# FLUIDOS II

## for DAZ Studio

## LITE EDITION

**v 2.3**

**ALVIN BEMAR**

TRADEMARK & COPYRIGHT NOTICE

**Daz Studio 4**™ software and logo are a copyright of ©2001-2025 DAZ 3D, Inc. **Daz Studio 4**® is a registered trademark of DAZ 3D, Inc.

**GridFluidSim3d** and **FLIPViscosity3D** are copyright of © 2016 Ryan L. Guy

**OpenVDB 9.1.0** is copyright of © 2012-2022 DreamWorks Animation LLC

**Fluidos II for Daz Studio** plugin is copyright of ©2019-2025 Alvin Bémar

**Fluidos II for Daz Studio Lite edition** plugin is copyright of ©2025 Alvin Bémar

**Fluidos for Daz Studio** plugin is copyright of ©2018-2025 Alvin Bémar

## Contents


INTRODUCTION....1
BASIC OPERATION....3
RESOLUTION SCRIPTS....8
COMPONENTS OF THE SIMULATION....9
FLUIDOS DOMAIN....9
ENGINE:....9
FLUID TYPE:....9
OBSTACLES:....10
FLUID OBJECTS....10
SOURCES AND SINKS....10
FLUIDOS MESHER....11
VISCOSITY....11
UNITS....12
FORCES....12
DIFFUSE PARTICLES....13
FILES....13
MENUS AND PROPERTIES....14
Fluidos Domain....15
Fluidos Source / Sink....30
Fluidos Forces....36
Objects Properties:....41
Running a simulation....46
Visualizing the results of simulation....48
References....57
Acknowledgements....58
Licences:....59
APPENDIX A: How to create a Properties preset for the Fluidos Domain....62
APPENDIX B: How to use the preset examples included in Fluidos....65
APPENDIX C: Example of use of Any Node Geometric Type Source....68
APPENDIX D: Effects of the settings....72
Main settings....72
Cell size....72
Subdivision Level:....73
Vorticity confinement....74
Remove isolated particles....76
Gravity forces....77
Viscosity subgroup....78
Viscosity....78
Diffuse particles subgroup....79
Max number of diffuse particles....79
Wavecrest particle emission rate and Turbulent particle emission rate....80
Max lifetime of a particle....81
Scaling exponent....82
Advanced setting subgroup....83
Meshing type....83
Jitter factor:....85

Marker Particle Scale....87
Filters....88
Iterations....93
Surface adaptivity....94
PIC/FLIP ratio....96
Smoke specific properties....97
Smoke temperature....97
Beta buoyancy factor....98
Ambient temperature....99
Forces....100
Point Force....101
Linear Force....102
Torque Force....103
Directional Force....104
Flow Force....105
Force field....106
Meshers....107
Mesh type....107
Meshers for smoke....108
Meshers for fire....108
Erosion of flames....109
Shaders....111
New in Version 2.3....113
New in Version 2.2....113

# INTRODUCTION

FLUIDOS II for Daz Studio is a plugin that runs a modified version of an external fluids simulator, the **GridFluidSim3d** (Ryan L. Guy http://rlguy.com/gridfluidsim/) with some algorithms adapted from **FLIPViscosity3D** https://github.com/rlguy/FLIPViscosity3D and C. Batty VariationalViscosity3D (https://github.com/christopherbatty/VariationalViscosity3D): These programs are implementations of PIC/FLIP liquid fluid simulation written in C++11 based on the works of Robert Bridson and Christopher Batty. The fluid simulation engine outputs the surface of the fluid as a sequence of triangle meshes stored in the Stanford .PLY file format. The engine incorporates also OpenVDB tools to achieve high quality results (https://www.openvdb.org) .

Key Features:

- Use of high performance OpenVDB tools for meshing, filtering, level sets and pressure projection.
- GPU accelerated steps using OpenCL.
- Multithreaded.
- Liquid, smoke and volumetric fire simulation. (Fire is not available in LITE edition)
- Accurate viscosity for buckling, coiling, and rotating liquids.
- Diffuse particle simulation.
- Stabilization steps to generating only one frame.
- Autosave and load state of a simulation.

For more technical explanations about fluid simulation, see the references.

**How to register the plugin (needed for using it):**

1. On Daz website, go to *My Account.*
2. Click on *Serial Numbers* and copy the Serial number of the “FLUIDOS II for Daz Studio - LITE edition” product.
3. Start Daz Studio.
4. Go to *Help – About installed plugins.*
5. Scroll down to the “FLUIDOS II for Daz Studio - LITE edition” product, paste the Serial number and click OK.
6. Restart Daz Studio and you should be up and running.

**Installation and compatibility issues with *Fluidos for Daz Studio* v1.3**

- Fluidos II can read all the scenes, and all the .ply and .state files created by Fluidos v1.3.
- It is not necessary to uninstall the **contents** of Fluidos v1.3 as they are compatible with Fluidos II.
- It is necessary that OpenCL for your GPU vendor is installed in your computer.

## BASIC OPERATION

A fluid simulation takes place in a spatial region called Fluidos domain. The objects with a geometry inside the domain will be considered as fluids or solid obstacles. There are other objects the simulator can manage: forces, fluid sources and fluid sinks. The forces affect the fluids dynamical behavior over time. The fluid sources are inputs of a fluid inside the domain while sinks are outputs. The simulator saves the results in files, at least one for each frame.

To visualize the simulated fluids, there is a Mesher that shows fluid surfaces or diffuse particles.

**Color of properties:**
The properties in the user interface now have color. Blueish for basic properties, greenish for intermediate and grayish for advanced.

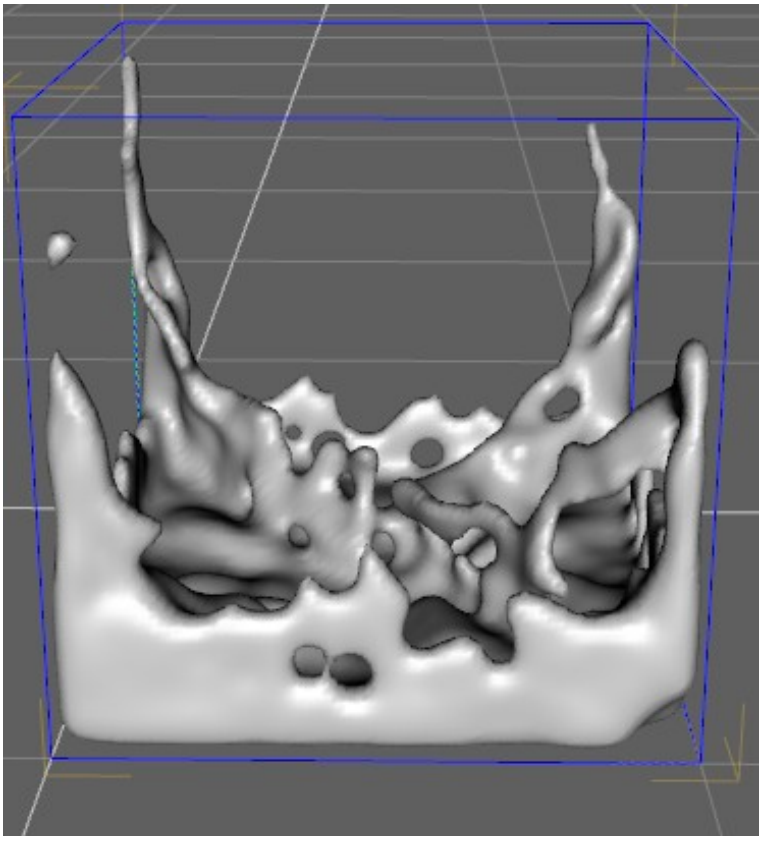
**First example (liquid):**

1. Prepare a folder where the simulator can save the fluid simulation files, anywhere you wish. A fast drive (as SSD) may enhance performance when rendering.
2. Create a new scene.
3. Insert a Fluidos Domain (menu Create – New Fluidos Domain and Accept).
4. Insert a sphere primitive (0.6 m diameter is a good option).
5. Parent the sphere to the Fluidos Domain.
6. Move the sphere to the center of the Fluidos Domain. Scale the sphere if you wish but avoid to fill all the Domain. Don't scale the domain for now.
7. In the Parameters tab of the sphere, as it is parented to the Fluidos domain, there are new parameters, in the FLUIDOS group. Select *Fluid mass* in the *Object type* option.
8. Select the Fluidos Domain.
9. In Fluidos Domain Properties (Parameters tab, subgroup FLUIDOS/Main Settings), change the number of frames to 30 to obtain one second in the simulation. Let the cell size in 2.50 (cm).
10. In the same Fluidos Domain Properties, subgroup FLUIDOS/Main Setting, click on Baked files folder and select *Browse* to locate the folder mentioned in the step 1.
11. Call the simulator (menu Edit – Run Fluid Simulation).
12. Click Accept in the dialog.
13. Wait for the simulator to finish (a progress bar is displayed).
14. Set the inserted sphere to invisible.
15. Now insert a Fluidos Mesher (menu Create – New Fluidos Mesher and Accept). This must be in the same position as the Fluidos domain.
16. Choose the same Baked files folder (step 10) for the Mesher in Parameters tab, subgroup FLUIDOS Mesher/General, as in the step 10.
17. Set a keyframe at frame 0 in the Timeline. Then, put a second one in frame 30 by setting *Completion* to 100 %.
18. Set *ON Enabled* in properties (subgroup FLUIDOS Mesher/General).

19. You will see a preview. This is intended to get a fast scroll around the Timeline.
20. The final simulation can be seen by setting *OFF* the Preview parameter of the Mesher (subgroup FLUIDOS Mesher/General).

You can stop the simulation by pressing the escape key or click in Cancel in the progress bar. To resume, call again the simulator and set *ON Continue saved state* before clicking the *Accept* button in the dialog box.

The final results at frame 30 must be similar to the following image:



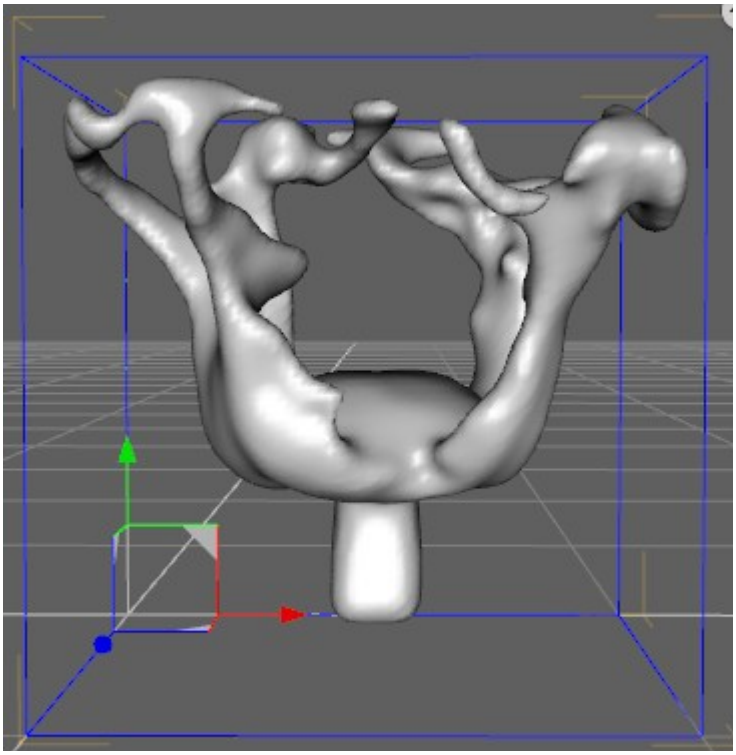
*Note*:
If the user change the default values in *Preferred device to CPU* or in *Enable OpenCL* property, the plugin will remember these two settings for the next time you create a New Fluid Domain[1].

1 The defaults are:
"Enable OpenCL" = On
"Preferred device" = GPU

**Second example (smoke):**

1. Create a new scene.
2. Insert a Fluidos Domain (menu Create – New Fluidos Domain and Accept).
3. Insert a Fluidos Source/Sink.
4. Parent the Source to the Fluidos Domain; this will be the Source. Set its *velocity (x)* property to 0.0 and the *velocity (y)* to 100.0. Set the *Temperature* to 150 ºC and the *Smoke concentration* to 2.0 %.
5. Move the Source by setting *X Translate* = 50.0, *Y Translate* = 15.0 and *Z Translate* = 50.0.
6. Select the Fluidos Domain.
7. In Fluidos Domain Properties, change the number of frames to 30 to obtain one second in the simulation. Let the cell size in 2.50 (cm). In *Fluid type* choose *Smoke*. In the *Fluidos/Advanced settings* subgroup, set the *Jitter factor* to 1.0.
8. In the same Fluidos Domain Properties, click on Baked files folder and select *Browse* to locate the folder where the plugin will save the simulation files.
9. Call the simulator (menu Edit – Run Fluid Simulation).
10. Click Accept in the dialog.
11. Wait for the simulator to finish (a progress bar is displayed).
12. Now insert a Fluidos Mesher (menu Create – New Fluidos Mesher and Accept). This must be in the same position as the Fluidos domain.
13. Set the same Baked files folder (step 9) for the Mesher.
14. Set a keyframe at frame 0 in the Timeline. Then, put a second one in frame 30 by setting *Completion* to 100 %.
15. Set *ON* the *Enabled* property
16. You will see a preview. This is intended to get a fast scroll around the Timeline.
17. The final simulation can be seen by setting *OFF* the Preview property.
18. Apply a *Smoke* shader to this mesher. Alternatively, you can change the *Mesh type* to *Fluid particles* and apply a *Smoke particles* shader to the Mesher.

The final results at frame 30 must be similar to the following image (Daz default shader in Mesher):



**Using the stabilization steps to get only a frame.**

In any of the previous examples, instead of setting the *number of frames* property to 30, set to 0 and set the *Stabilization steps* to 30. Insert the Mesher, set the Baked files for it and choose the corresponding *Mesher type*. Let aside to set the keyframes for the Mesher, but set ON the *Enable* property and set OFF the *Preview* property (if it is available).

As the fluid simulation is running the Mesher will show the evolution of the simulation. When this is finished, the frame 0 will show the final result.

**Volumetric fire mesher example.**

Fluidos II LITE edition, unlike the Complete edition, does not support *Volumetric fire* simulations. However LITE edition can read baked simulations created with the Complete edition. The included scene "*Fluidos II candle fire mesher exampl*e.*duf*" contains a baked *Volumetric fire* simulation. Enable the *Fluidos mesher* in the scene to see the baked simulation.

## RESOLUTION SCRIPTS

There are four resolution scripts included with Fluidos packages. These scripts are only for the convenience of the user; they are not required for the fluids simulation.

The scripts should be applied to the **Fluidos Domain** *after setting its size*, although could be reapplied in any moment before the running of the simulation.

The resolution scripts function is to select a suitable *cell size* and the *subdivision level* for the size of the Domain. However, after applying a script, the user can modify those properties to tune the simulation.

***L*** Low resolution: fast, but low quality. The simulator could need up to **18 MB** of RAM.

***M*** Medium resolution. The simulator could need up to **146 MB** of RAM.

***H*** High resolution. Slow, but high quality. The simulator could need up to **1.1 GB** of RAM.

***VH*** Very high resolution. Could be very slow. The simulator could need up to **9.2 GB** of RAM.

The actual RAM needed depends on the quantity of fluid inside the Domain and the type of fluid, smoke and fire *will need more memory* than liquid.

**Take into account the resources of your computer before running the simulation!**

## COMPONENTS OF THE SIMULATION

FLUIDOS DOMAIN:

The fluid simulator performs its computations on a 3D grid, and because of this, the simulation domain is shaped like a rectangular prism. You can modify the width, the depth and the height of the domain using the sizeX, sizeY and sizeZ properties. Or you can use the scale tool.

All the objects to be part of simulation, except by forces, must be inside the 3D space of the Fluidos Domain. Even more, all these objects, including forces, must be parented to the Fluidos Domain. There is no limit of level of depth in hierarchy inside the domain.

Multiple *Fluidos Domains* can be in a scene, but only one is calculated at a time.

ENGINE:

*Fluidos II* plugin has two engine modes. The default is the *Fluidos II* engine, but there is *Fluidos I* engine. The last will reproduce the behavior and properties of the old Fluidos for Daz Studio v 1.3, although at a faster speed than the old plugin.

FLUID TYPE*:*

There are two types of fluids to choose: Liquid, Smoke (Fluidos II LITE edition does not support Volumetric fire). Unlike smoke, liquid simulations do not make use of temperature parameters. Smoke could be slower and more memory-hungry as they does the complete calculations on the whole Domain, whereas liquids only focus in cell with fluid inside, and their boundaries.

Smoke has only one solver type, a simplified Boussinesq method (The complete edition supports three alternative solvers).

*Fluid Buoyancy:*

The buoyancy allows a smoke fluid to float in the air, unlike a liquid. For the buoyancy to operate, the smoke fluid must be less dense than the surrounding air. There are two forms to achieve this result in Fluidos II LITE edition,

- The temperature of the fluid should be higher enough than the ambient temperature (a high concentration of fluid will need a higher difference in temperature than a lower concentration).
- Increasing the *Beta bouyancy factor.*

OBSTACLES:

Any 3D object with a mesh can be used as a solid obstacle in the simulator. The meshes *must be closed surfaces*, to function properly.

A thin wall could leak fluids. This is fixed decreasing the cell size, increasing the *Thicken* parameter or adding a Geometry shell.

It is not necessary for the whole solid volume to be inside the Fluid Domain, but the outside regions will be ignored by the simulator.

The solids could have a surface force in them. This can be used to stick the fluid to the solid or to repel it.

The engine *has no functionality to simulate the affect the motion of moving obstacles*, but you can animate the solids manually. The default behavior of the simulator considers the obstacle as static, so if you keyframed its motion, the collision with fluid cells causes the obstacle substitute them and therefore. If there are fluids sources in scene, this can be mitigate by using surface forces to simulate a push and by increasing the FPS of the simulator. A slightly slower option is to *Enable the moving obstacles;* this functionality simulates the effect of the obstacle *on* the fluid.

