Liquid Source

This page provides information about the LiquidSrc component in Phoenix for 3ds Max.

Overview

The **Source** controls the emission of fluid and where the fluid emits from, so that the simulator knows where in 3D space the fluid can be born.

An **Emitter** can be geometry and/or particles, and is what actually emits the fluid inside the simulation grid. The emitter(s) must be selected by the Source in order to emit fluid inside the simulator, unless you are using the **Initial fill** option in the simulator's settings, or specifying a geometry to do an initial fill with liquid.

The Source also contains its own settings that determine how much fluid is emitted, what is emitted, and so forth. A **Liquid Source** can be used to emit fluid into a Liquid Simulator using Liquid FLIP particles, or any of the Source's **Grid Channel** options, and it can also emit fluid for multiple channels at once.

This includes emitting options such as **RGB color** and **Viscosity**, as well as emitting secondary particle types like **Splashes, Mist** and **Foam**.

If you have many Simulators in the scene, by default each Simulator will interact with a Source's **Emitter Nodes**, as long as they are inside that Simulator. You can exclude Sources or Emitters from a Simulator's Scene Interaction rollout.

If you switch to use the **Include List** mode, you have to pick both the Source and its Emitter Nodes in the **Interaction List**.

The Source can emit in three different **Emit Modes**:

- **Surface Force** creates fluid only at the surface of emitters
- **Volume Brush** fills the entire volume of emitters
- **Volume Inject** fills an emitter's volume while adding pressure for an explosive effect

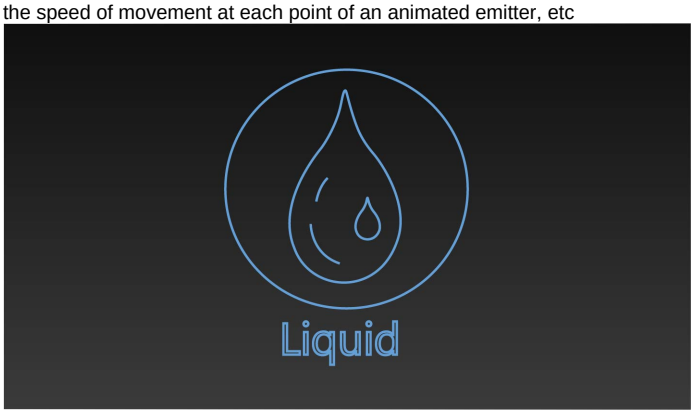
You can also use **textures as masks** for each of the emission channels, to create more interesting emission behaviors, with more variation. Specifically, **Masks** make it possible to emit unevenly from only some areas, or emit unevenly from the entire volume of an emitter.

Using **textures as masks** can help to break up the emission, and can lead to a more varied or natural looking result.

