

# FireSmoke Rendering

This page provides information on the Rendering rollout for a Fire/Smoke Simulator and how the grid's content is rendered and controlled.

## 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 Fire/Smoke Simulator object also contains a Rendering rollout, which provides flexible options for rendering the simulator grid's content, as well as 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, 3ds Max and V-Ray materials cannot be used for shading. Instead, their shading is described in the **Volumetric Render Settings** window, which you can open with the **Volumetric Options...** button.

The Volumetric Render Settings 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** parameter dropdown.

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 polygon mesh with a 3ds Max 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 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 Fire Smoke Simulator object|| > **Modify panel > Rendering rollout**

## Actions

Volumetric shading settings are not stored within the caches themselves, so if you want to use the same render settings for another simulator or project, there is the option to save and load them as **Phoenix Render Presets** in the ".tpr" file format.

**Expand** – Opens a floating dialog that contains the selected rollout and automatically folds the command panel rollout.

**Re-Center** – Resets the position of the floating rollout.

**?** – Opens up the help documents for the Fire/Smoke Rendering.

**Render Presets...** – Opens a menu for loading and saving different presets, which contain the Phoenix Rendering Rollout settings for future use. The following options are available:

- Load from File...
- Save to File...
- Default .aur Render Settings
- Fire/Smoke .aur Fast Render
- Fire/Smoke from FumeFX
- Fire/Smoke .vdb from Houdini
- Liquid .vdb from Houdini
- Fire/Smoke .vdb from Maya Fluids
- Fire
- Fuel Fire
- Gasoline Explosion
- Explosion
- Large Smoke
- Candle
- Clouds
- Cold Smoke

**Mode** | *rendmode* – Specifies the method for visualizing the grid content.

**Volumetric** – Visualizes the content as a standard 3ds Max atmospheric. 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 VRayMtl 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 **Iso surface** mode instead.

**Mesh** – The content is converted into a standard 3ds Max 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.

**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 FD 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.

**Volumetric Options...** – Opens the Volumetric Render settings window which contains the following rollouts:

- [Fire](#) – Controls the emissive (fire) color of the volumetric shader, and the light emitted by the Simulator.
- [Smoke Color](#) – Controls the diffuse color of the volumetric shader.
- [Smoke Opacity](#) – Controls the transparency part of the volumetric shader.

The **Volumetric Render Settings** provide Fire and Smoke shader controls that are designed to be straightforward and easy to use.

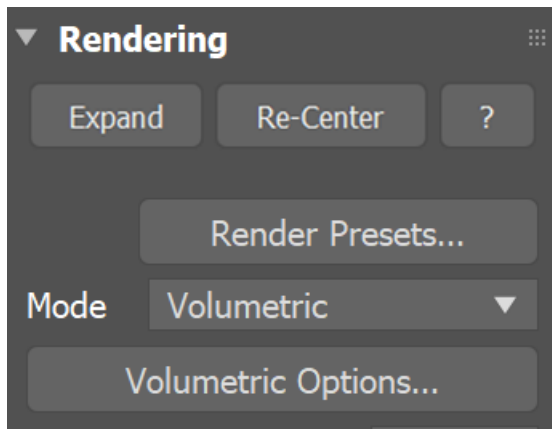
For example, there are simple color gradients and graph diagrams that you can tweak using curves, which also give enough flexibility to achieve a wide variety of different results. If you need more advanced controls, like the ability to multiply the data with textures, or mix a few grid channels, you can also do that with the available options in Phoenix as well.

These controls make it possible to shade a variety of different types of fire, smoke, and other volumetric scenarios, from candle flames to fireplaces, to cigarette smoke or massive explosions, as well as sci-fi content such as nebulae, and other effects entirely up to your imagination.

For even more advanced control over the shading, you can use the Phoenix Grid Texture.