Due to the intrinsic characteristics of the FLIP algorithms, the fluid volume could increase or decrease over time. The moving obstacles tend to enhance the loss, but enabling the moving obstacles lessen this behavior, although not entirely.

FLUID OBJECTS:

As in the case of solids, any 3D with a closed surface can be a fluid object. It will react as a fluid mass in the simulator.

It is not necessary for the whole fluid volume to be inside the Fluid Domain, but the outside regions will be ignored by the simulator.

SOURCES AND SINKS:

The simulator can manage sources of fluids to get a stream. There is a special FLUIDOS object for them: *Fluidos Source / Sink.* This source can have a rectangular prism shape or a sizable spherical one (this is not visible as a sphere in the screen). They can be resized and moved around the domain. The velocity of the fluid is animatable too.

Since version 1.3, *Any node* can be used as a source or sink (see *Fluidos Source/Sink*, p. 17 and Apendix C, p. 42 for details).

A similar object is the sink, but it absorbs the fluid. It doesn't need a velocity.

It is not necessary for the whole source or sink to be inside the Fluid Domain, but the outside regions will be ignored by the simulator.

FLUIDOS MESHER:

The simulator saves the baked files in a folder of your choice. But to render the simulation you need a Fluidos Mesher. This object reads the geometry generated by the fluid simulator. Do not scale the Mesher unless you want to scale or deform the simulation results.

If you want to run the simulator to extend or modify a previous simulation, you may want to disable the *Mesher* until the simulation is finished, because this can slow down the process.

Meshers can be in a scene, and all of them can be enabled.

There are four types of mesh: *Fluid*, *Fluid particles*, *Fire* and *Diffuse particles*. Although Fluidos II LITE edition does not support *Volumetric fire* simulations, it can read baked fire simulations created with the Complete edition.

The *Fluid mesh* is a "solid", closed mesh. The *Fire mesh* is also a "solid" mesh, but it has associated temperatures data used by a special shader. The *Fluid particles mesh* and the *Diffuse particles mesh* are triangles representing each particle. *Fluid particles* mesh, as the *Fire mesh*, has associated temperature data used by a special shader.

VISCOSITY:

The simulator can manage viscous liquids too, as oils, honey or asphalt.

Sometimes, with low viscosity values, bits and pieces of fluid can be left hanging in the air. To solve this problem, increase the *Marker Particle Scale* value.

Spatially variable viscosity can be simulated with the aid of *Viscosity controls*, that is, any object with a closed surface. The viscosity values at a point in space are the sum of all the controls that cover this point plus the general viscosity value (at Viscosity in Fluidos domain's properties).

Viscosity simulation are slower than inviscid ones; be patient.

UNITS:

The units of length used in the simulator are centimeters unless is indicated otherwise.

FORCES:

The forces in *FLUIDOS* are nodes that can affect the fluid dynamics by attracting or repelling the fluid masses in space.

The *FLUIDOS* plugin is able to manage spatially constant forces; that is, forces that act with the same intensity in all space, as the usual gravity force in earth (these forces are controlled with the Fluid Domain Properties Body Forces options).

Other type of forces are the spatially variable forces. These act with variable intensity in space. FLUIDOS can manage five types of spatially variable forces: Point, Torque, Linear, Directional and Field force.

*Point Force*: This force acts from a point in space. The fluid is attracted to this point (strength positive values) or repelled from it (strength negative values). The intensity of the force depends on the distance of the fluid mass from the central point. The decay rate is quadratic, i.e., if the distance doubles, the intensity decreases to a quarter. This force is the equivalent to Point light.

*Linear Force*: This force acts from a straight line in space. The fluid is attracted to this line (strength positive values) or repelled from it (strength negative values). The intensity decay rate is quadratic.

*Torque Force*: this is a rotational force. As in the case of point force, the position is relevant, but the rotation too. This force acts in all space exerting a tangential push over the fluid, so it rotates around the Torque force axis.

*Directional force*: This force extends its action all over the space. The direction is relevant, but position do not. This force is the equivalent to Distant light.

*Field force:* this one extends its influence over a confined space and its surroundings. The space is delimited by the user with a geometric node. The intensity decay is linear.

The last type of force are the surface forces. These act only *on* the surface of the fluid, not *in* its body. FLUIDOS has two surface forces:

*Flow Force:* this force acts randomly on points of the fluid surface that are not in contact with a solid obstacle. It can simulate wind and for that it has a main thrust direction.

*Surface force in Obstacles*: this force acts in the contact surface between a specific solid obstacle and the fluid. It's constant in all the contact surface, and its intensity is the same for all solids. Positive Intensity is for attraction and negative Intensity is for repulsion.

The forces units are in acceleration units (as if the mass is unitary), m/s2.

All the forces are animatable.

DIFFUSE PARTICLES:

The diffuse particles, an additional option of a simulation, are the equivalent of Whitewater from another software. These represents foam, spray and bubbles. In fact, the Mesher uses triangles to represent them. But these triangles can be considered billboards.

The diffuse particles add realism to a fluid simulation.

FILES:

The plugin creates four subfolders inside the folder selected by the user: *baked files*, *logs*, *savestates* and *temp*. In *baked files* are saved the simulation files in a .ply format; *logs* contains the logs of each simulation; inside *savestates* is the file that saved the last state of the simulation (it is used to resume and stop the simulation); *temp* is a reserved folder.

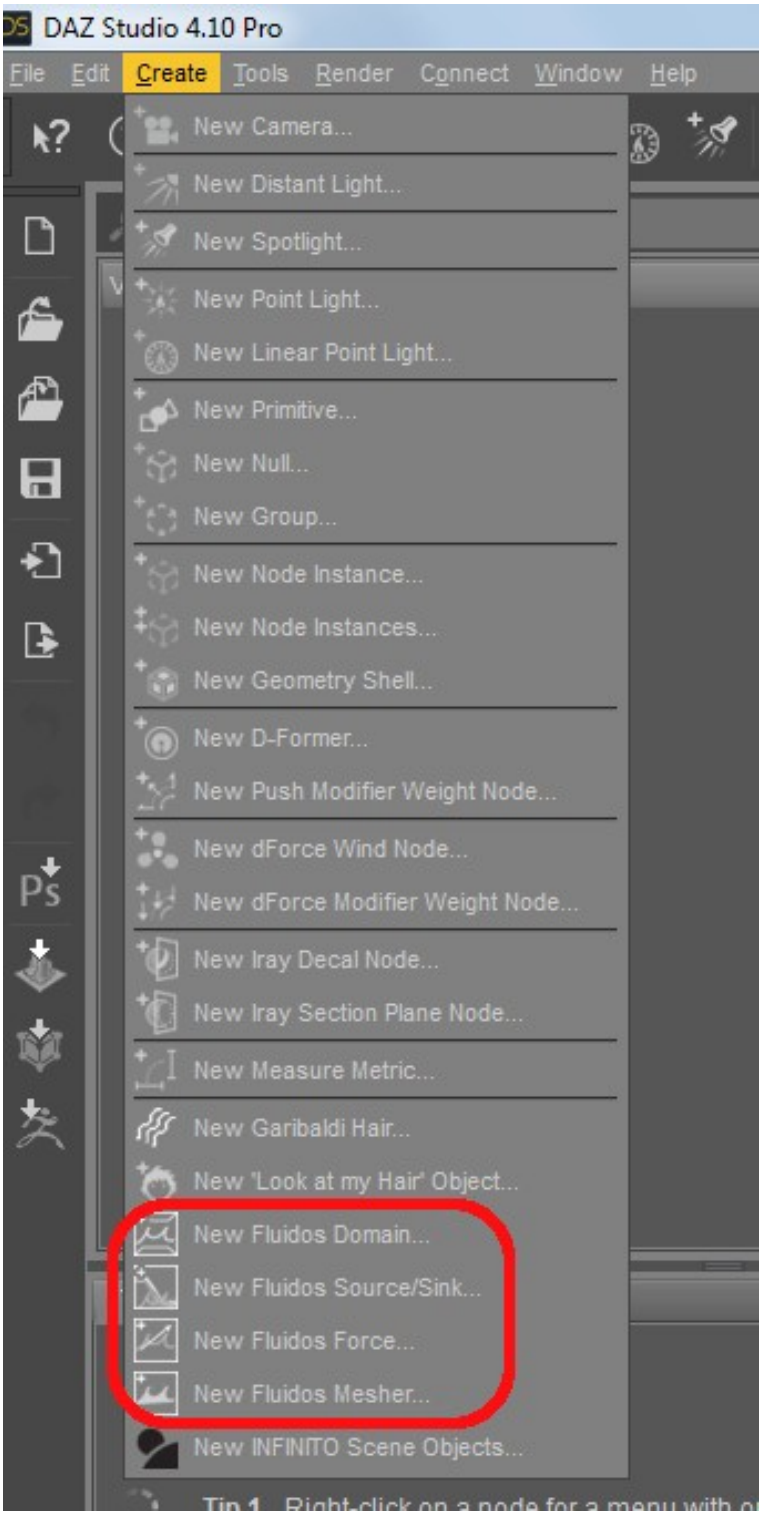
The geometry is not saved (except the present in screen) with the .duf, but in the folder output.

As the simulations overwrites their files, it is advisable to reserve a folder output for any relevant work.

## MENUS AND PROPERTIES

In this manual, the new properties and features available in Fluidos II in relation with the old *Fluidos for Daz Studio* plugin are preceded by a red asterisk (*).

- To set the FLUIDOS nodes in scene, select the menu Create.

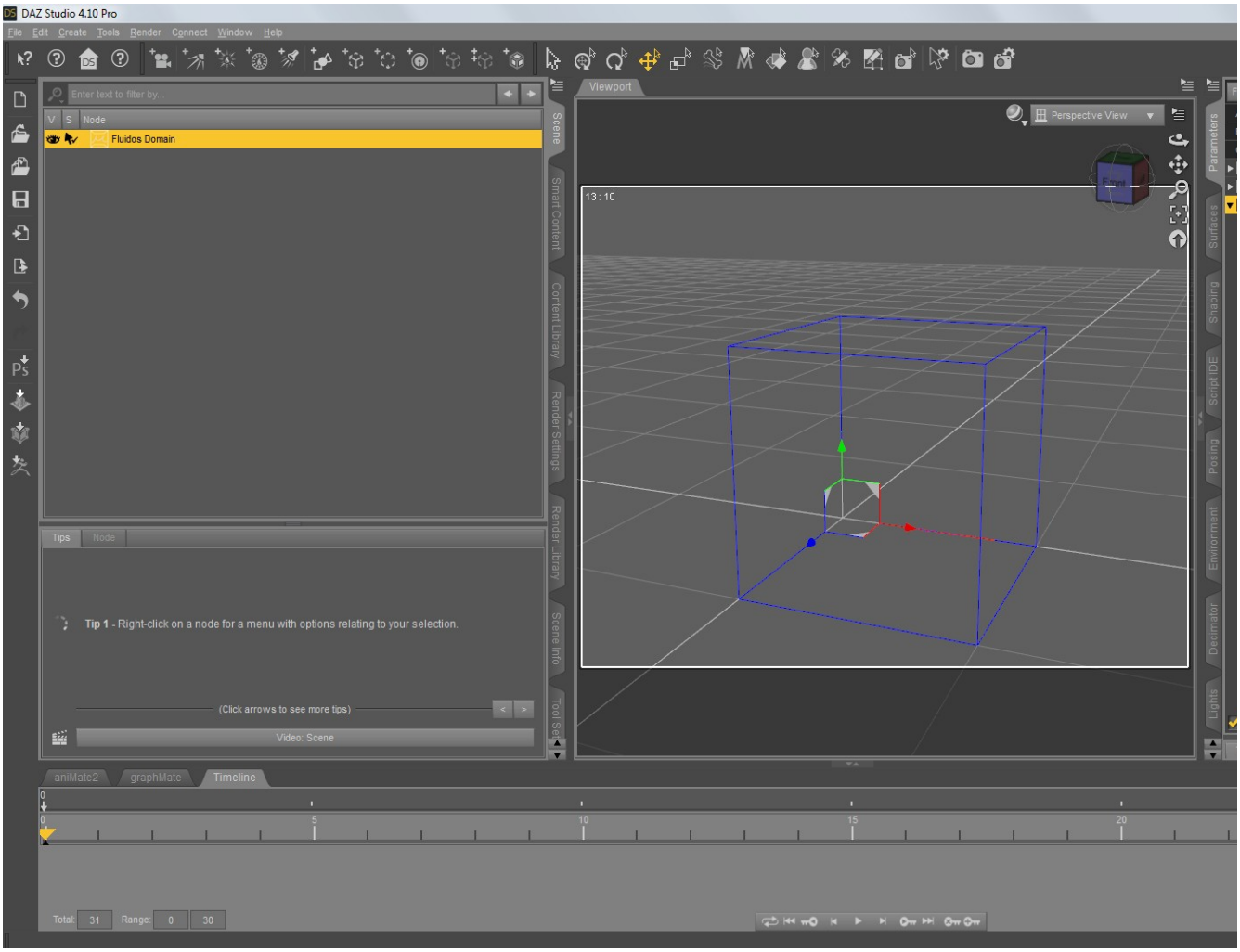


The user can now choose one of four nodes:

- Fluidos Domain
- Fluidos Source/Sink
- Fluidos Force
- Fluidos Mesher

**Fluidos Domain**

The Fluidos Domain is represented in screen as an empty cube. This cube can be resized as a rectangular prism.

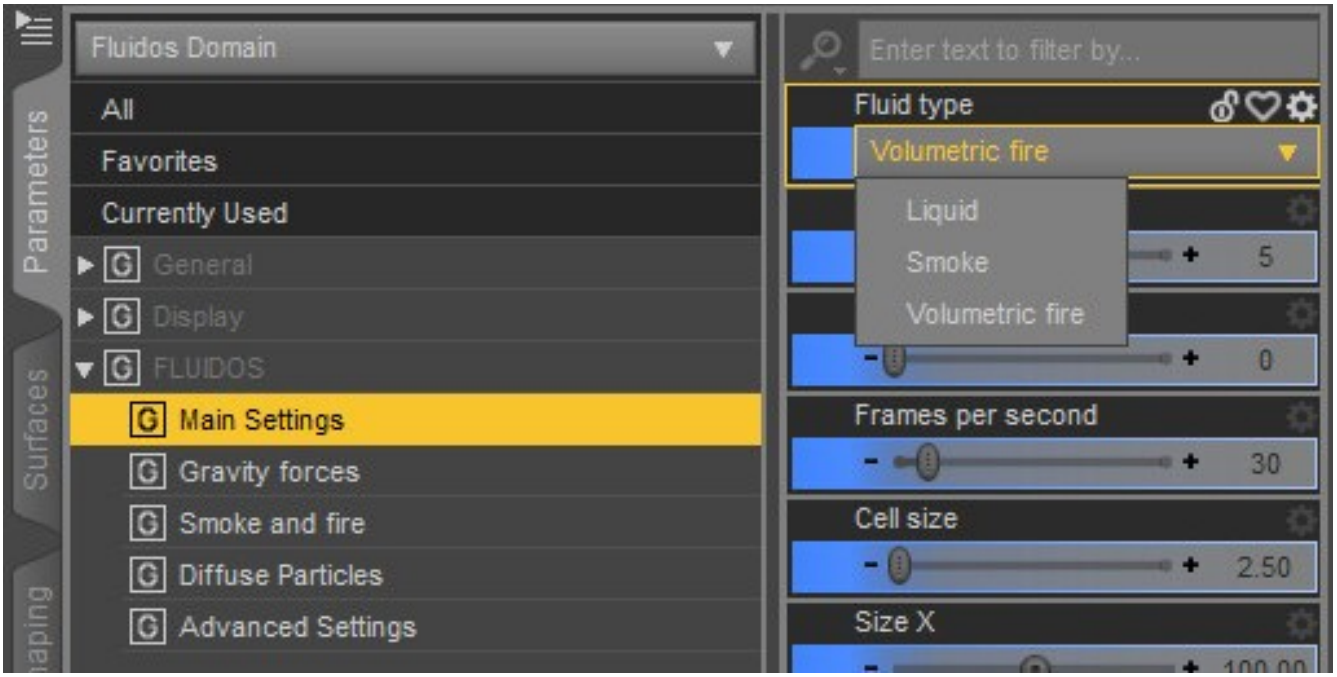



In Parameters, there are six groups for the Fluidos Domain:

- Main Settings
- Gravity forces
- Smoke

- Diffuse Particles
- Viscosity
- Advanced Settings

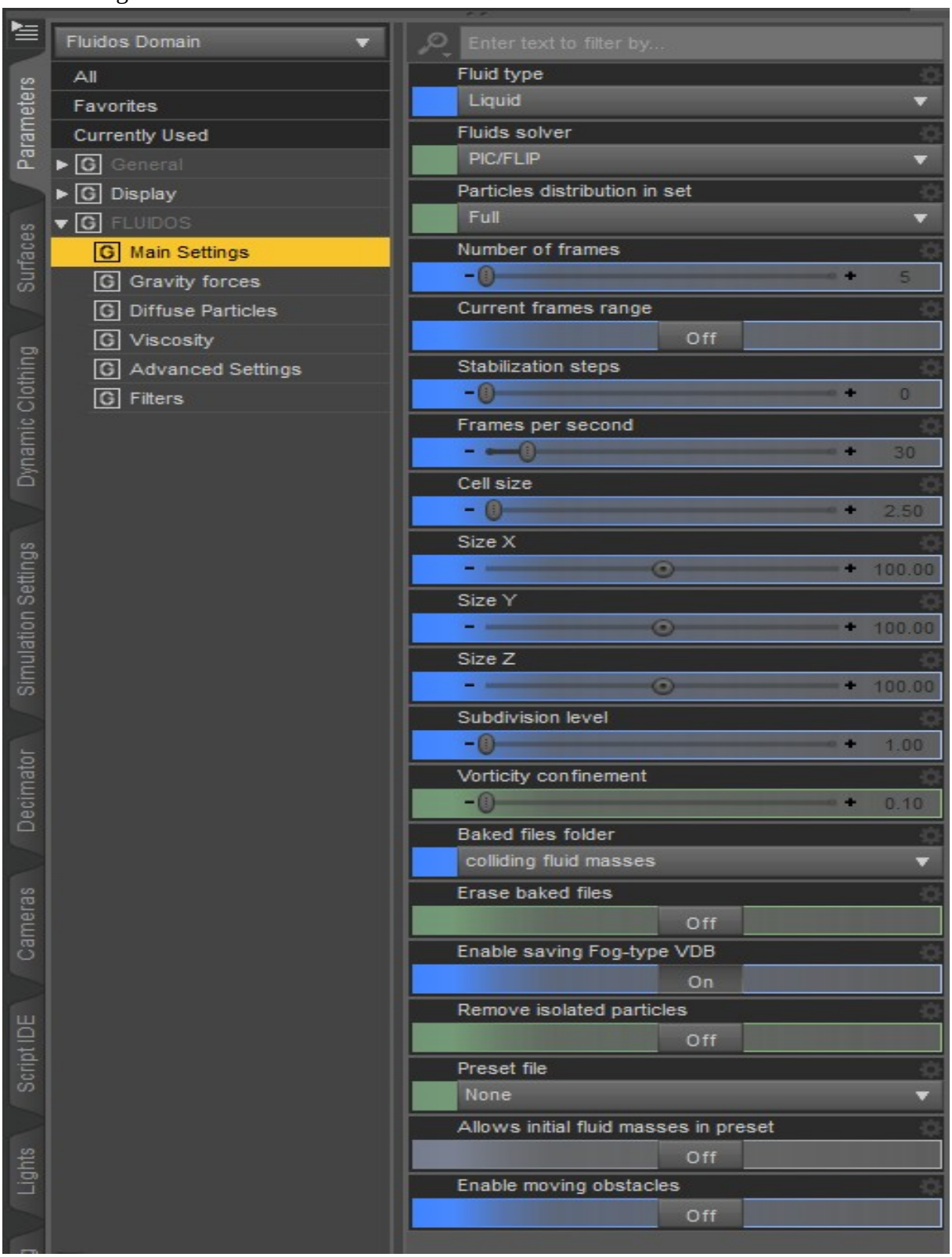
*Smoke and fire* group is visible only if the current *fluid type* is *Smoke* (*Volumetric fire* is supported only in the Fluidos II complete version). Viscosity *group* is visible only if the current *fluid type* is *Liquid*.