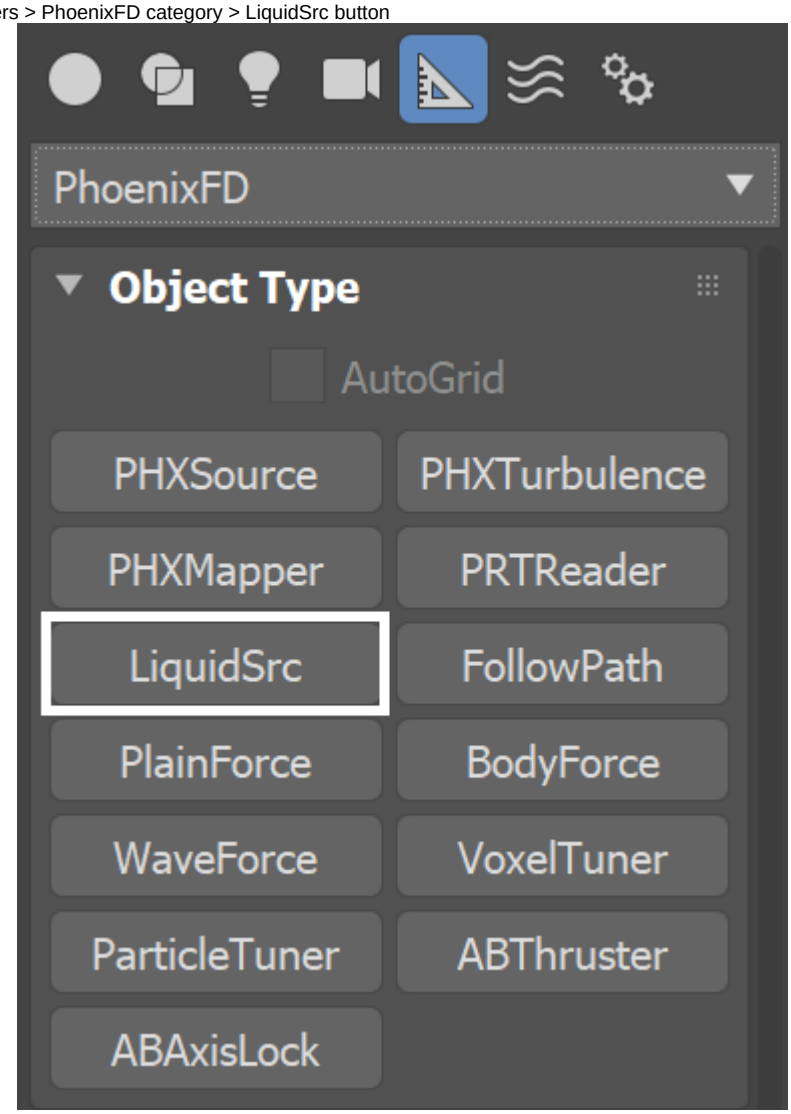
For example, you could use a black and white noise texture as a **Mask** when emitting **Liquid particles**, to make it so that the black parts of the texture emit nothing, while the white parts emit Liquid.

Additionally, each channel can have one or many Discharge Modifiers. They can give you more precise procedural control over how the fluid gets emitted.

Discharge modifiers vary the emission over different parts of the emitter, depending on the properties of the emitter - e.g. the direction of its Normals, the speed of movement at each point of an animated emitter, etc



UI Path: ||Create panel|| > Helpers > PhoenixFD category > LiquidSrc button



Parameters

Emitter Nodes

Non-Phoenix particles, such as **Particle Flow** or **tyFlow particles**, can also act as emitters for a Phoenix Source. They can emit from a spherical 3D shape, or from instanced geometry.
Note that the **Source icon** itself does not emit fluid, so the position of the icon's viewport gizmo in the scene does not matter.

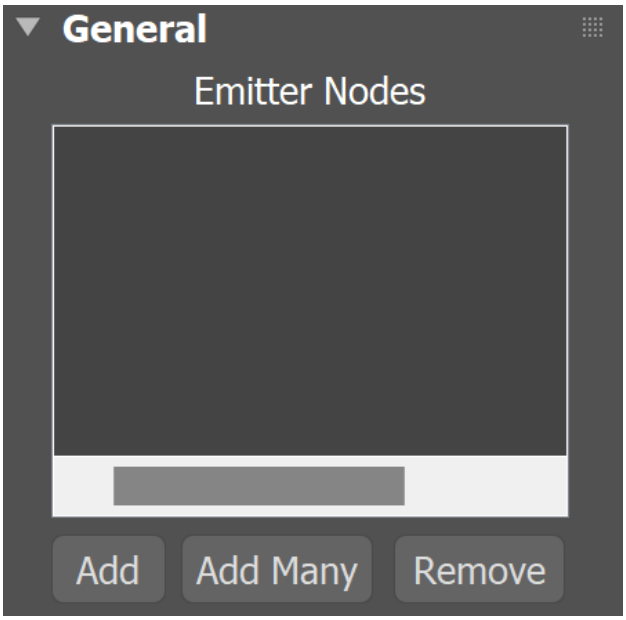
Instead, you must pick the geometry and/or particles that you want to use as emitters, in the Source's **Emitter Nodes** list.

Emitter Nodes | *sources* – Specifies a list of objects that will emit fluid. Both geometry and particle systems can be selected here. Press the Add button and pick an object from the Viewport, or a list of objects using the Scene Explorer.

Add – Adds the selected object or a particle system as an emitter.

Add Many – Adds many objects to the emitter list allowing to quickly add a list of nodes.

Remove – Removes the selected emitter nodes from the Source.