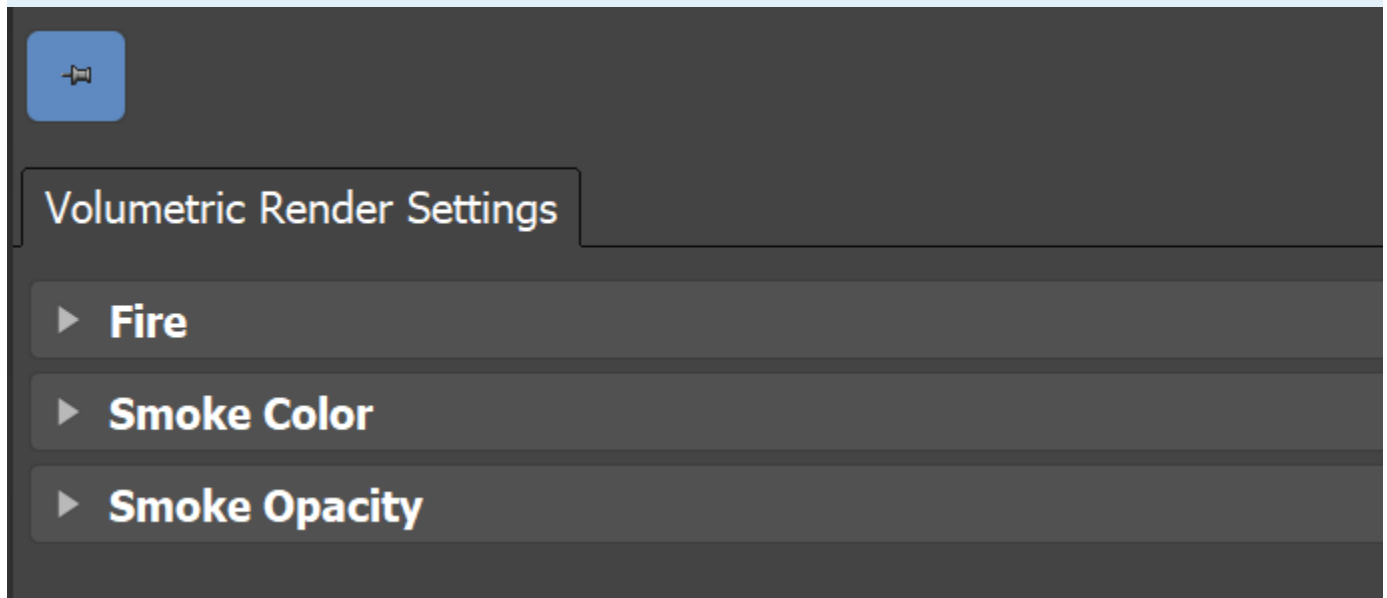
It reads from the simulation's **Grid Channels** to generate a procedural texture, which can then be used to shade the simulation wherever colors are needed. The Grid Texture can be used with the volume shader to color or modulate the opacity of Fire and Smoke, using any of the supported Grid Channels (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.



When the Pin icon is enabled, the Volumetric Render Settings window will remain open even after deselecting the Simulator, Voxel Shader, or V-Ray Volume Grid.

The option's default state is set to "Pinned", but if you need to change its behavior, you can do so from the Phoenix FD Global Preferences menu.



## Parameters

**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 **Mode** is set to **Volumetric, Volumetric Geometry, Volumetric Heat Haze** or **Isosurface**. See the [Step % example](#) below.

**Shadow Step %** | *rendshadstep* – 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** | *borderFade* – 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.

**Heat Haze** | *hhfactor* – Traces the ray changing direction according to the gradient of the **Surface** channel. A value of 1 corresponds approximately to the normal heat haze in the air caused by the temperature. If smoke or another channel is selected as the source, a larger value might be necessary to achieve a visible result. A value of 0 produces no heat haze. This parameter is available only when **Mode** is set to **Volumetric Heat Haze**. See the [Heat Haze example](#) below.