*Fluid type*: it is the type of fluid to be simulated: *Liquid*, *Smoke* and *Volumetric fire*. The default is *Liquid.*

Note: *Volumetric fire* is **not** supported in the LITE edition

Main settings:

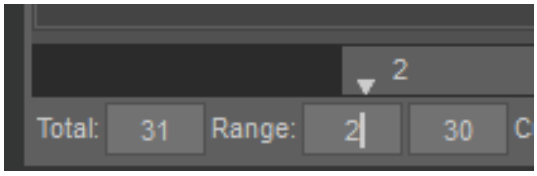


Besides *Fluid type*, *the Main Settings* parameters are the following:

*Number of frames*: is the number of frames to simulate.

* *Current frames range*: if enable, the simulation will run on the range of frames visible in the *Timeline*, instead of the *Number of frames* (it will not be showed when *Current frames range* is enabled).

You can continue or start a simulation from any frame. The engine will run from the first to the last frame in the Timeline range. You can change the range by editing the Range boxes of the Timeline or by dragging the white sliders at the bottom of the Timeline:




*Stabilization steps:* The engine will run the number of steps indicated by this property without advancing the timeline (there will be substeps if CLF is low enough) and no baked file will be written until the stabilization steps are finished unless the *Stabilization visibility* is ON.

*Stabilization visibility*: if this one is enabled, the engine will write all the baked files during the stabilization steps and if the user has a mesher enabled, he will be able to see the advancing of simulation. however, the simulations are faster if the *Stabilization visibility* is disabled. This property is hidden if the *Stabilization steps* property is set to 0.

*Frames per second*: it is the frames per second of the simulation result, not the rendering ones.

*Cell size*: is the cell size of the fluid grid simulation in internal units of simulator. The smaller, the better simulation and the slower too. If the cell size halves, the total number of cells for the calculation grows cubically (8-fold increase). Beware, it can use all the physical memory, specially in smoke or fire simulations.

*Size X*, *Size Y* and *Size Z* correspond to the width, height and depth as usual in Daz Studio. The units are centimeters, the default of Daz Studio.

*Subdivision level*: The surface subdivision level determines how many times the simulation grid is divided when converting marker particles into a triangle mesh. For example, a simulation with dimensions 256 × 128 × 80 and a subdivision level of 2 will polygonize the surface on a grid with dimensions 512 × 256 × 160. With a subdivision of level 3, the polygonization grid will have dimensions 768 × 384 × 240. A higher subdivision level will produce a higher quality surface at the cost of longer simulation times and greater memory usage.

Subdivision level value in *Fluidos II* is a real number (it accepts decimals), unlike the old *Fluidos* where is an integer number.

*Anisotropic:*
Isotropic triangle meshes are the default form of program output. These triangle meshes represent the surface of the fluid and are constructed by the fluid simulator from a set of spheres with uniform radius. The meshes are written to the baked files/ subdirectory as a sequence of .PLY files in the form 000000.ply, 000001.ply, 000002.ply, ..., where the file numbers correspond to the frame number.

Anisotropic triangle meshes represent the surface of the fluid and are similar to isotropic meshes except that the meshes are constructed from a set of ellipsoids instead of a set of spheres. The benefit of constructing the surface from a set of ellipsoids rather than a set of spheres is that sharp/smooth features of the fluid surface are better preserved. This benefit comes at the cost of a longer surface mesh computation time.
The anisotropic meshes are written to the baked files/ subdirectory as a sequence of .PLY files prefixed with the keyword anisotropic in the form anisotropic000000.ply, anisotropic000001.ply, anisotropic000002.ply, ..., where the file numbers correspond to the frame number.

To get anisotropic meshes, the user must select the *Anisotropic* button.
*Fluidos II* does not generate anisotropic meshes, although the mesher could read them (however the *Fluidos* engine mode could generate these meshes)

*Vorticity confinement constant*: this property allows preserving and enhancing the vorticity, the higher the property the more vivid the fluid. Too high values could produce exaggerated results. It is animatable.

*Baked files folder*: The user must set here the folder where the simulation files will be saved.

*Erase baked files*: By *en*abling this property, the old files in the current folder will be erased the next time the simulator is running. However, if the user set ON the “Continue saved state” option in running simulation, the *Erase baked files* property will be ignored.

*Enable saving Fog-type VDB*: If this property is enabled (the default is *On*), the plugin will save fog-type .vdb files that Iray can read as volumetric smoke and fog, by using the DAZ Studio’s *OpenVDB MDL Volume shader.* The prefix of the files is “*fogfluid*”. See *Volumetric OpenVDB* mesh type for more details.

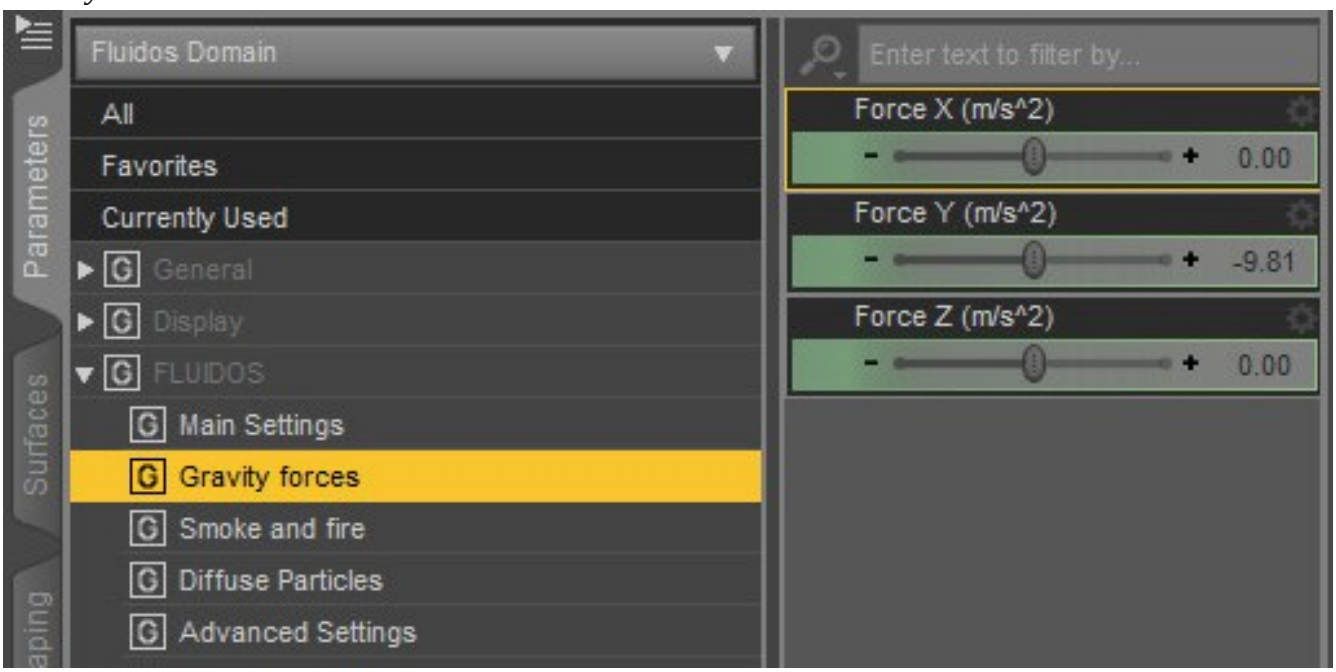
*Remove isolated particles*: If this property is enabled, the particles that are away of the mass of fluid in the simulation, are removed. It could be useful in fog simulations to dissipate the isolated particles that are as little spheres floating in the scene. It is animatable.

*Preset file*: The user should set here a preset file, to do so you must change the name of the file or move it to another folder (the default name is autosave.state) *Start from preset* (at *Run Simulation*) will start a 1simulation from this preset, that is, from a simulation's *state* file (see appendix B for more details).

*Fluid mass nodes in preset*: by enabling this property, the user can add new fluid masses to a preset (see appendix B for more details).

*Enable moving obstacles:* this option allows to simulate the effect of the obstacle in the fluid. *The CFL Condition number*, however must *be low,* less than 2.0, specially when the velocity of solid is high.

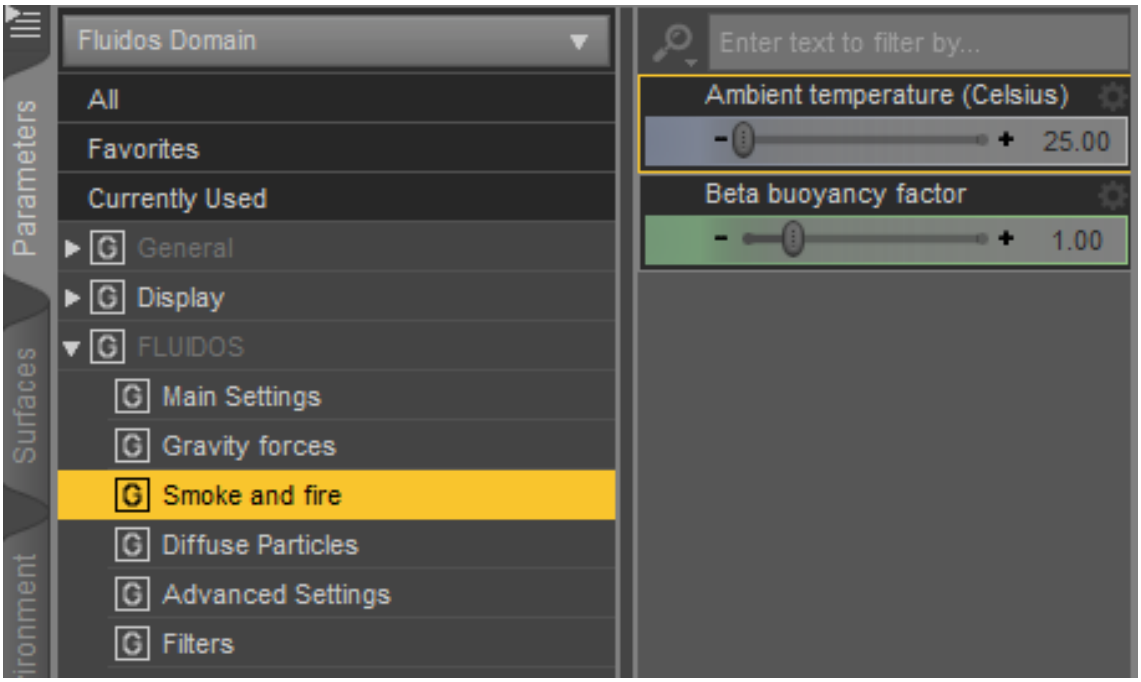
*Gravity Forces:*




*Gravity forces*: Add a spatially constant force such as gravity to the simulation. Nevertheless, this force is an animatable parameter. The units are m/s2.

Gravity forces are *the same* as the old *Fluidos* body forces. They were renamed because smoke simulations uses them in a different way than the other body forces.

*Smoke and fire:*




These properties are only available for *Smoke* simulations.

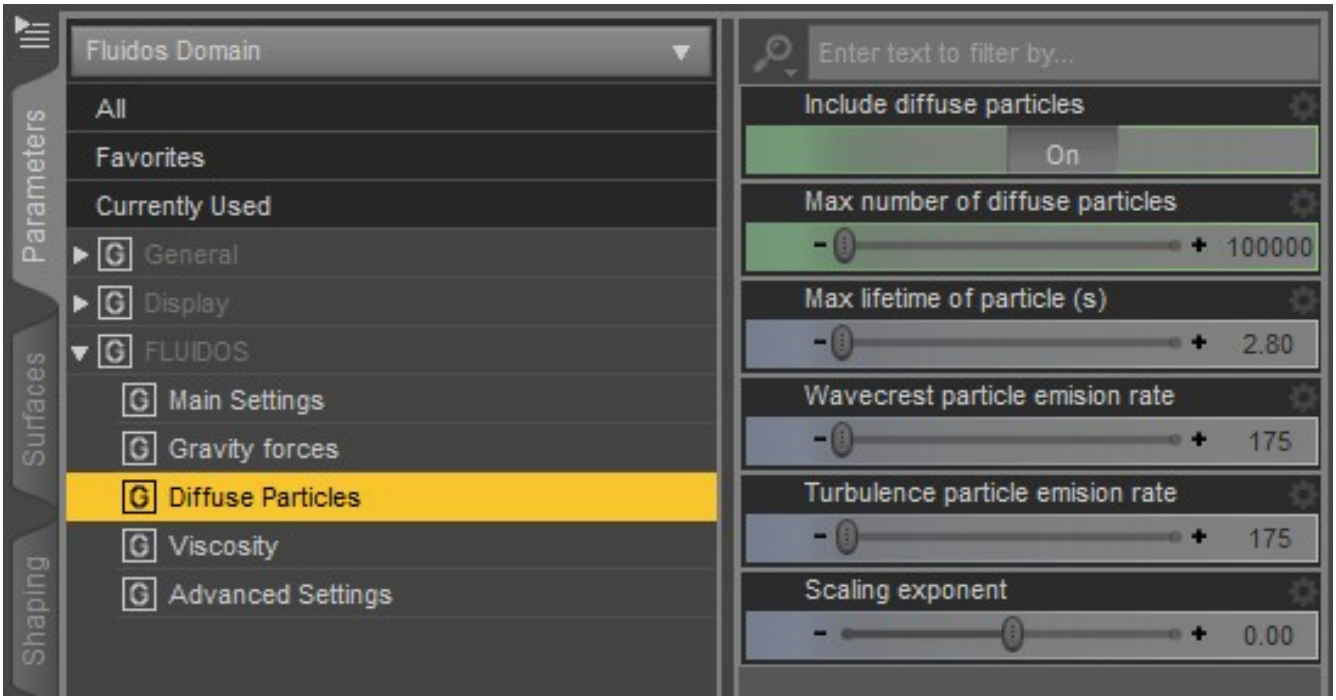
There is only one method to solve smoke simulation in LITE edition, a simplified form of the Boussinesq method. This does not depend on concentration nor density properties.

*Ambient temperature*: the air temperature.

*Beta buoyancy factor*: It increases or decreases the buoyancy without changing the temperatures (but remember that ambient temperature and fluid temperature must be different anyway). Default is 1.0

Go to appendix D to see the effects and differences of *smoke and fire* properties.

*Diffuse Particles*




*Diffuse particles*:
The diffuse particle feature is a post-processing simulation that runs on top of the PIC/FLIP fluid simulation that generates spray, bubble, and foam particles. These diffuse particles can be combined with a surface mesh in a render to give the fluid highly detailed small-scale aeration effects. The diffuse particles are stored as vertex only .PlY meshes where each vertex represents a single diffuse particle.

The diffuse particle meshes are written in the baked files/ subdirectory as a sequence of .PLY files prefixed with the keyword diffuse in the form diffuse000000.ply, diffuse000001.ply, diffuse000002.ply, ..., where the file numbers correspond to the frame number.

To enable diffuse particles click on the *Include diffuse particles* check box. Beware, although the simulator can manage easily a great number of particles, its representation on Daz Studio can be very costly and result on a very slow rendering or locking of the program. You can limit the *maximal number* of diffuse particles with the slider.

The *maximum lifetime* of a diffuse particle is measured in seconds. Set this value to control how quickly/slowly diffuse particles fade from the simulation.