General

Fluid can be emitted from a geometry's surface, or from the entire volume of an emitting geometry.

Note that if the **Emit Mode** is set to **Volume Brush** or **Volume Inject**, and you have a **Texture Mask** using either *Explicit Map Channel* or *Vertex Color Channel* mapping, then the **Mask** will be applied on the whole volume, based on the closest geometry surface.

Even if you do not simulate visible fluid like liquid, there can still be **Velocity** simulated within the grid, if for example, you animate an object to move around inside the grid to stir the Velocity channel. The simulated velocity can also be previewed in the viewport, or even rendered.

Emit Mode | *ifnotsolid* – Specifies the way the objects in the Emitter Nodes e mit fluid.

> **Surface Force** – The surface of the emitters will eject the selected fluid channels along the geometry normals. In this mode, the discharge is named **Outgoing Velocity** and it specifies the speed of the emitted fluid in units/sec. The displayed units will change accordingly if the scene units change.
>
> This mode can work with both Solid and non-Solid emitters. If you use a **Mask** for the discharge in **Surface Force** mode, white areas of the emitter's surface will eject fast fluid, while darker ones would emit more slowly. Black areas will not emit at all.
>
> **Volume Brush** – The fluid inside the volume of the emitters will gradually change towards the selected channel values. When this mode is selected, the discharge is named **Brush Effect** (%) and it specifies the rate at which the transition takes place. When **Brush Effect** is 100%, the fluid will immediately reach the selected channel values, and if **Brush Effect** is less, it specifies how close the fluid values will get to the values from the Source over 1 second. E.g. if the Temperature inside an emitter's volume is 1000 and the Source emits Temperature 2000 with Brush Effect of 80%, then after 1 second the temperature will have risen to 1800. Note that the **Volume Brush** mode can both increase or decrease the fluid values, for example if the Smoke in the Source is set to 0.5, the Source would increase the smoke in voxels that have 0.0 smoke until it reaches 0.5, but will decrease the smoke in voxels having 1.0 smoke, again - until it reaches 0.5. This mode is useful for creating standing volumes of fluid with a high Brush Effect, or alternatively - to slowly convert the fluid inside the volume of the emitters to the values selected below over a period of time. Note that you can both increase or decrease the values of the fluid channels in **Volume Brush** mode. When **Brush Effect** (%) is 0, then the Source has no effect.
>
> This mode requires that all selected emitters are set into non-Solid mode from their Per-Node Properties. If you use a **Mask** for the discharge in **Volume Brush** mode, white zones in the volume will have the **Brush Effect** you have specified, while darker zones will use a smaller Brush Effect. Completely back zones in the mask would not be affected at all by this Source.

Volume Inject – The volume of the emitters will discharge the selected fluid channels with added pressure. When this mode is selected, the discharge is named Inject Power and it specifies the added volume of the injected fluid per second. This mode is useful for getting explosive discharge. **Inject Power** can be negative, in which case the Source will suck in and delete the fluid.

This mode requires that all selected emitters are set into non-Solid mode from their Per-Node Properties. If you use a Mask for the discharge in **Volume Inject** mode, white zones in the volume will have the Inject Power you have specified, while darker zones will use a smaller **Inject Power.** Completely back zones in the mask would not be affected at all by this Source.

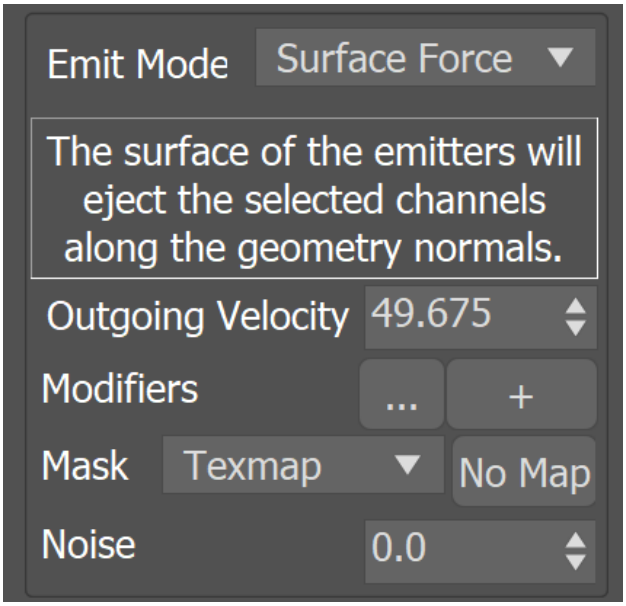
When emitting from **particles** in any of the 'Sphere' **Prt Shape** modes, the **Surface Force** Emit Mode is **not supported** - Phoenix will automatically fall back to **Volume Inject** mode. Only **Use Particle Shape** supports all 3 emit modes.