**Sampler Type** | *sampler* – 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.

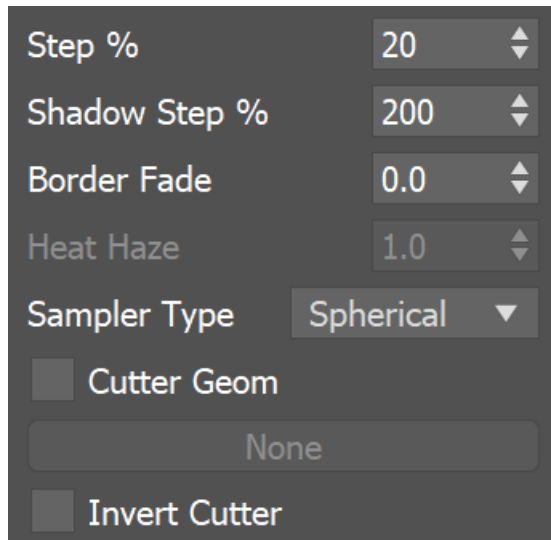
**Linear** – Linear blending occurs between neighbor voxels to smooth out the fluid's look. Sometimes this mode may unveil the grid-like structure of the fluid. Up to 20-30% faster than the **Spherical** option.

**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.

**Cutter Geom** | *usegizmo, gizmo* – 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 **Cutter Geom** will not work when the render **Mode** is set to **Volumetric Geometry**.

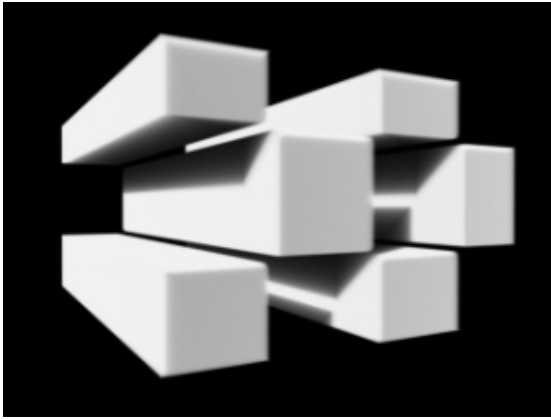
**Invert Cutter** | *invgizmo* – When enabled, rendering will occur only outside of the render cutter. This is not the same as a cutter with inverted geometry because any rays that do not intersect the cutter will be shaded as well.

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

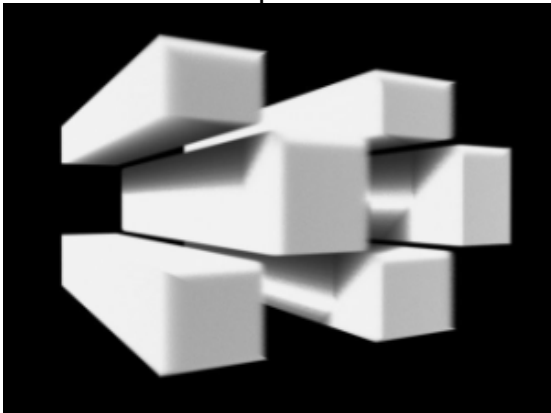


### Example: Step %

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



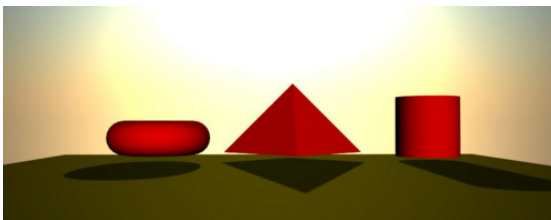
Step %: 50



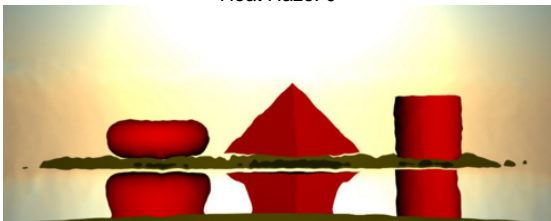
Step %: 150

### 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 Channel

This section controls the conversion of the grid content into geometry. If **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 mesh surface. You can see how this works in [this How-to video](#).