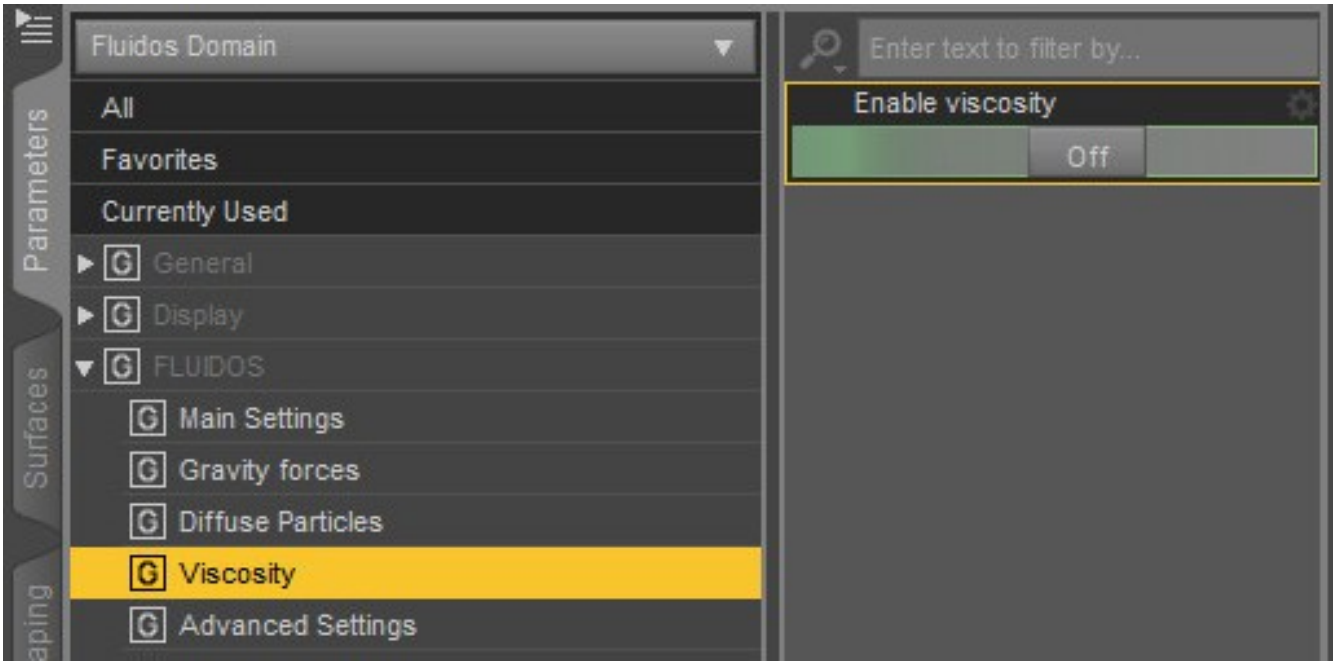
*Diffuse particle emission rates:*
The diffuse particle simulator spawns particle emitters in areas where the fluid is likely to be aerated such as at wave crests and in areas of high turbulence. The number of emitters spawned in an area is proportional to how sharp a wave crest is and how turbulent the fluid is at a location.

The number of particles generated by an emitter is controlled by two rates: *Wave crest emission rate*, and *Turbulence emission rate*. An emission rate is the number of particles generated by an emitter per second. The wave crest emission rate controls how many particles are generated by wave crest emitters. The turbulence emission rate controls how many particles are generated by turbulence emitters.

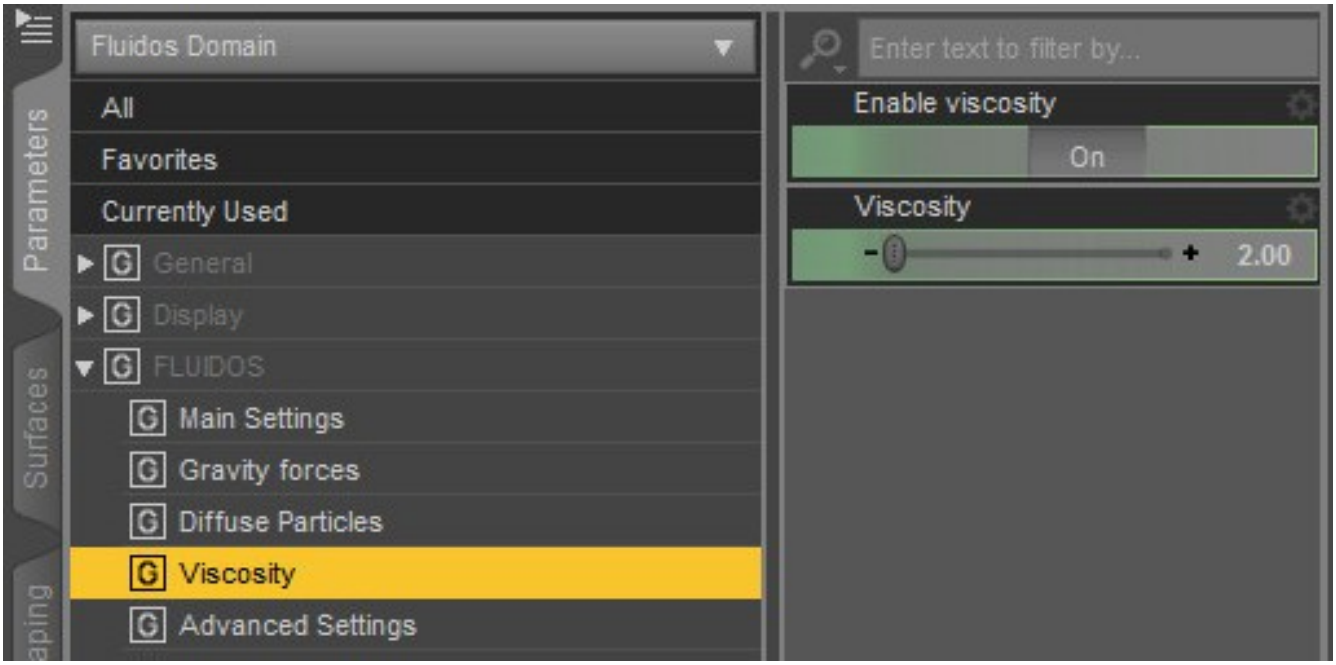
*Scaling exponent*: This property tune the distribution of the diffuse particles by their potential energy. This energy depends strongly on the domain size. A low exponent could emit diffuse particles mainly in small turbulent regions, whereas a high exponent will cause the emission mainly in large turbulent regions.

An important note to make about emission rates is that the number of particles generated scales as the simulator dimensions scale. This means that a simulation with dimensions $128 \times 128 \times 128$ will generate about eight times as many diffuse particles than a simulation with a dimension of $64 \times 64 \times 64$ when using the same rate values.

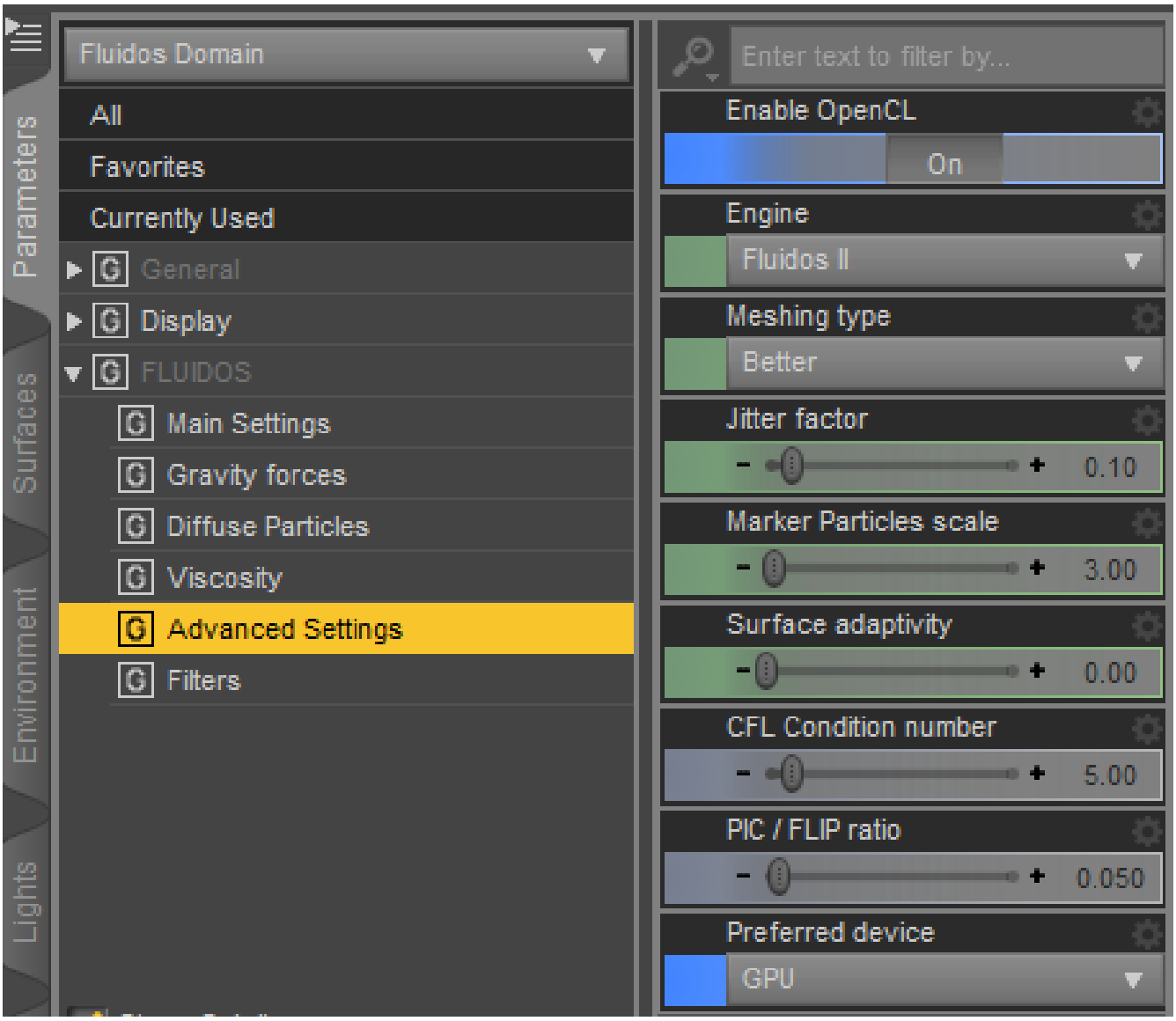
*Viscosity*




*Viscosity*: To simulate high viscosity fluids (as oils, honey, etc.), set *Enable viscosity* On and put a viscosity value. The value is animatable.

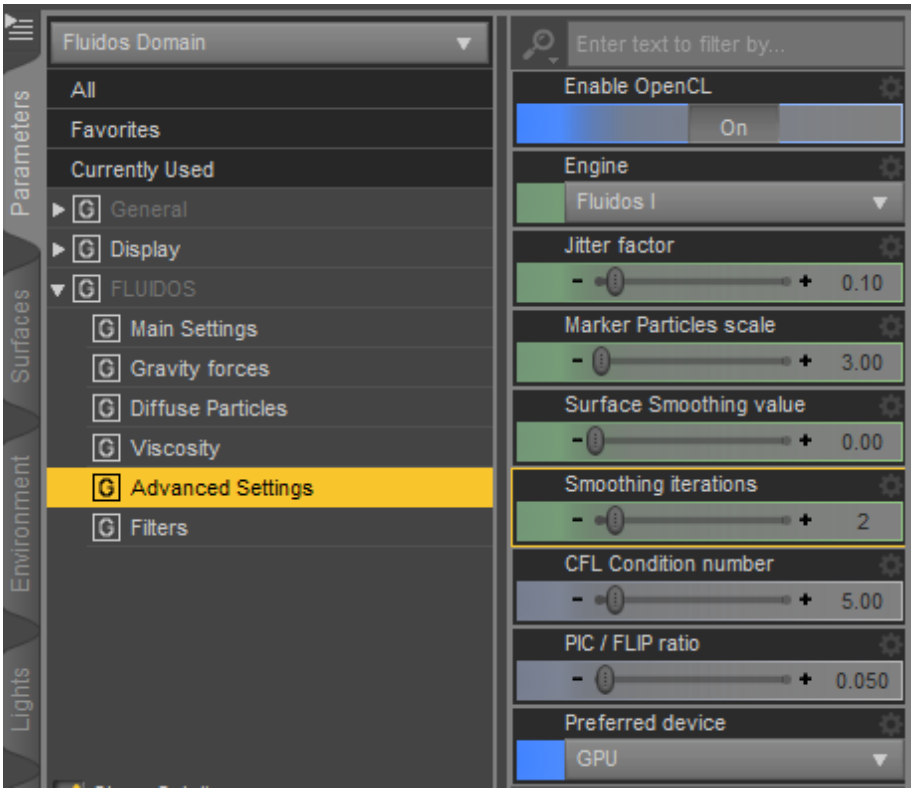


*Advanced Settings*




*Enable OpenCL*: this allows to switch off or on the OpenCL. The simulation without OpenCL is somewhat slower and the results are slightly different.

*Engine*: This property has two options: *Fluidos I* and *Fluidos II*. The first one force the plugin to use the old (but faster) Fluidos engine. Selecting *Fluidos I* will hide the new properties, including smoke and volumetric fire.




*Jitter factor*: This property controls the amount of random jitter added to newly spawned fluid particles. Sometimes, low values may produce symmetric artifacts into the resulting fluid simulation. This can be avoided by increasing the jitter factor.

A high jitter factor is more convenient in smoke and fire simulations.

*Marker particle scale*: Marker particles track where the fluid is and carry velocity data. The marker particle scale determines how large a particle is when converting a set of particles to a triangle mesh. A marker particle with a scale of 1.0 will have the radius of a sphere that has a volume 1/8th of the volume of a grid cell. It is very useful for viscosity simulations because sometimes the fluids freezes in the air. Increasing the marker particle scale fixes this.

*Meshing type*: there are two main meshing types: *Faster* and *Better.* The latter is more precise but slower. There is another one, *Better-Trail*. This one, elongates a little the mesh in the direction of the particle velocity.

*Surface smoothing value* (only *Fluidos I* engine): The amount of mesh smoothing when generating the mesh surface

*Surface adaptivity*: this adaptively reduces the number of polygons in the fluid mesh. The more planar regions will have a lower amount of polygons.

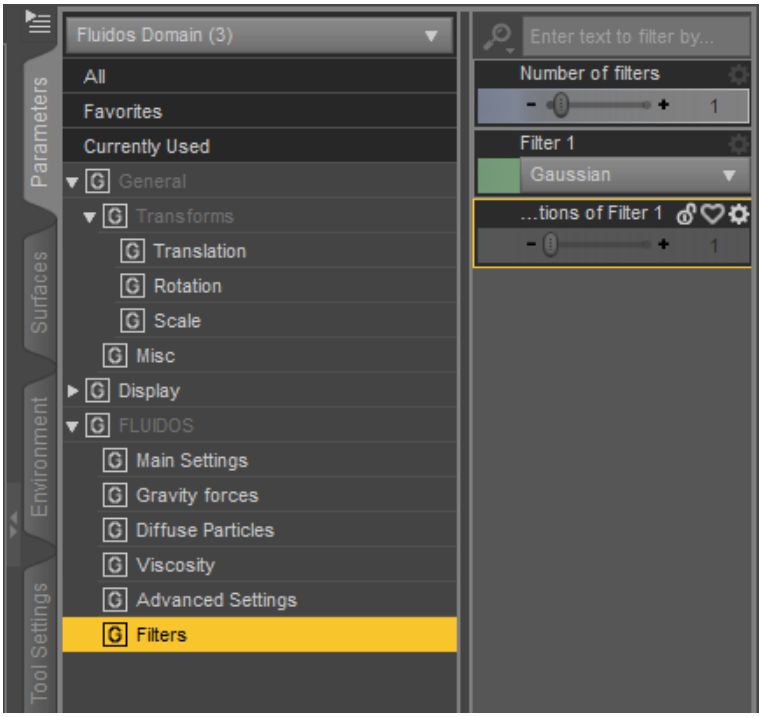
*CFL Condition number*: This is the maximum number of grid cells that a particle may travel in a single time step. **The larger is the number, the faster; the smaller and the more accurate.**
**BEWARE**: combining *low values* of CFL numbers with *small* cell size could cause the simulations to take a lot longer to finish.
If you experience long-time simulations, you may want to increase the CFL number.
The low limit of CFL in the plugin is *0.1*. But **a *0.0* value disables the CFL condition number, so the simulations are faster but less accurate**.

*PIC/FLIP Ratio*: The ratio of PIC velocity to FLIP velocity to use when updating particle velocities. Particle-in-Cell (PIC) and its variant Fluid-Implicit-Particle (FLIP) are the engine simulation methods. The PIC method is not very accurate but it's stable. The FLIP velocity method is very accurate, but less stable. Using a value of *0.0* results in a completely FLIP simulation, while using a value of *1.0* results in a completely PIC simulation.

*Preferred OpenCL device*: The primary OpenCL device (GPU or CPU). For some computer systems, CPU is better. If you have a second GPU, you can select it ("GPU 2").