Inject Power / Brush Effect (%) / Outgoing Velocity | *discharge / brusheffect / outvel* – These parameters control the strength of the source. Check Emit Mode for more info.

Mask | *dmap, usedmap* – Allows you to vary the **Outgoing Velocity, Inject Power** or **Brush Effect (%)** over the surface or the volume of the emitters. White areas of this map will have the strongest discharge, while black areas of the map will not discharge at all. The individual fluid channels can also be modulated using dedicated maps from the options below. See the info on the **Emit Mode** option above for more info on how the Mask affect s each mode.

Modifiers | *dmoddisch* – Discharge Modifiers can be attached here in order to affect the **Outgoing Velocity, Inject Power** or **Brush Effect (%)** parameters.

Noise | *noise* – Varies the **Outgoing Velocity, Inject Power** and **Brush Effect (%)** across the surface or the volume of the emitting geometry or particle. The variation also changes over time. This is a shorthand for using an animated noise in the Mask slot.



Liquid & RGB

Emit Liquid – Enables or disables liquid emission. Disabling this option is useful in situations where only foam and splashes need to be emitted but not the liquid itself.

RGB | *uvw, useuvw* – If the RGB Map is not enabled, the emitted fluid's RGB channel will contain the specified color. If the RGB Map is enabled, the RGB values from the texture map will be used instead of the color swatch. If the RGB channel is not enabled in the Output rollout of the Simulator, this parameter will be ignored. Also, note that if **Emit Mode** is set to **Volume Brush** or **Volume Inject** and the **Map** uses *Explicit Map Channel* or *Vertex Color Channel* mapping, then the **Map** will be applied on the whole volume, based on the closest geometry surface.

Modifiers | *dmodrgb* – A discharge modifier can be attached here in order to affect the **RGB** parameter.

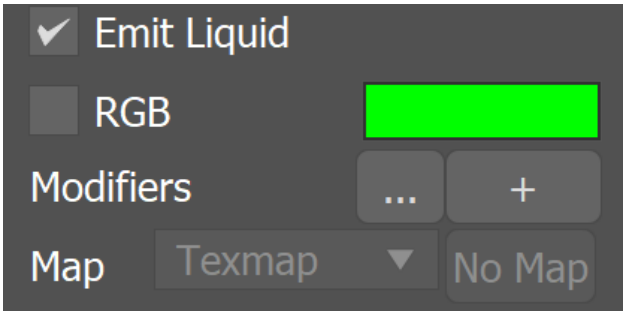
Map | *uvwmap, useuvwmap* – Allows you to vary the RGB over the surface or the volume of the emitters. If this is not used, the Source will emit equal RGB over the entire surface or volume of the emitters.

> **None** – The RGB channel will not vary.
> **Texmap** – Allows you to specify a texture map to color the fluid emitted by the **Source**. If this is used, the color swatch is ignored and the RGB comes entirely from the texture. For more information on texture mapping in Phoenix, please check the Texture mapping, moving textures with fire/smoke/liquid, and TexUVW page.
> **Vertex Color** – The RGB channel of the emitted fluid is determined by the emitter node's vertex colors. If this is used, the color swatch is ignored and the RGB comes entirely from the vertex color. The texture map slot is also ignored.

To render these RGB colors for smoke, set the Smoke Color **Based On** parameter to **RGB**.

For rendering of meshed liquids, set a Grid Texture as the **Diffuse** map for a Standard or V-Ray Material, and set the Grid texture's **Channel** to **RGB**.

For more information, see the RGB Map Vertex Color example below.