**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.

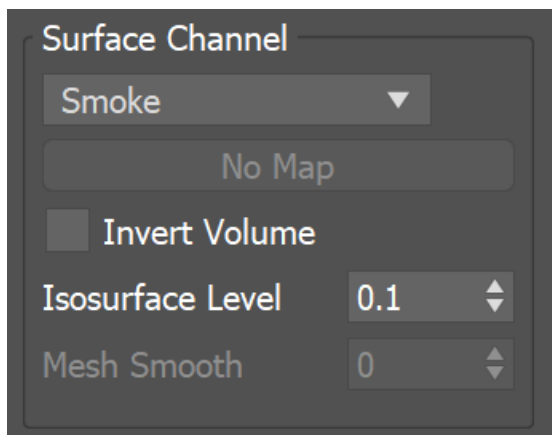
**Fuel** - the Fuel channel will define the liquid surface. Fuel channel output has to be enabled for this to work.

**Texture** | *stex* – 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.

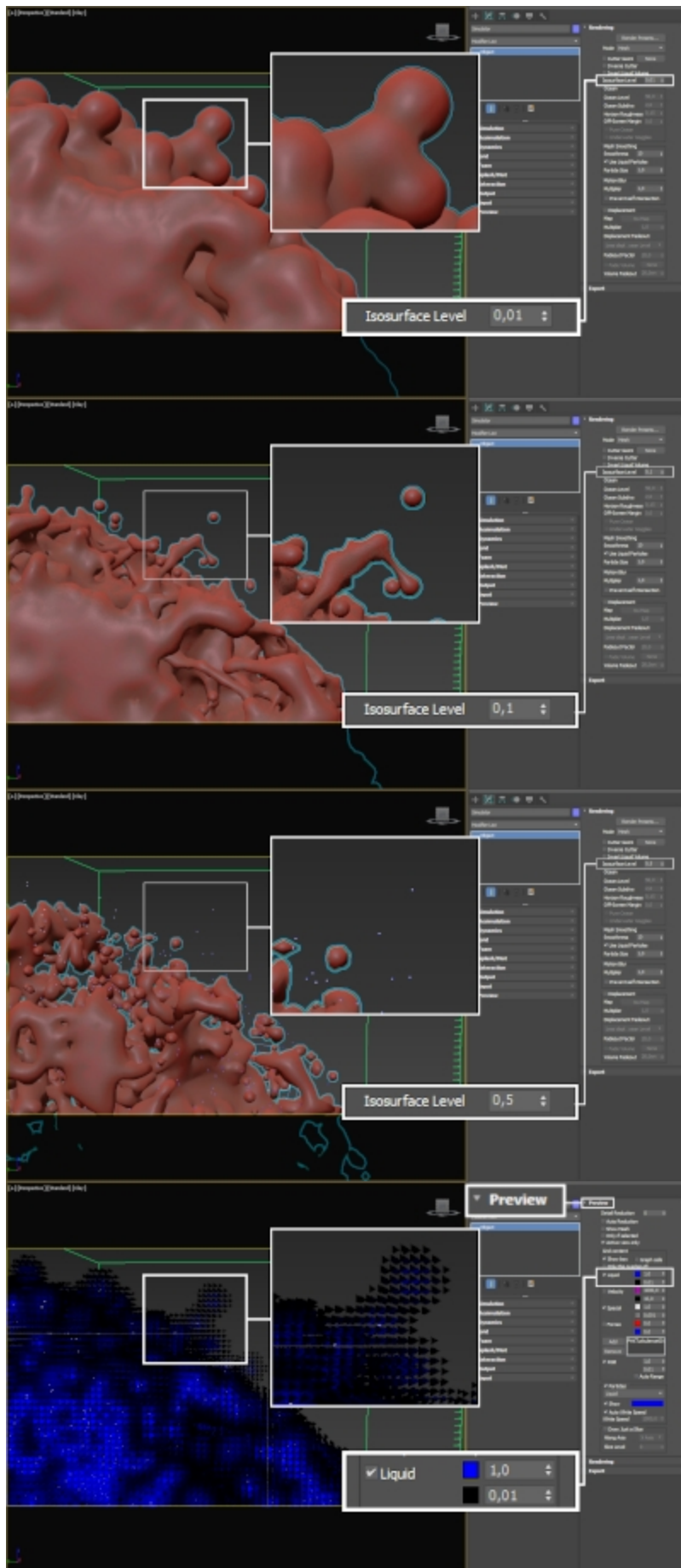
**Invert volume** | *solidbelow* – By default the values above the surface level are considered internal. When enabled, this option swaps the inside and outside.