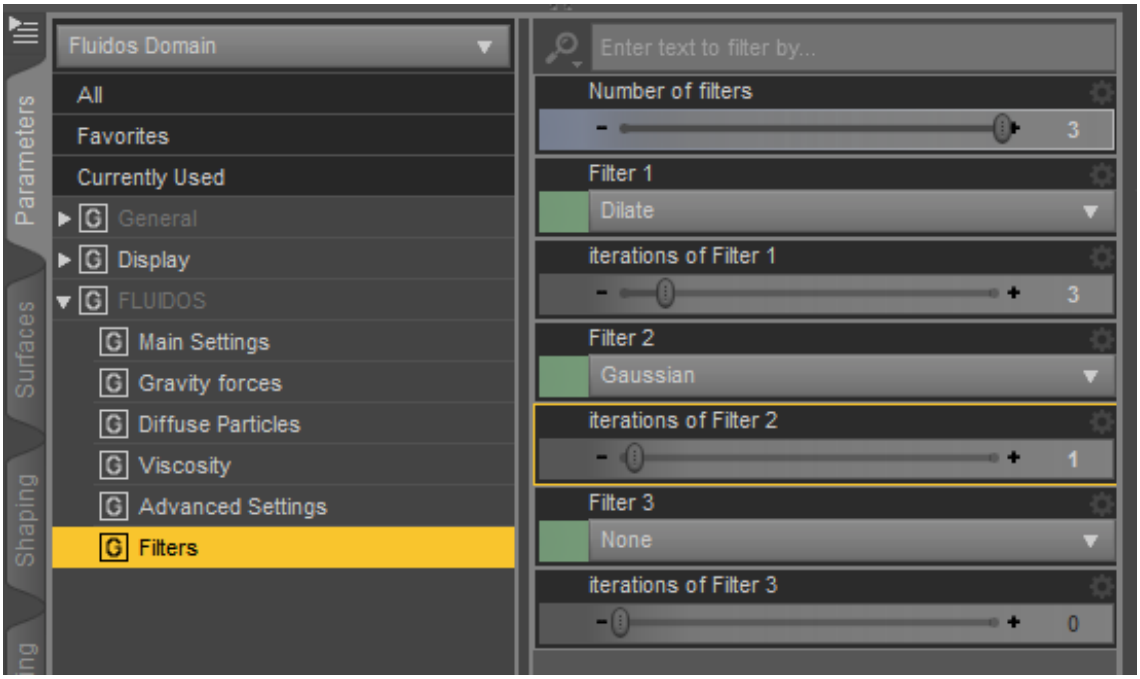
Note 1: If your system has only one GPU, and you select GPU 2, the plugin will use the unique GPU.
Note 2: As *Fluidos II* is multithreaded, for systems with less powerful graphic cards, disabling OpenCL could yield faster calculations.

*Filters*

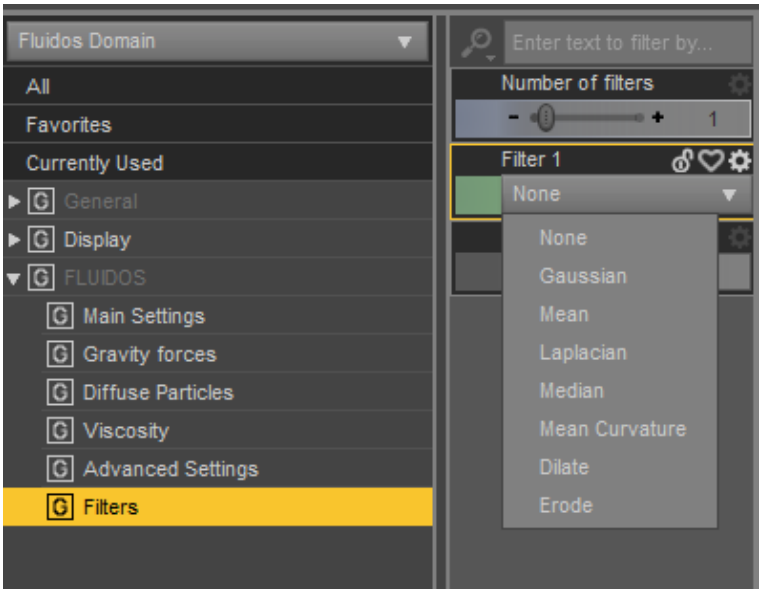



The OpenVDB filters are not in the *Advanced Settings* group anymore, but in a new group: *Filters*. By default there is a Gaussian filter enabled.

*Number of filters*: this property determines the number of filters to be applied. Each filter will be applied on top of the previous filter. You can apply up to 3 stacked filters.

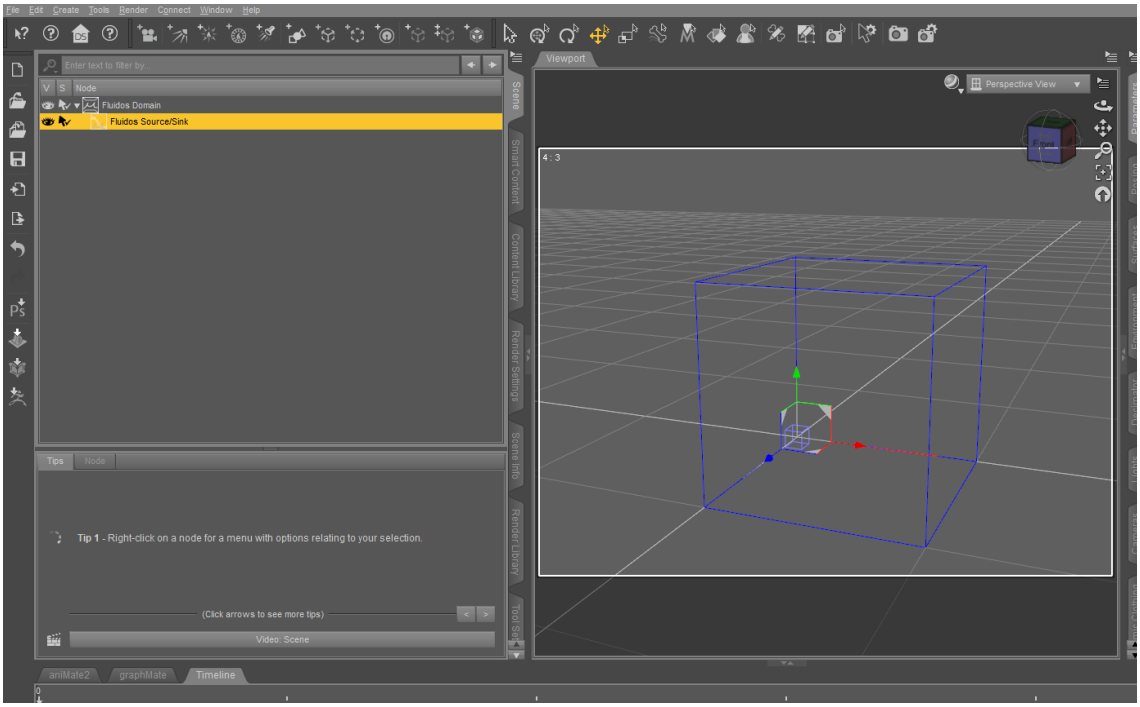



*Filter type*: there are five options: *Gaussian*, *Mean*, *Laplacian*, *Median* and *Mean Curvature*. There are two new filters available: *Dilate* and *Erode*. The first one will tend to fatten the final mesh, whereas the latter will produce the opposite effect. See the appendix D for graphical description.




*iterations:* The number of iterations of the filter.

**Fluidos Source / Sink**

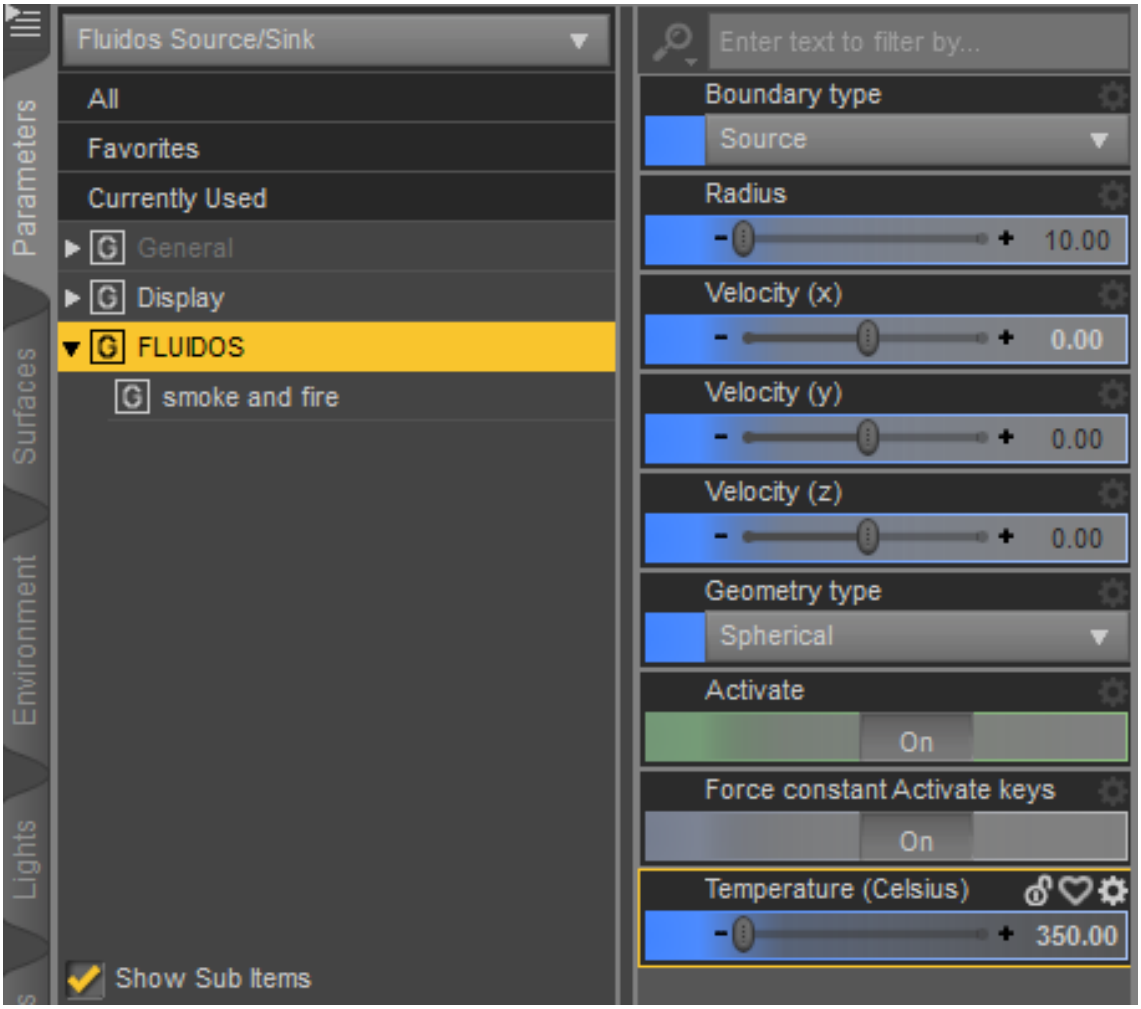



The source (or the sink) are represented as an empty cube, but can be resized just like the fluidos domain.

A cyan line will indicate the velocity's direction and magnitude.

If two or more Source/Sink objects completely overlap in some region, the first one in the hierarchy takes precedence. A partial overlap could produce undesired effects due to interference between the objects.

Parameters of Fluidos Source / Sink



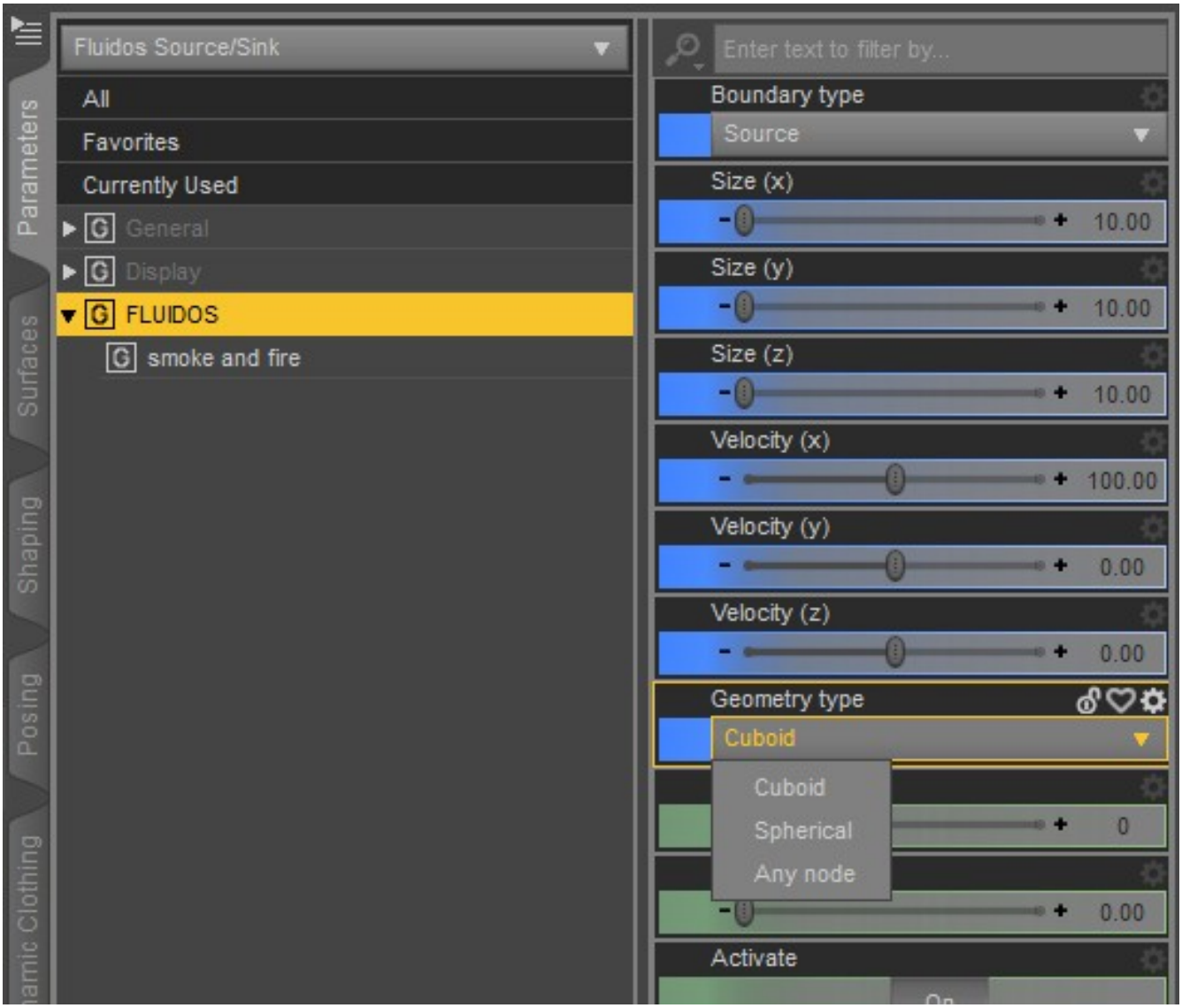

The size can be controlled with the parameters *Size (x)*, *Size (y)* and *Size (z)*, the units are centimeters. The velocity of flux is in cm/s and is controlled with *Velocity (x)*, *Velocity (y)* and *Velocity (z)*.
Size and velocity are animatable parameters

*Boundary type*: this property substitutes the old *Is Source* property. It has two options: *Source* and *Sink*.

The new group *smoke and fire* contains the next properties (all of them are animatables):

> *Temperature (Celsius)*: this is the temperature of the fluid. It will be ignored if the Domain's *Fluid type* is *Liquid* or if the *Boundary type* is *Sink*. If a *Source* is inactive, its spanned region will force this temperature in non-liquid simulations.

*Geometric type:* This property allows to choose the geometry of the Source/Sink object. There are three options: *Cuboid*, *Spherical* and *Any* node. If *Cuboid* is selected, the geometry of the source (or sink) is a rectangular prism; *Spherical* produces a spherical geometry (not visible in the screen representation of the object). *Any node* lets the user to use any geometrical (closed mesh) object inside the hierarchy of the Domain.

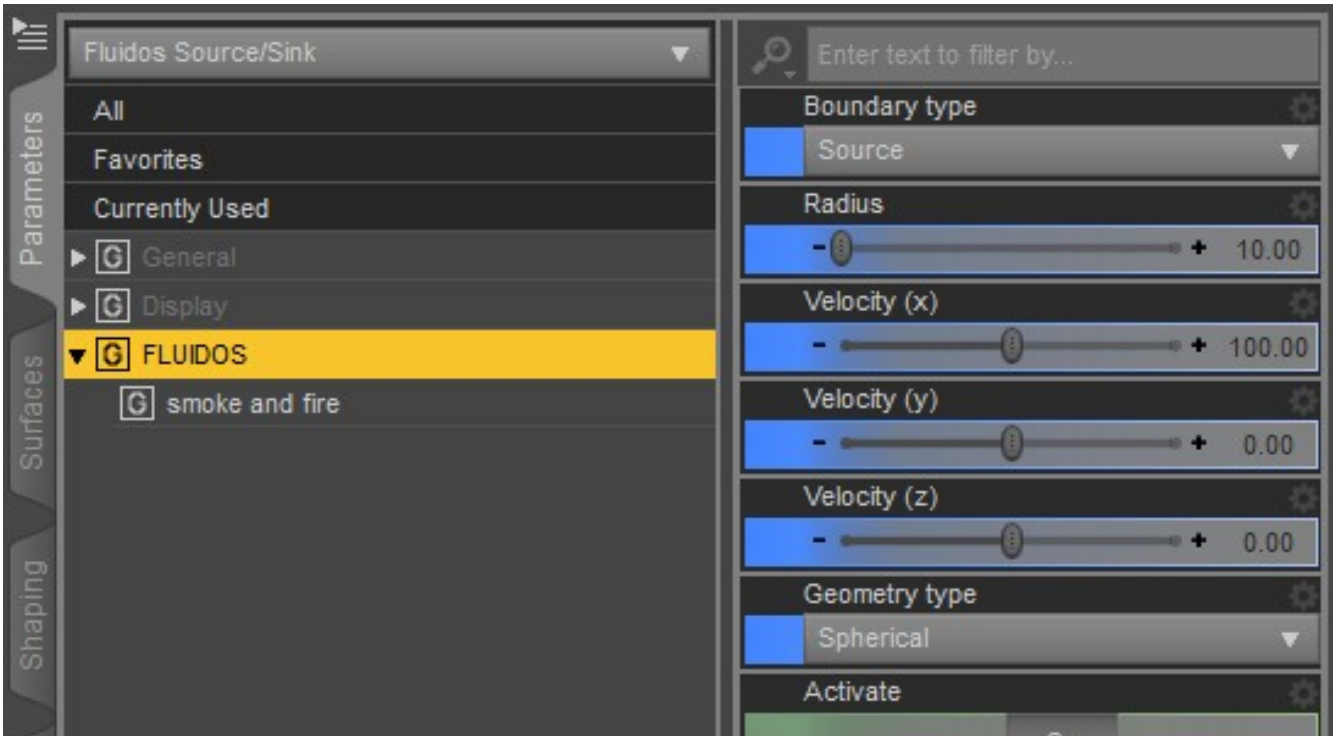



If *Cuboid* or *Any node* is selected there will be two extra properties: *Holes density* and *Randomness*. These properties can be used to generate a spatially discontinuous fluid source, simulating a set of sub-sources instead only one source. They can be useful for the simulation of rain or showers. Increasing *Holes density* increase the distance between the sub-sources (and decrease their quantity). *Randomness* controls the changes of sub-sources positions frame by frame; high *randomness* produces individual fluid drops useful to simulate rain. The default values produce only one cuboid source.