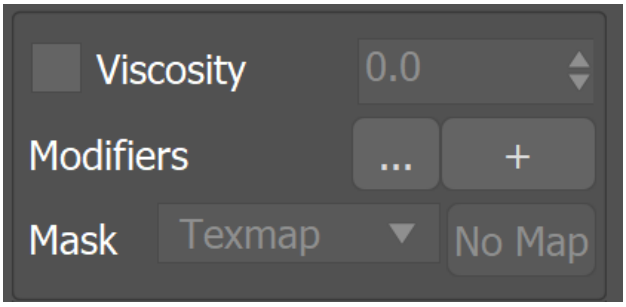
Viscosity

Viscosity | *viscosity, usevisc* – Specifies the viscosity of emitted liquid. If the viscosity channel is not enabled in the Output rollout of the Simulator, this parameter will be ignored.

Modifiers | *dmodvisc* – Discharge Modifiers can be attached here in order to affect the **Viscosity** parameter. The modifier ramp works as a multiplier to the Viscosity value.

Mask | *viscmap, useviscmap* – Allows you to vary the viscosity over the surface or the volume of the emitters. If this is not used, the Source will emit equal viscosity over the entire surface or volume of the emitters.

> **None** – The Viscosity channel will not vary.
> **Texmap** – Allows you to specify a texture map for the Viscosity channel. For more information on texture mapping in Phoenix, please check the Texture mapping, moving textures with fire/smoke/liquid, and TexUVW page.
> **Vertex Color** – The Viscosity channel of the released fluid is determined by the emitter node's vertex colors.



Particles

Foam Particles | *usefoamprt* – Allows the source to emit Foam particles into the Simulator. Note that Foam simulation must be enabled from the Simulator so this type of particles can be emitted into it. The particle birth rate is in thousands of particles per second.

Splash Particles | *usesplashprt* – Allows the source to emit Splash particles into the Simulator. Note that Splash simulation must be enabled from the Simulator so this type of particles can be emitted into it. The particle birth rate is in thousands of particles per second.

Mist Particles | *usemistprt* – Allows the source to emit Splash particles into the Simulator. Note that Splash | Mist simulation must be enabled from the Simulator so this type of particles can be emitted into it. The particle birth rate is in thousands of particles per second.

Modifiers | *dmodprt* – A Discharge Modifier can be attached here in order to affect the **Particles'** parameter.

Mask | *prtmap, useprtmap* – Allows you to vary the amount of particles over the surface or the volume of the emitters. If this is not used, the Source will emit an equal amount of particles over the entire surface or volume of the emitters.

Emission

Direct Velocity | *useDirectedVelocity* – Creates velocities in a certain direction or based on a Texmap.

Modifiers | dmoddirvel – Discharge Modifiers can be attached here in order to affect the Direct Velocity parameter.

Map | usedirvelmap – Allows you to vary the Direct Velocity over the surface or the volume of the emitters. If this is not used, the Source will emit equal Direct Velocity over the entire surface or volume of the emitters.

> **None** – The Direct Velocity channel will not vary.
> **Texmap** – Allows you to specify a texture map to direct the velocity of the fluid emitted by the Source. When using Texmap, the Direct Velocity is generated from the selected texture and is not multiplied by the value added in the Directed Velocity X/Y/Z slots.
> **Vertex Color** – The Direct Velocity of the emitted fluid is determined by the emitter node's vertex colors. When using Vertex Color, the Direct Velocity is generated from the created vertex color and is not multiplied by the value added in the Directed Velocity X/Y/Z slots.

When using a Texture map to direct the Velocity of the fluid emitted from the Source:

The **Red** color affects the Direct Velocity on the **+/- X** axis.

The **Green** color affects the Direct Velocity on the **+/- Y** axis.