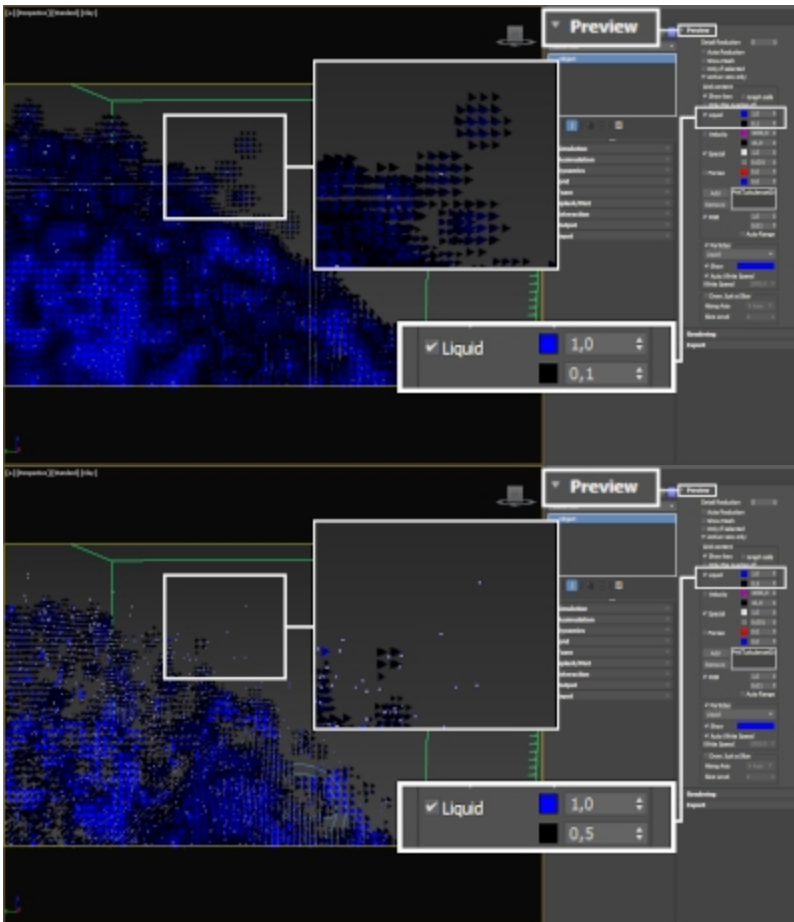
**Isosurface Level** | *surflevel* – Allows you to specify a threshold value for the generation of the geometry surface. Grid cells below this value will be ignored. Isosurface Level is used only in **Isosurface**, **Mesh**, **Ocean Mesh** and **Cap Mesh** Modes.

**Mesh smooth** | *smoothmesh* – Used when **Mode** is set to **Mesh**, **Ocean Mesh**, or **Cap Mesh** to reduce the roughness of the mesh. The larger the value, the slower the mesh builds.



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.