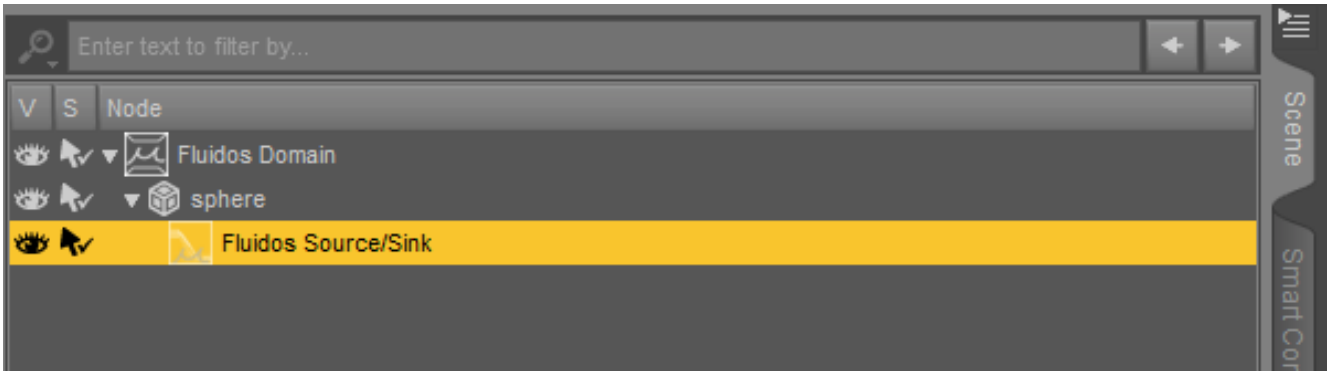
*Activate*: ON to activate the source (or sink), *OFF* to deactivate; this is animatable (e.g. you can interrupt a flux at any time).

*Force constant Activate keys*: if this parameter is turned *ON*, the key frames of *Activate* are treated as a constant interpolation: if the user adds a first key frame as *ON* and the next as *OFF* for the *Activate* parameter, the plugin will add an *OFF* key just a frame before the last one. This way, the source is active until the user sets it to *OFF*.

When *Spherical* geometry type is chosen, the size of the object is controlled by only one number: the *radius* (cm), it behaves as a sphere in the simulation.

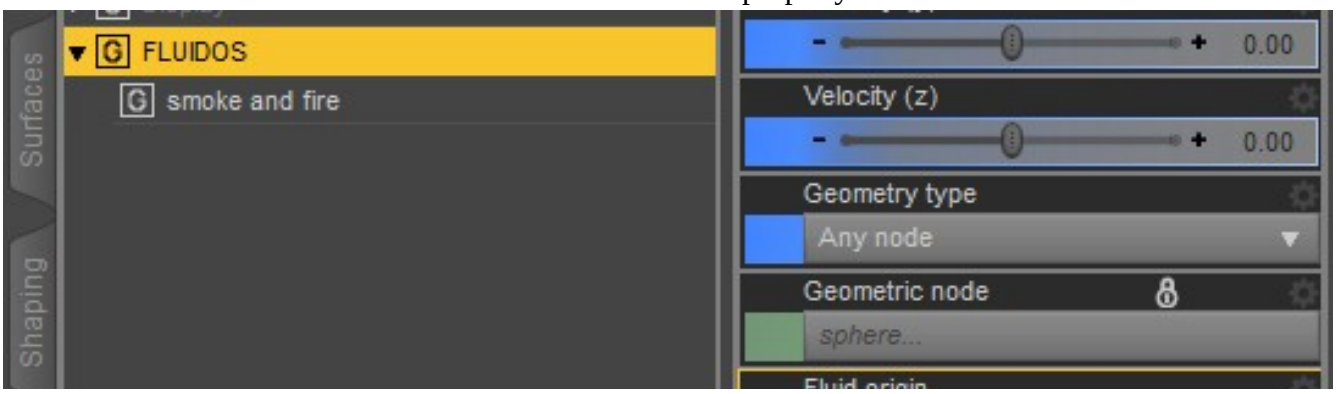



To use *Any node* geometric type, the user must directly parent the Source/Sink to the desired node. This node should be a geometric one, closed mesh.

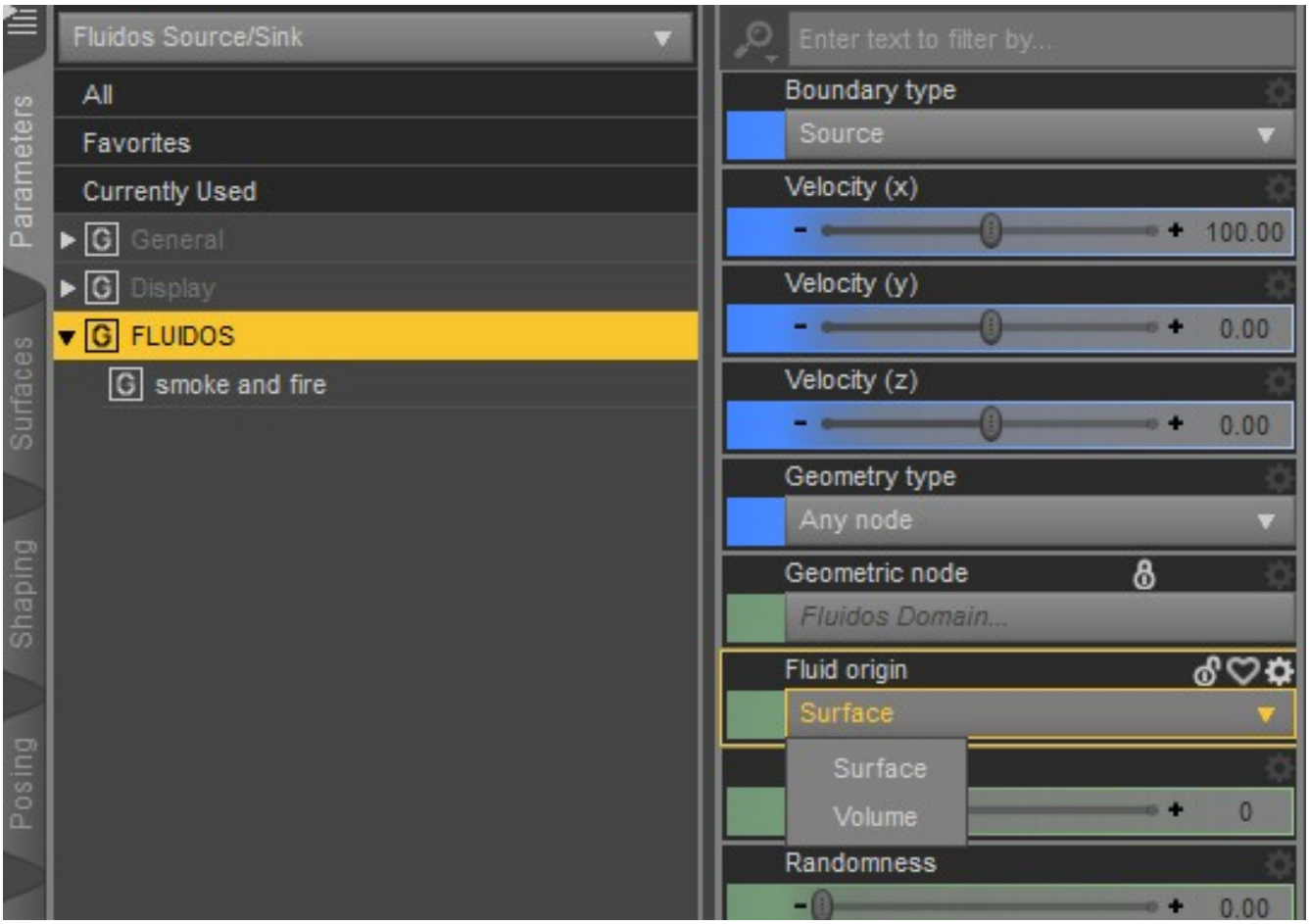



Anyway, when the user asks the plugin to run the simulation, the plugin will ask for a geometric node if a Source/Sink *Any node* type is parented to a node without geometry (e.g. a null, a bone or the Fluidos Domain itself), the plugin will ask the user to choose other node. If the user decides not to choose other node, the simulation will ignore this Source/Sink.

The chosen node will be indicated in the Geometric node property.




*Fluid origin options*



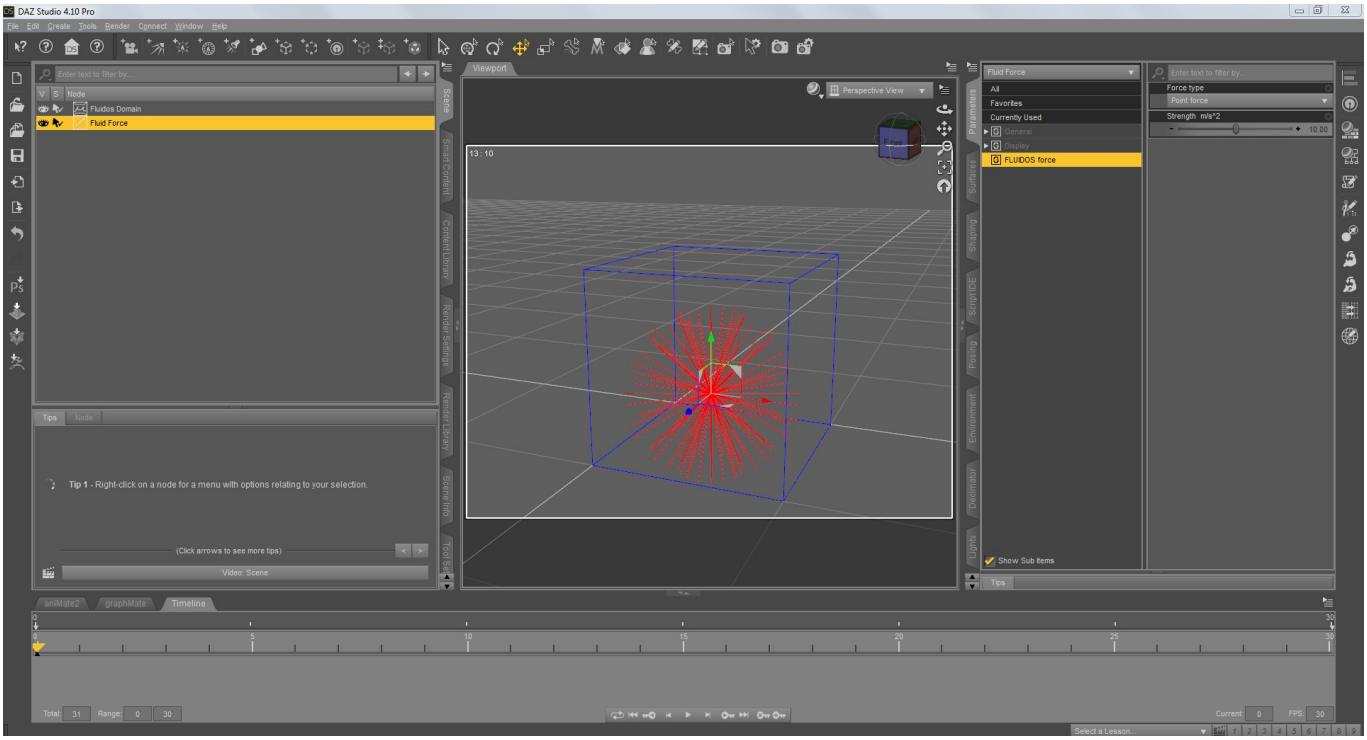

The user can choose where in the node the fluid will come from: *Surface* or *Volume*. The first option will produce a fluid layer over the geometric node. From this layer, the fluid will enter to the Domain. In the other hand, when *Volume* option is selected, the fluid will input from the whole current geometric node's volume, without adding extra layers.

Nonetheless, the geometry node itself *will conserve its properties as Fluidos object*. It could be a fluid mass, an obstacle, a viscosity control or ignored, as usual. The source won't affect these. If you source is Surface fluid origin, the geometric node could an obstacle, fluid mass, etc. without problem. But for Volume fluid origin, the node mustn't be an obstacle because it would block the flow (for Surface there is no restriction as the fluid layer is outside the node geometry).

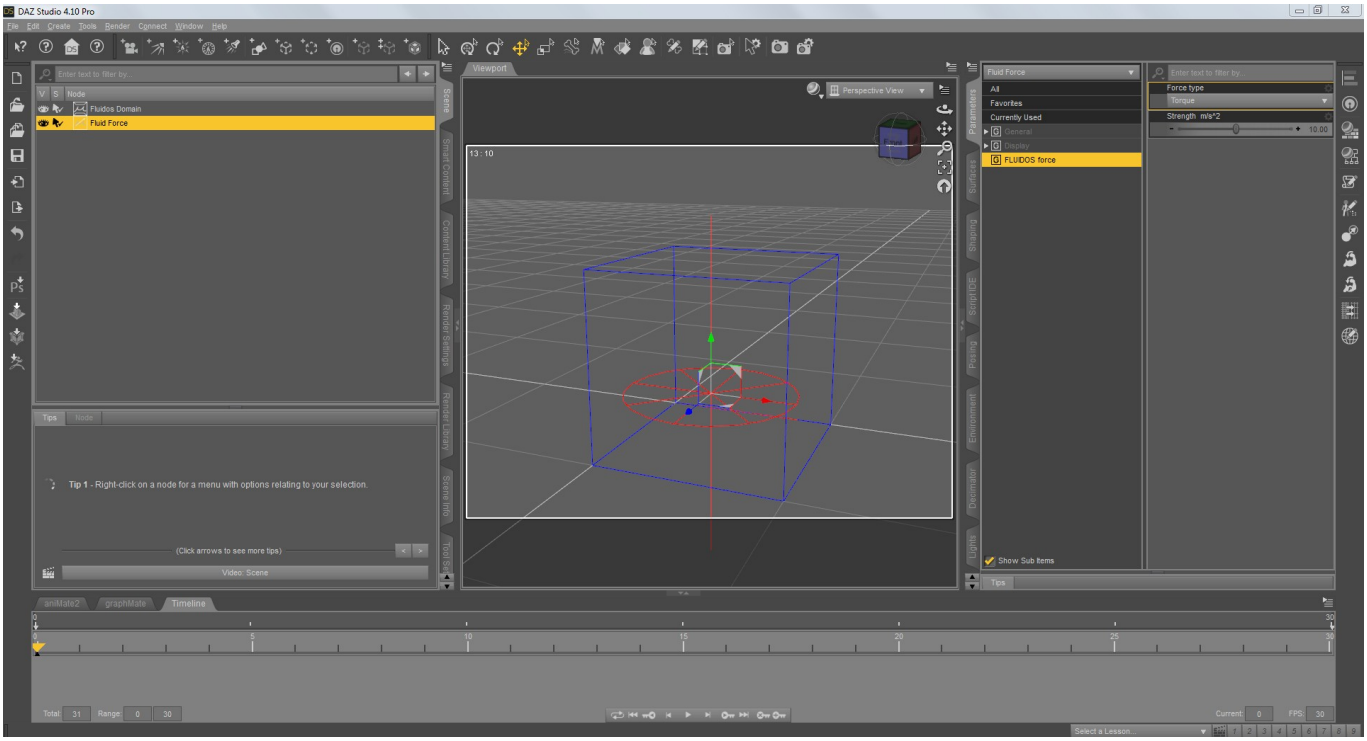
**Fluidos Forces**

The Fluidos forces are represented in this way:

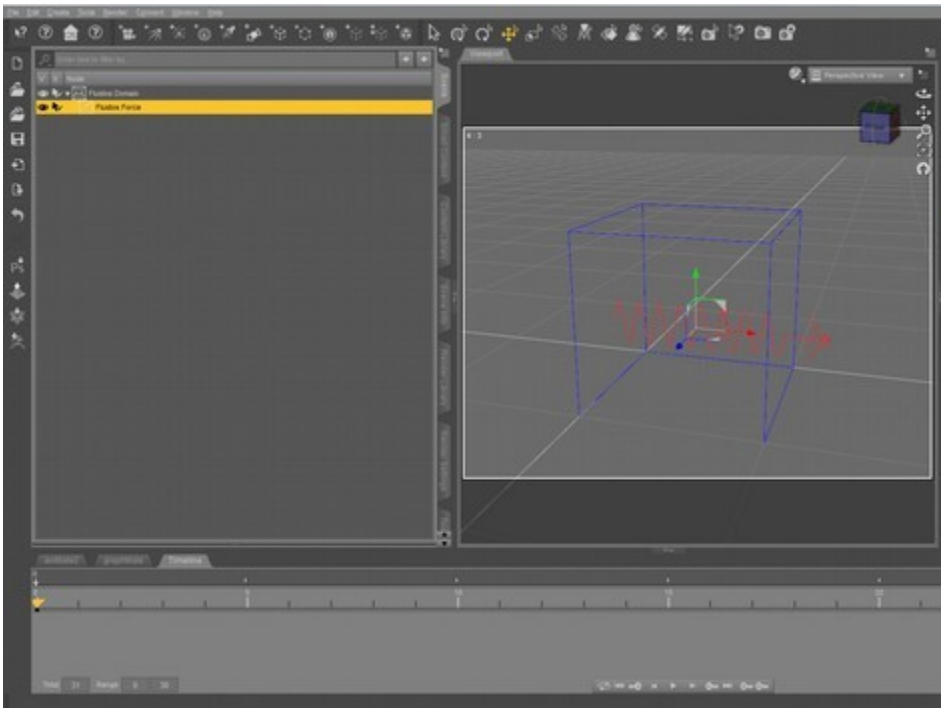
*Point force*



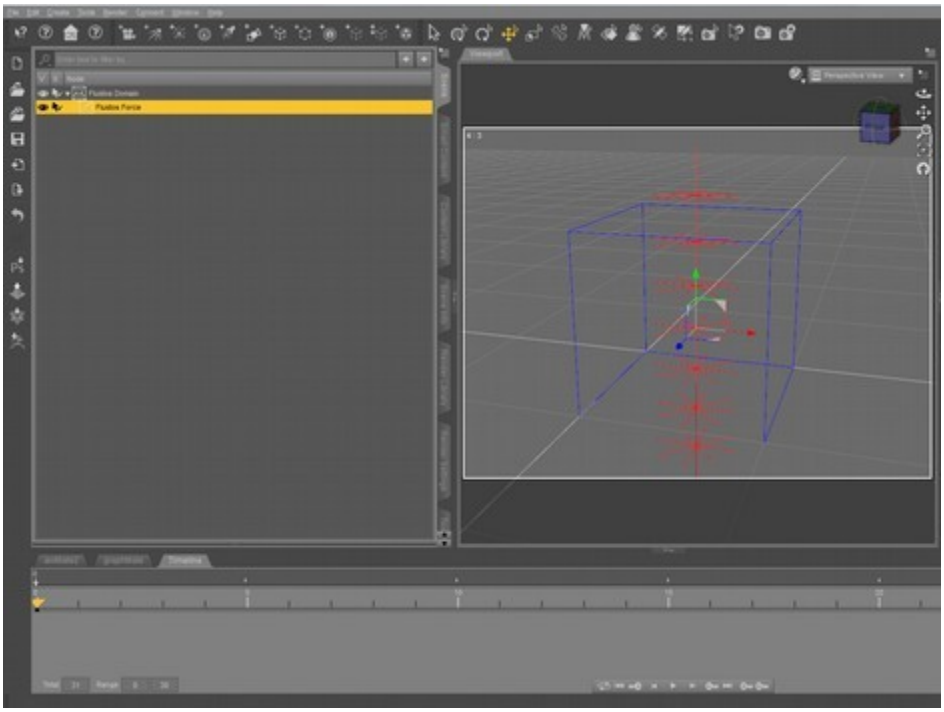

*Torque force*



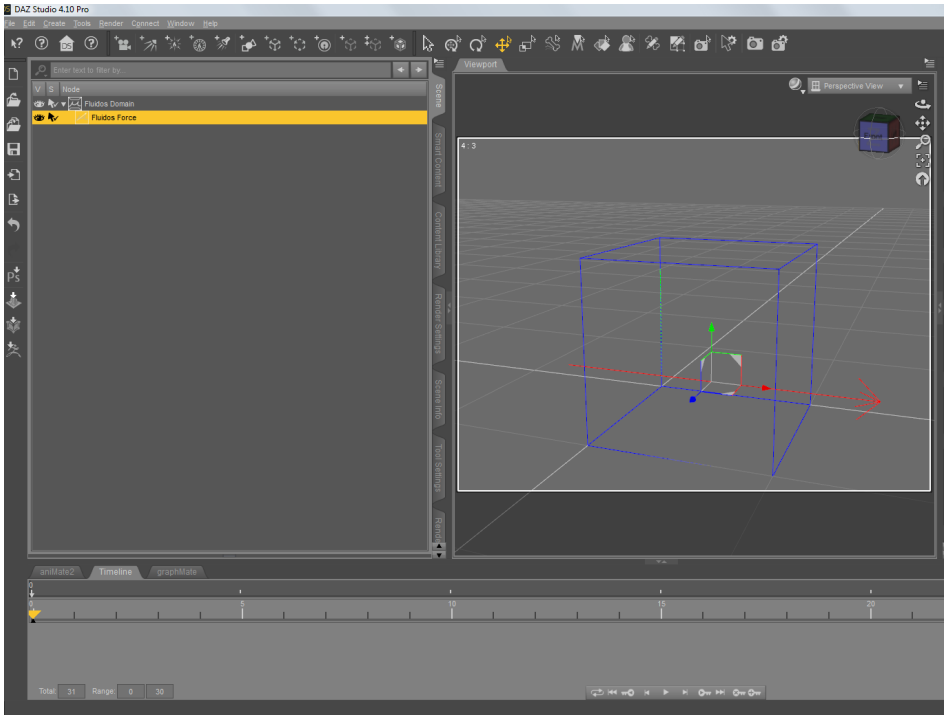
*Flow force*



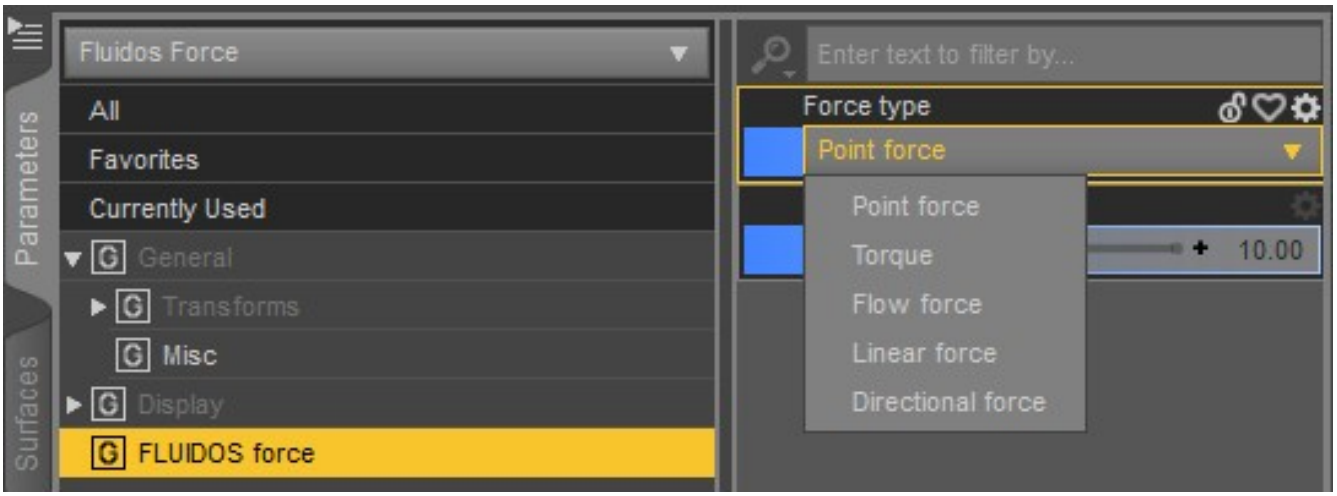
*Linear force*

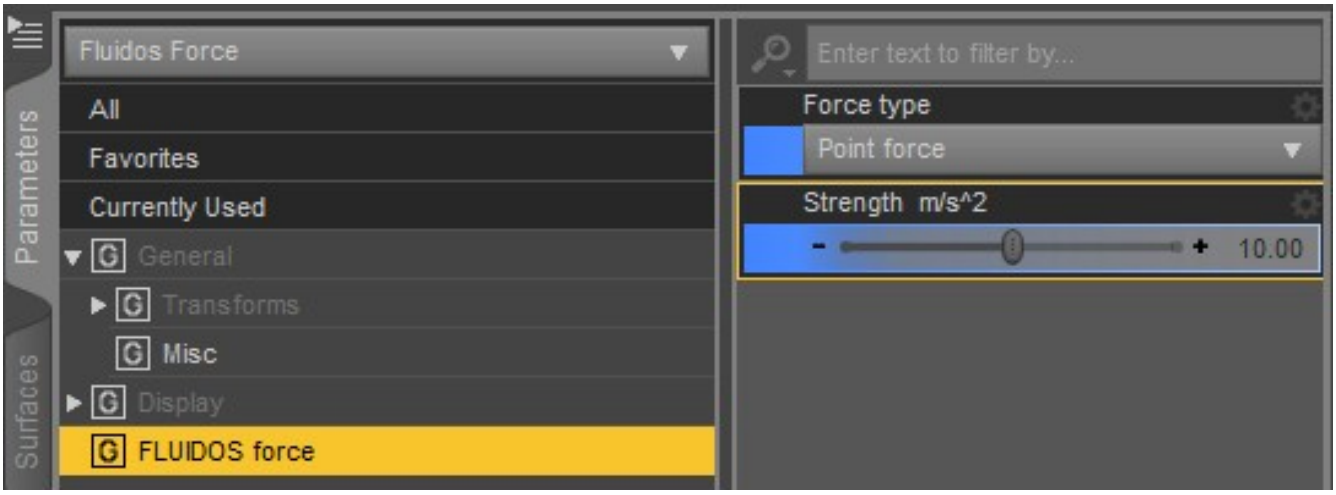


*Directional force*

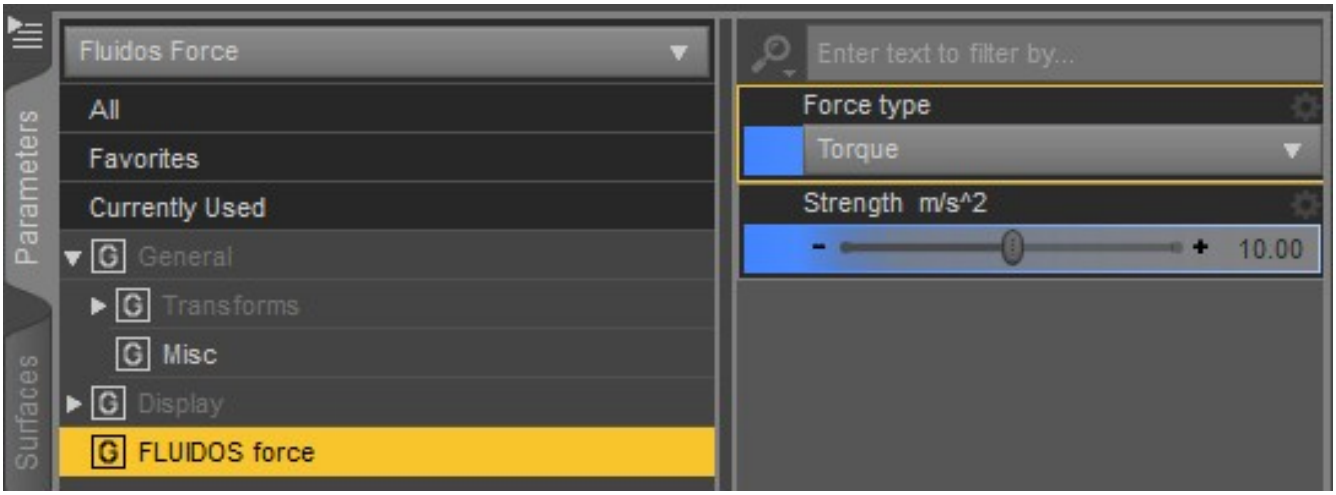


Choosing the *Force type:*

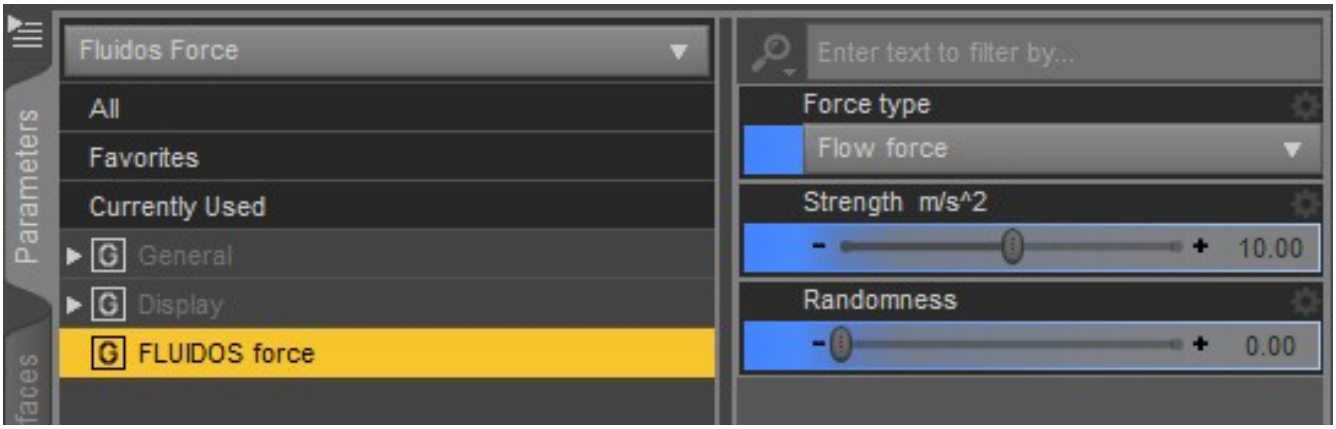




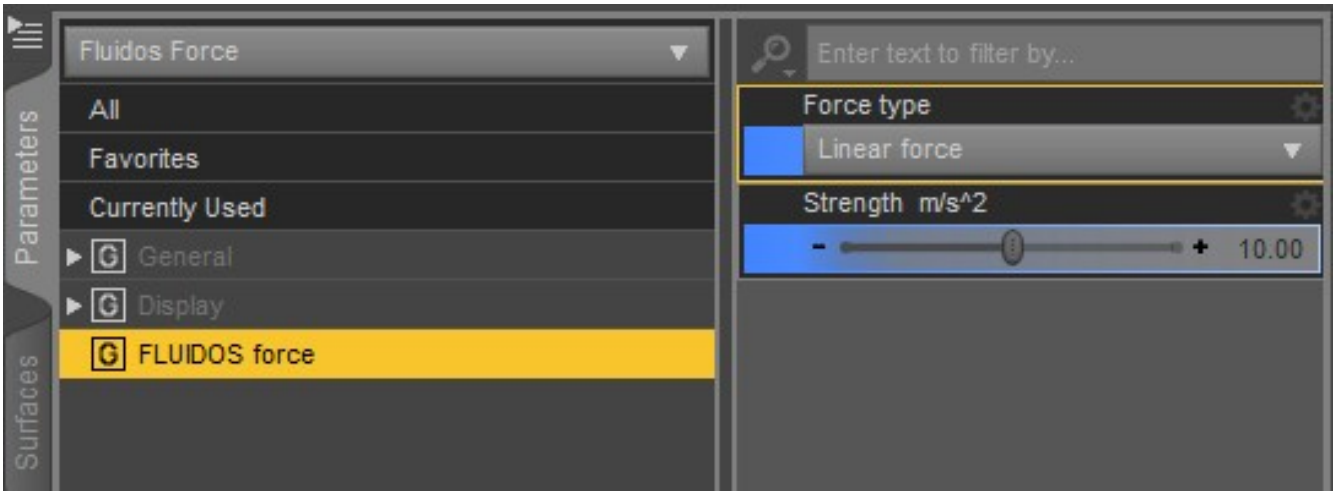

*Point Force*: This force extends its action from a point in space, the decay rate is quadratic. Its unis are in *m* /s2 (acceleration).



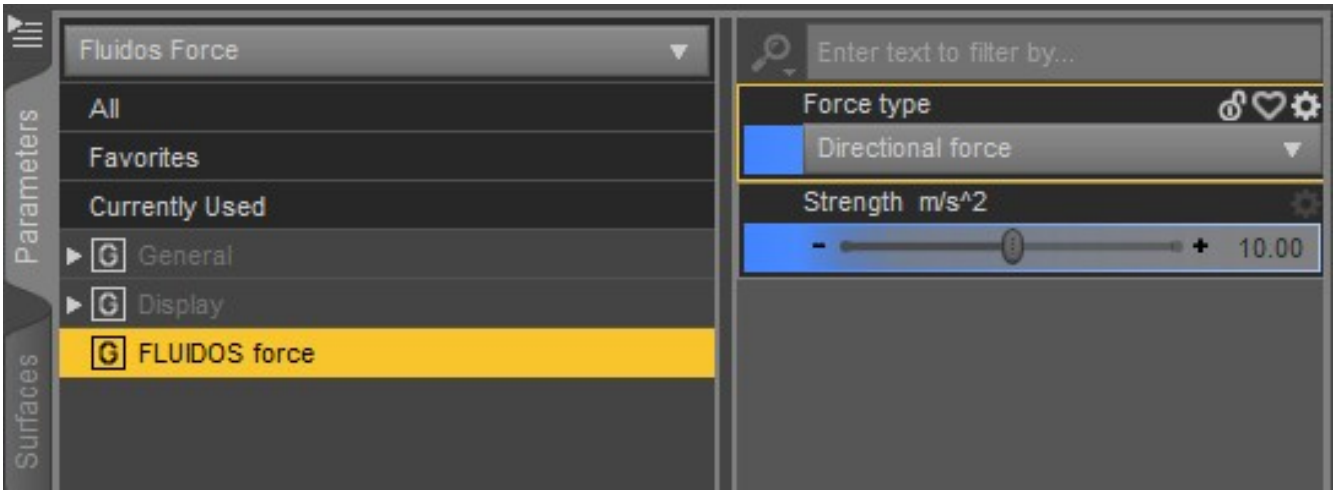

*Torque Force*: this is a rotational force. As in the case of point force, the position is relevant, but the rotation is also relevant. Its unis are in *m* /s2 (acceleration).