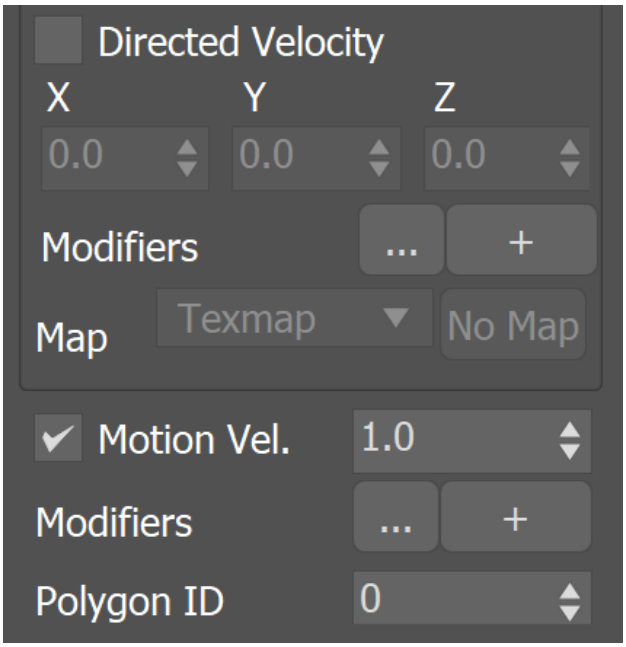
The **Blue** color affects the Direct Velocity on the **+/- Z** axis.

Motion Vel. | *usevel, velmult* – When enabled, moving emitters will affect the velocity of the fluid and make it follow the emitter. This effect is controlled with the specified multiplier. If the emitter is not moving, this option has no effect.

- Particles with attached geometry (such as PFlow particles using a **Shape Instance** operator) will act like standard geometry obstacles and will push the fluid by default, as long as their **Solid** Phoenix property is enabled.
- Particles without geometry (even PFlow particles with a plain **Shape** operator) will need to be attached to a Source in order to interact with the fluid in any way.
- Particles without geometry can emit fluid using their shape and size using the Source's **Particle Shape** and **Custom Prt Size** options.
- Particles without geometry will not affect the fluid's motion by default, so you need to enable **Motion Vel.** explicitly if you need this effect.
- If **Motion Vel.** is enabled, the fluid emitted by a particle may refuse to leave the particle shape area and continue to move together with the particle, because it will have the same velocity.

Modifiers | *dmodvel* – Discharge Modifiers can be attached here in order to affect the **Motion Vel.** parameter.

Polygon ID | *poly_id* – Only the polygons with the specified ID of the emitter geometry will emit the fluid.



Texture UVW

The main purpose of Texture UVW is to provide dynamic UVW coordinates for texture mapping that follow the simulation. If such simulated texture coordinates are not present for mapping, textures assigned to your simulation will appear static, with the simulated content moving through the image. This undesired behavior is often referred to as 'texture swimming'. In Phoenix such textures can be used for mapping the fire or smoke color and opacity of volumetrics, as well as the color and opacity of meshes. Texture can be also used for displacing volumetrics and meshes.

UVW coordinates are generated by simulating an additional **Texture UVW Grid Channel** which has to be enabled under the **Output rollout** for the settings below to have any effect.

For additional information on the Texture UVW feature, please check the Text ure mapping, moving textures with fire/smoke/liquid, and TexUVW page.

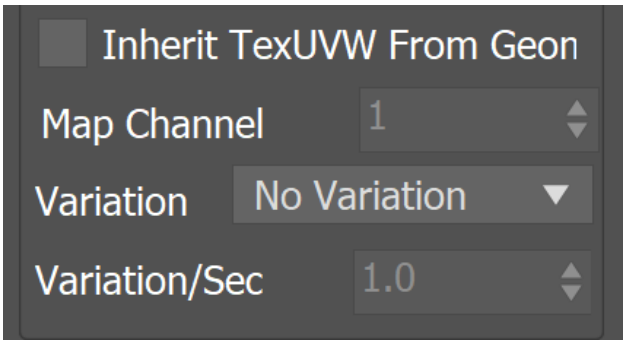
Inherit TexUVW From Geom | *texuvw_geom* – Sets the **UVW Grid Channel** value for each cell where fluid is emitted to the UV value of the emission geometry in that cell. As a consequence, for example, modulating the Smoke Color with a texture on the very first frame will produce a render that looks very close to the original geometry, if the same texture was applied to it. When this option is disabled, the TexUVW values will be based on the position of the emission object inside the Simulator. Please check the Texture UVW example below.

Map Channel | *texuvw_geom_ch* – Specifies the Map Channel index to sample. This is useful when your geometry has multiple UV sets (called Map Channels in 3ds Max) with different layouts.