## Motion Blur

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

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

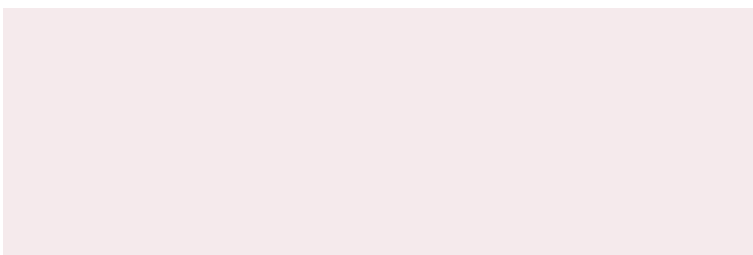
**Volumetric Motion Blur** | *vol\_mobblur\_method* – Specifies the type of Motion Blur that will be used.

**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.

**Prevent Self Intersection** | *mbself* – The motion blur of the mesh is obtained by shifting each vertex toward the velocity by the shutter time. In certain cases the shifted vertex may penetrate the opposite side of the geometry, causing problems in the rendering. When enabled, this option prevents such situations. The self-intersection analysis is expensive, so enable this parameter only when the issue is noticeable.



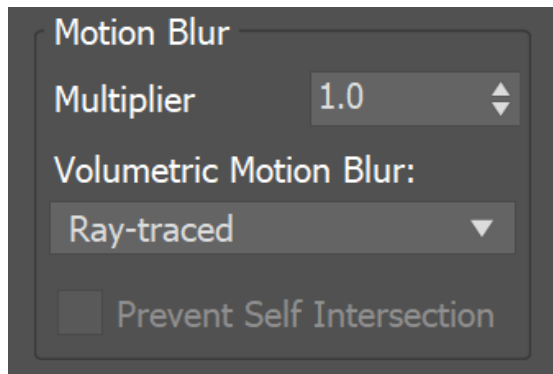


Phoenix meshes are motion blurred in a different way than regular transforming and deforming geometries. When rendering regular meshes with motion blur, the entire mesh is moved along its transformation path back and forward in time, and so each individual vertex of the mesh follows this path. However, 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.



