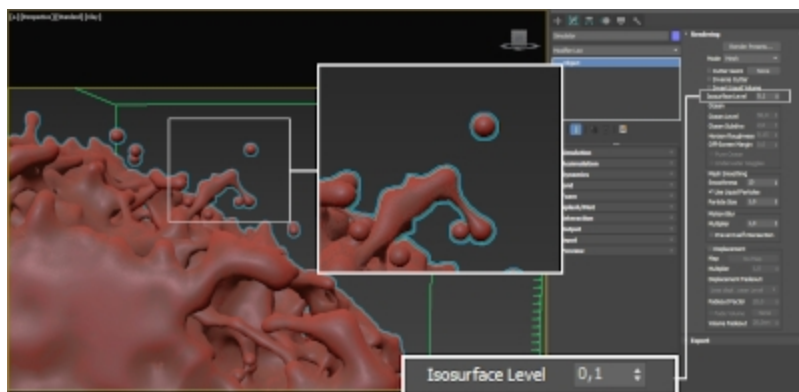
Isosurface Level | *rendSurfLevel_x*, *rendSurfLevel_t*, *rendSurfLevel_s*, *rendSurfLevel_v*, *rendSurfLevel_f* – Allows you to specify a threshold value for the generation of the geometry surface. Grid cells below this value will be ignored. By default, the Isosurface Level is set to 0.5 and should only be modified if there is flickering in the generated geometry. Isosurface Level is used only in **Isosurface**, **Mesh**, **Ocean Mesh** and **Cap Mesh** Modes.

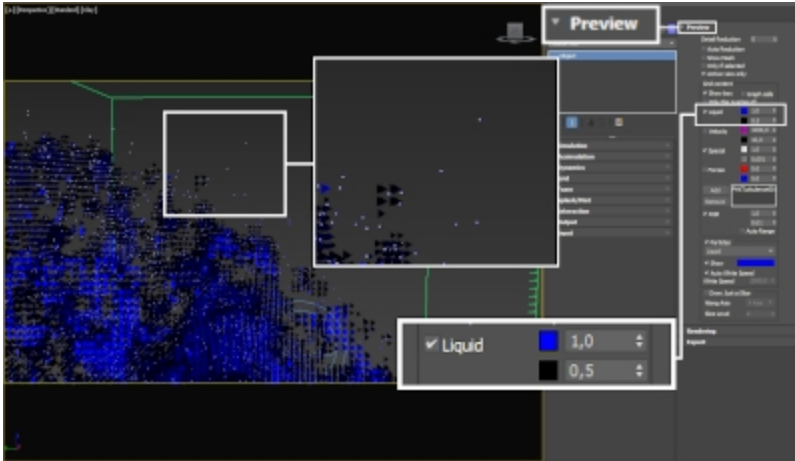
Invert Volume | *rendInvertNorm* – By default the values above the surface level are considered internal. When enabled, this option swaps the inside and outside.



The proper value for the **Isosurface level** parameter depends on the numerical range of the surface channel. For example, Phoenix liquids are kept in the range of 0 to 1. A value of 0 means there is no liquid in a certain voxel, and a value of 1 means the cell is 100% full of liquid. Values in between indicate a certain mixture of air and liquid. For such cache files, an **Isosurface level** value of 0.5 is best for visualizing the surface between the air and liquid. Imported caches from Houdini, on the other hand, use positive and negative values to indicate whether a voxel is inside or outside the liquid volume, so a correct "halfway" **Isosurface level** value would be 0.0. For Phoenix smoke, the proper value is about 0.01, and for Phoenix temperature, which is in Kelvins, the value is several hundred. Please check the Grid Channel Ranges page for information about other grid channels.







Render Cutter

If using a **Render Cutter** for a liquid pouring into a glass or otherwise contained into another refractive object, you may need to set the Render Mode to **Iso surface**. By default, the mode is set to **Mesh** which may produce artifacts in the rendered image.

Use Cutter | *rendUseGizmo* – When enabled, rendering will occur only inside the selected geometric object's volume. If Fire Lights are enabled, only those inside the cutter will be rendered. Note that the **Render Cutter** will not work when the **Render Mode** is set to **Volumetric Geometry**.

Set Selected Object as Render Cutter - When a polygon mesh and a Phoenix Simulator are selected, the selected mesh will be used as the render cutter for the geometry generated by the simulator.

Invert Cutter | *rendInvGizmo* – When enabled, rendering will occur only outside of the render cutter. This is not the same as a cutter with inverted geometry because any rays that do not intersect the cutter will be shaded as well.

Cutter Geometry | *rendGizmo* – Specifies a polygon geometry that will clip the rendering only to inside its volume.