Variation | *texuvw_var_mode* – Variation is used to offset the UVW coordinates upon emission to avoid visible tiling once a texture is applied to the resulting simulation. Similarly to a printer, if the UVW channel is not varied, it would be like printing out the same sentence over and over again on each new line. When varied, the printer will change the line being printed. The following methods are available:

- **No Variation** - The emitted TexUVW will not change over time.
- **Along U/V/W** - The emitted TexUVW value will change over time in the U/V/W direction and by the amount set in Variation/Sec.
- **Along Grid/Object/World Normals** - The emitted TexUVW value will change over time in the direction of the emitting geometry's normals in Grid/Object/World space and by the amount set in Variation/Sec. Grid space is the system of the Simulator, it always has the Z axis pointing up. Object space is the local transform of the emitter.

Variation/Sec | *texuvw_var_speed* – Controls the variation speed - the default value of 1 will cause textures assigned for rendering to repeat exactly once for every 30 frames (1 second) of the simulation.



Example: Inherit TexUVW with Variation

The following video provides examples of **Inherited Texture UVW** coordinates **repeating along the V-axis** over **0**, **1** and **2** seconds. When **Variation** is set to **0**, the V coordinate remains **static**. If **Variation** is set to **1**, it takes **one second** for a **full repetition/tile** along the V-axis.

Emission from Particles

Time Base | *timebase* – This parameter is used when emitting from particle systems. It allows you to animate the parameters using the age of the particle instead of the timeline frame time.

> **Absolute** – Parameters of the source will be animated based on the timeline frame time.
> **Particle Age** – Parameters of the source will be animated based on the age of the particle. This way, values at timeline frame 0 will apply to each particle at the moment of its birth, and e.g. values at frame 10 will apply to the particle 10 frames after it was born. This allows particles born at different moments to perform identical animations offset in time. This can be useful e.g. if you want all particles to emit strongly after they are born and reduce their emission after a while, but in case the particles are born through a long period of time, the **Ab solute** mode will change the discharge of all particles together, while Particle Age will allow each particle to have its own copy of the animation.

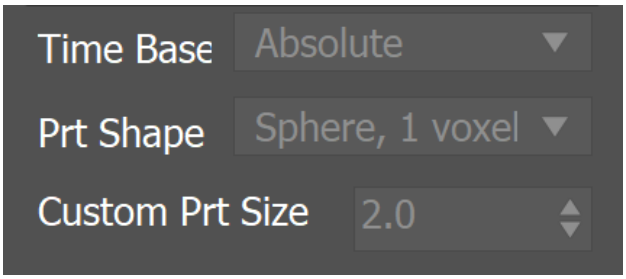
Time Scale different than 1 will affect the **Particle Age** in the Liquid Source. In order to get predictable results you will have to adjust the keyframes using this formula: **Time Scale * Time in frames.**

Prt Shape | *prt_shape* – This parameter is used when emitting from particle systems.

> **Sphere, 1 voxel** – Each particle will be the size of one grid cell. Particle sizes and shapes will be ignored.
> **Sphere, use size** – The particle sizes will be used, but the shape will always be spherical.
> **Sphere, custom size** – The particle shape will be spherical and the size will be taken from the **Custom Prt Size** field.
> **Use particle shape** – The particle shapes will be used as they are provided. This might slow the simulation down if there are a large number of particles.

When a source emits from non-solid particles in any of the Sphere modes, the simulator traces each moving sphere and emits continuously throughout its trajectory, no matter how fast the particle is moving and how many steps the simulation uses. In a contrary way, in **Use particle shape** mode, the particles are getting evaluated the same way as regular mesh geometries, so in motion they are sampled only at the simulation steps without filling the trajectory in between the particle positions in time. In such case, if a particle moves very quickly and the simulator has low **Steps per Frame**, the trajectory of the particle would get interrupted and you should increase the simulation steps in order to keep it continuous.

Custom Prt Size | *prtsphsz* – Specify a custom size for the particles using this option. The size is in scene units.



Example: RGB Map Vertex Color

Smoke Simulation
Map = Vertex Color

Liquid Simulation
Map = Vertex Color