---

### Example: Volumetric Motion Blur for Fire



*Grid-based Volumetric Motion Blur*



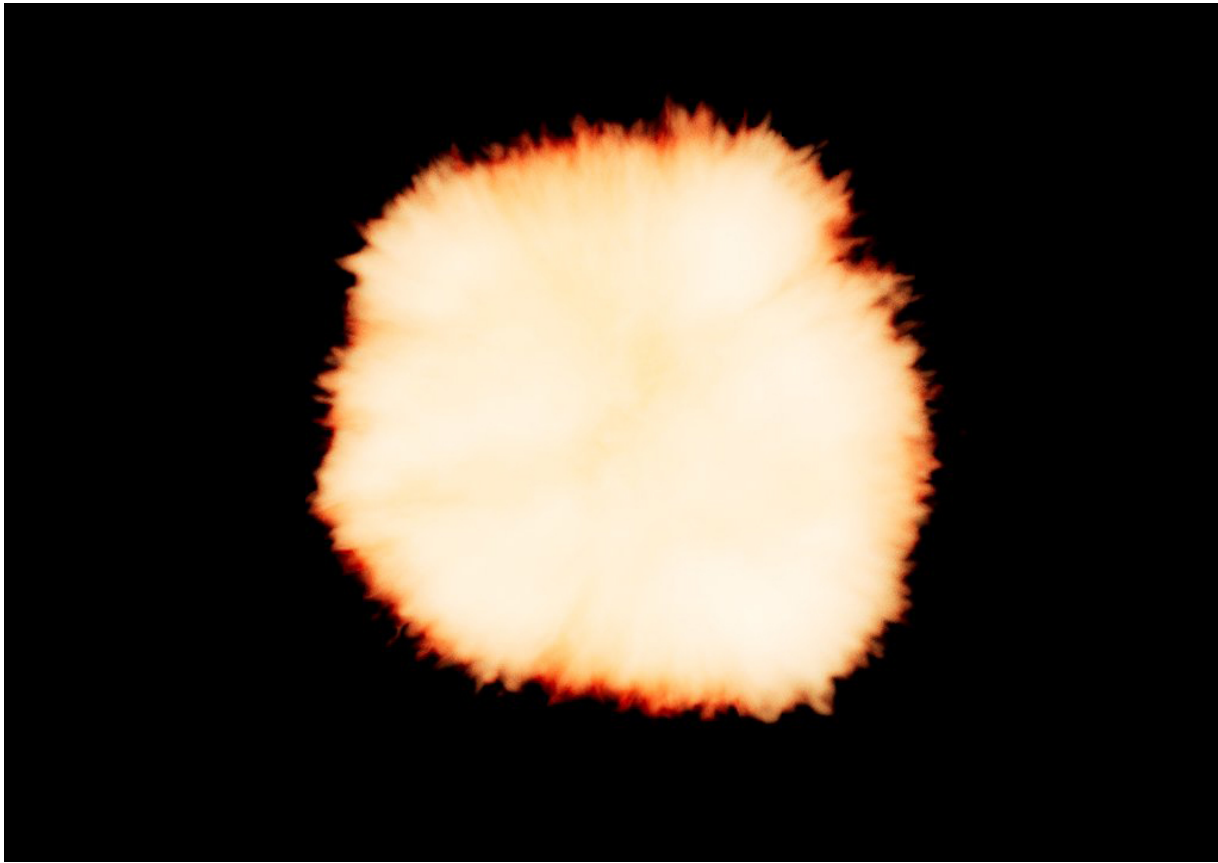
*Ray-traced Volumetric Motion Blur*



*No Volumetric Motion Blur*

---

**Example: Volumetric Motion Blur for Explosions**



*Grid-based Volumetric Motion Blur*



*Ray-traced Volumetric Motion Blur*



*No Volumetric Motion Blur*

## Displacement

Displacement is a technique intended to add detail to the simulation during the rendering. The idea of the Phoenix displacement is similar to the usual geometry displacement: a texture is sampled, and the corresponding point of the fluid volume or surface is shifted in a direction at a distance determined by the texture. You can plug any V-Ray, 3ds Max or Phoenix texture maps.

You can use the Phoenix Simulator's Mesh Preview option to check how the attached displacement map is affecting the surface when **Mode** is set to **Mesh**, **Ocean Mesh** or **Cap Mesh**.

**Enable** | *displacement* – Enables the displacement effect, and enables the use of a multiplier value to increase or decrease the overall displacement effect.

The **Type** parameter is ignored when **Mode** is set to **Mesh**, **Ocean Mesh** or **Cap Mesh**. In these modes Phoenix automatically recognizes whether the texture map is monochrome or colored and uses respectively **Surface-driven** or **Vector** displacement.

The difference between **Surface driven** and **Vector** displacement is that vector displacement can produce more complicated surfaces. For example, a wave texture in **Vector** mode produces waves that have a convex back side and a concave front side, in contrast with the symmetrical forms produced by **Surface driven** displacement.

**Multiplier** | *displmul* – Specifies the displacement amount.

**Type** | *displ2d* – Specifies the displacement technique.

**Gradient driven** – Requires a monochrome texture map. The displacement amount is the texture's brightness at each point. Each point of the fluid is shifted along the gradient of the **Surface** channel. This means that each point in space could have a different displacement direction. This method is suitable for smoke and fire.

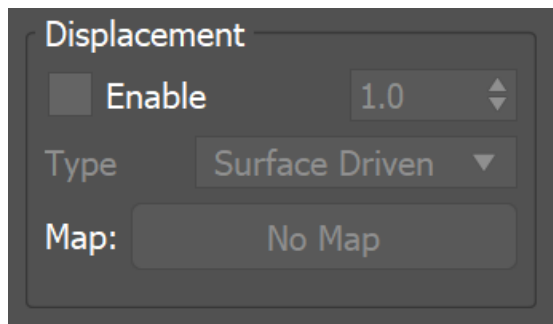
**Surface driven** – Requires a monochrome texture map. The displacement amount is the texture's brightness at each point. Each point of the fluid is shifted along the normal of the point's projection on the isosurface of the fluid's **Surface** channel. The texture is also sampled at the projection point. Unlike the **Gradient driven** displacement, this ensures that all points above of below the fluid surface will be displaced in the same direction, and so displacing fire/smoke simulations produces better results that are more similar to displaced solid geometry surfaces. However, the **Surface driven** method is slower than **Gradient driven**.

**Vector** – Requires a colored vector texture map (with negative and positive values). The point is shifted by the texture color, interpreted as a 3D vector. This displacement mode is intended to be used with the [Phoenix FD Ocean Texture](#) but can be used with any other vector displacement texture.

- If **Mode** is set to **Mesh**, **Ocean Mesh** or **Cap Mesh**, then it requires a texture in the format used for V-Ray Tangent Vector displacement, where X and Y of the texture are 0.5-based, and the Z direction is 0.0-based. This means that if you use a texture where the Red and Green colors are gray and the Blue color is black, it will produce no displacement; brighter color than these will move the fluid points along the positive axes, and darker and negative colors will displace the fluid point along the negative axes. A texture in such a format is the [Phoenix Ocean Texture](#) in **Vector Mode**.
- If **Mode** is other than the mesh modes, **Vector** displacement requires a texture which is 0.0-based, so black color means no displacement, brighter colors shift the fluid points towards the positive axes and negative colors - along the negative axes. Such a texture is the [Phoenix Grid Texture](#) with its **Channel** set to **Velocity**.

**Advection** – Requires a colored 0.0-based vector texture map (with negative and positive values). A very similar method to **Vector**, but does not produce grainy structures for fire and smoke. Can be combined with the [Phoenix Grid Texture](#), with its **Channel** set to **Velocity**, to produce render-time gridless advection. *For more information, see the [Advection Displacement example below](#).*

**Map | disp/2** – Specifies the displacement map. Depending on the **Type** option selected, a monochrome map or a color map could be required. If a colored map is specified when a monochrome map is needed, the strength of the displacement is determined by the total intensity of the color. If a monochrome map is specified when a vector map is needed, the entire displacement will point in a single direction. For more information on texture mapping in Phoenix, please check the [Texture mapping, moving textures with fire/smoke/liquid, and TexUVW](#) page. See the [Advection Displacement with a Monochrome Map example below](#).



**Example: Advection Displacement**



Regular smoke and fire, 5M cells

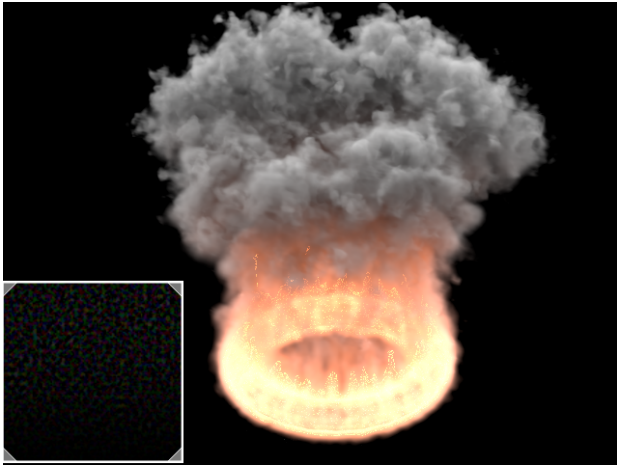


Advection displacement with the simulation's own velocity, using a PhoenixFDGridTex and multiplied by a noise map.

### Example: Advection Displacement with a Monochrome Map

This example illustrates how displacement is affected when a monochrome map is passed when a vector map is needed.





Advection Displacement with a vector map between -1 and 1



Advection Displacement with a monochrome map between 0 and 1  
Note that the displacement effect points in a single direction.