*Flow Force:* This force extends its action over all the fluid free surface (not in solid contact). Its units are in *m* /s2 (acceleration). The *randomness* property controls the wave's density on the fluid surface. The position of this force in not relevant, only the rotation.




*Linear force:* This force extends its action form a straight line in space, the decay rate is quadratic. Its unis are in *m* /s2 (acceleration). The position and rotation are relevant variables.



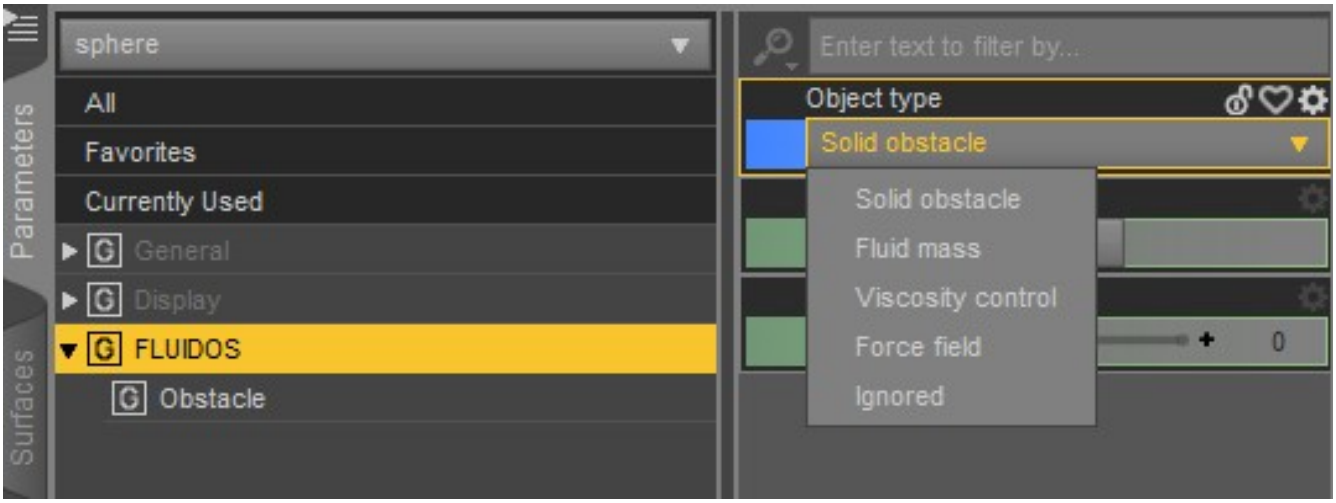

*Directional force:* This force extends its action all over the space. Its direction is indicated by the arrow. Its unis are in *m* /s2 (acceleration). The rotation is a relevant variable, but position is not.

**Object Properties:**

When any geometric object is parented directly or indirectly to a Fluidos Domain, it will acquire some FLUIDOS parameters, the most important one is *Object type*.

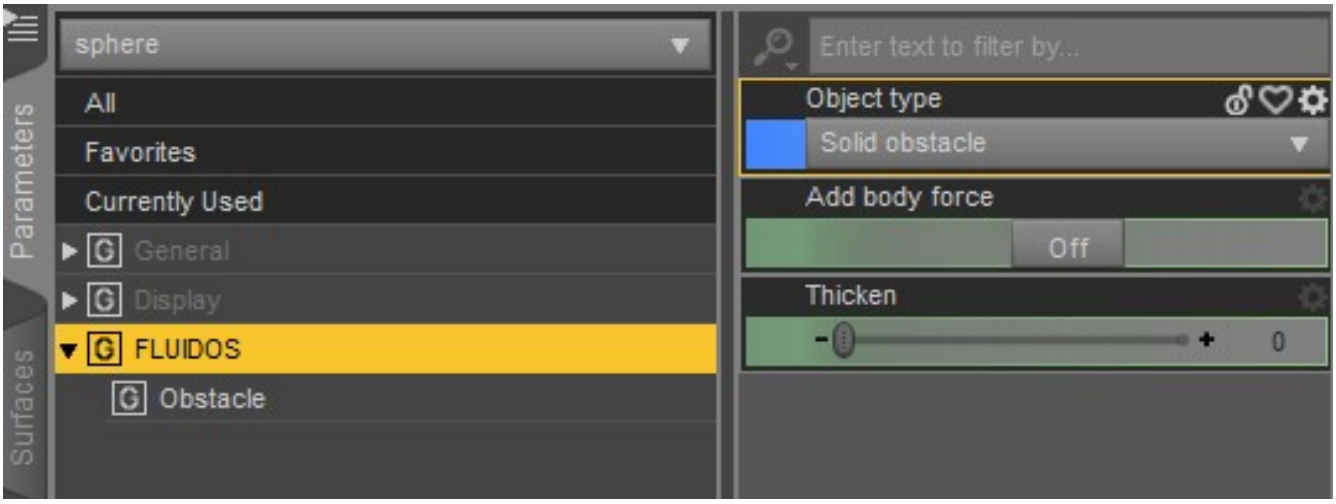
There are five types of objects:

- Obstacle (default)
- Fluid mass
- Viscosity control
- Force field
- Ignored




There are three groups of parameters depending on the *Object type* selected: *Fluid*, *Obstacle* and *Viscosity control.*

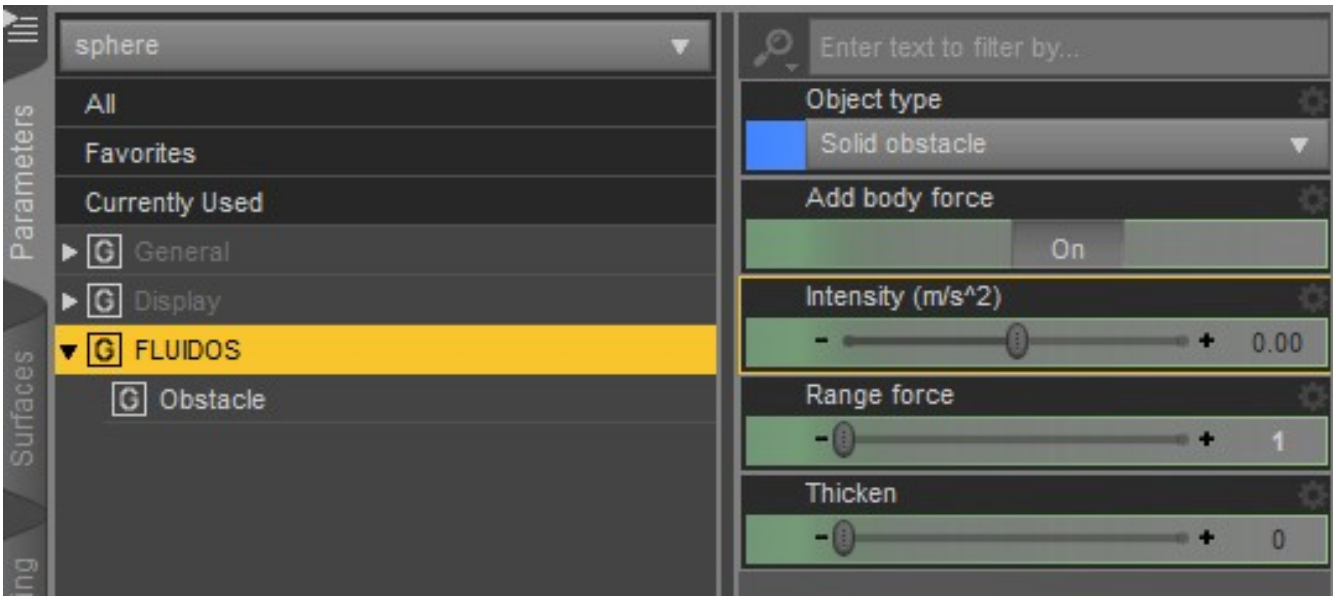
*Solid obstacle*




The *Obstacle* parameters are shown if *Solid obstacle* is selected in *Object type* property.

*Thicken*: this option adds invisible layers in the solid grid. This thickens the solid walls, so the fluids cannot spill out. Each extra layer is a cell size width. It is animatable.

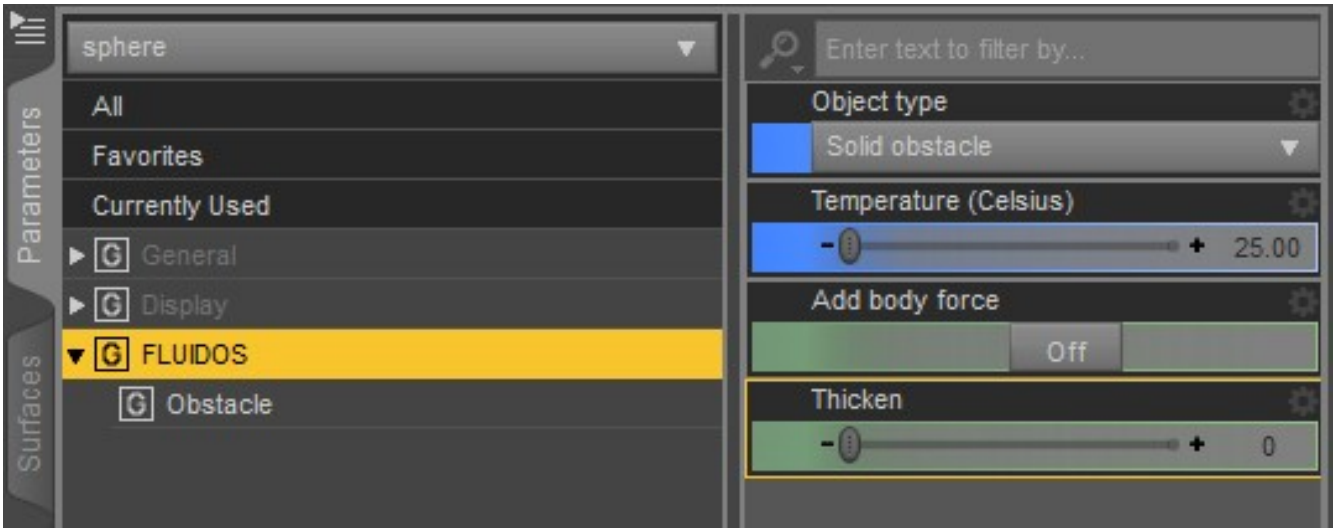
*Add body force*: It is a surface force linked to the solid obstacle. If this button is *ON*, additional properties are shown:




*Intensity*: this set the intensity of the force, it can be positive or negative. It is animatable. Attraction forces (positive ones) between 5-30 is better for most common simulations.

*Range force* slider allows to “thicken” the extent of this force to one or two cells away the solid.

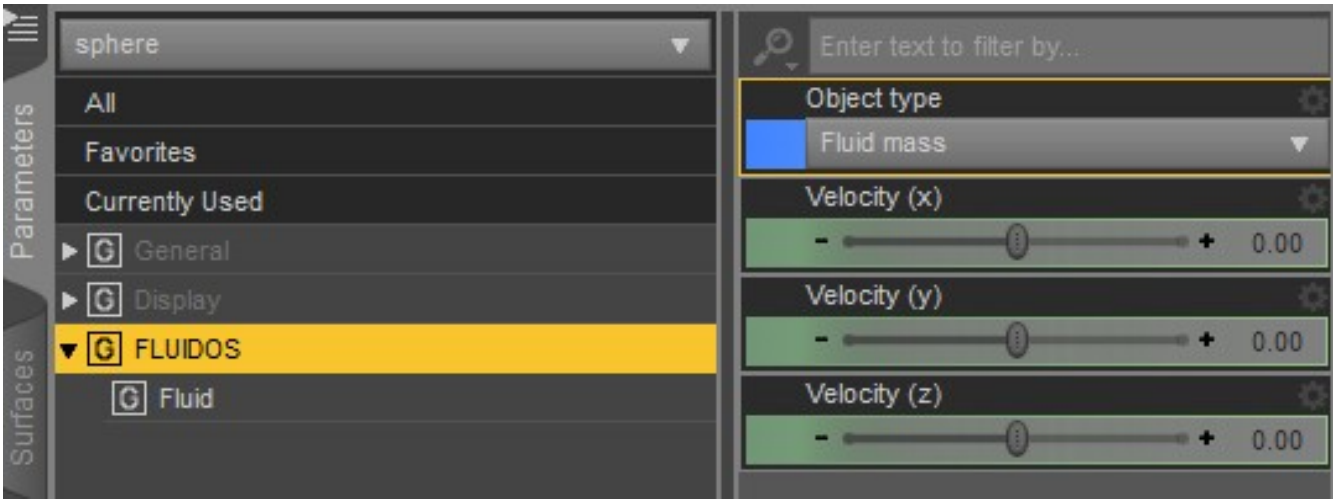
If *Smoke* simulation are selected in the *Domain*, an extra property will show, *Temperature*:




*Temperature*: this is the temperature in Celsius degrees of the solid. An obstacle will interchange heat to surrounding particles.

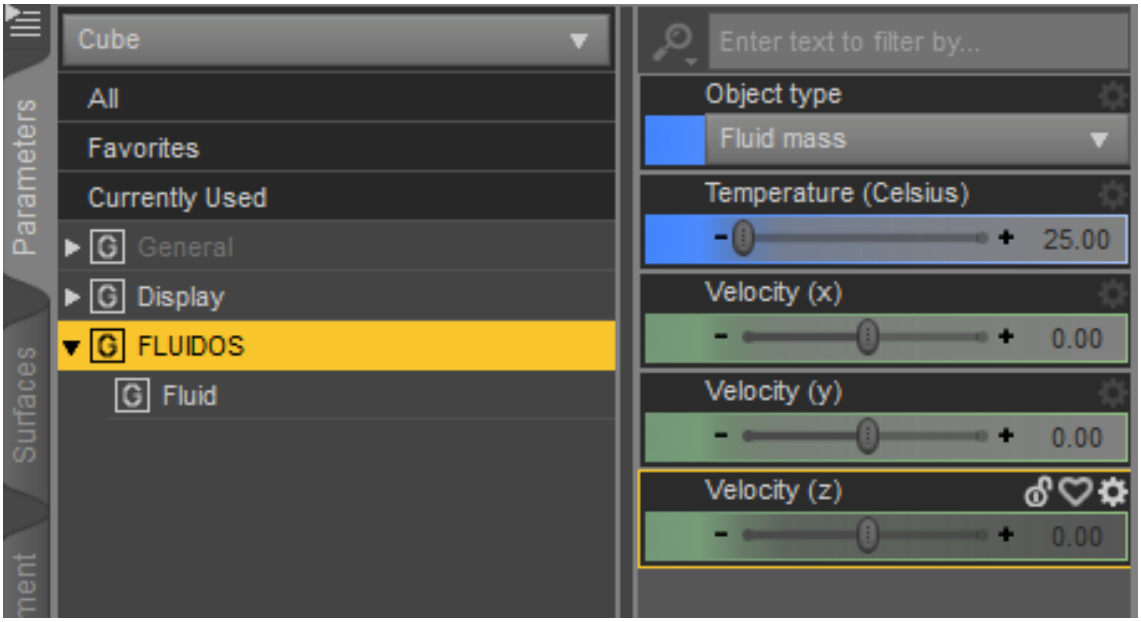
If two or more *Solid obstacle* objects overlap in some region, the *last* one in the hierarchy takes precedence.

*Fluid mass*



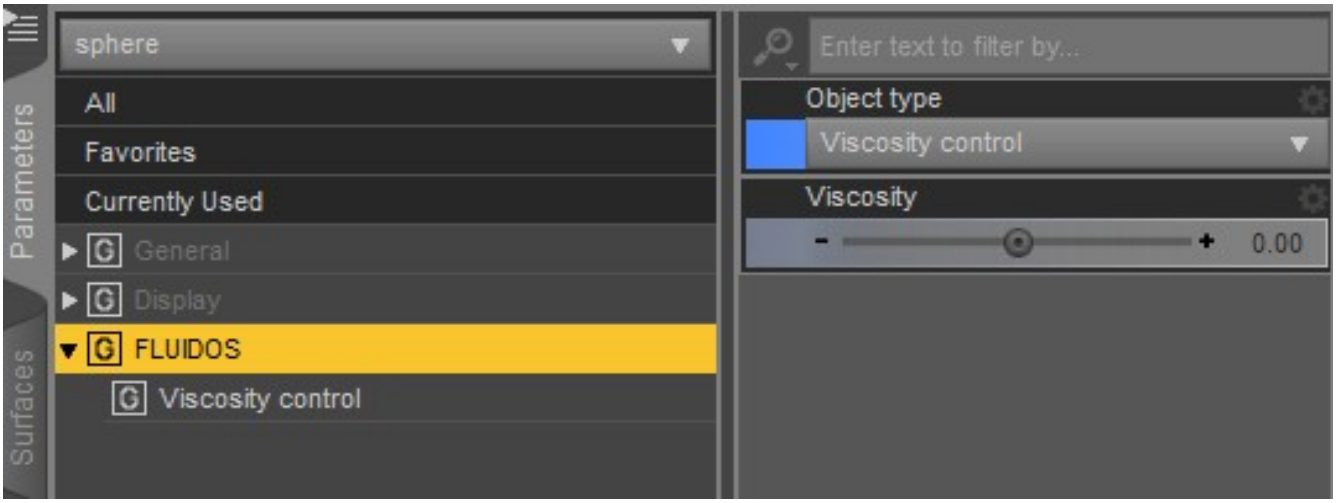

If *Fluid mass* is selected in *Object type* property, the object will be considered as a fluid. The Fluid parameters are shown. With them you can assign an initial velocity to the fluid. The initial velocity of the fluid mass is in cm/s and is controlled with *Velocity (x)*, *Velocity (y)* and *Velocity (z).*

In case of *Smoke* simulation selected in the Domain, a new property will show: *Temperature*. Lite edition do not use *Smoke concentration. Temperature* have the same application as the equivalent *Source/Sink* property.




If two or more *Fluid mass* objects overlap in some region, the *first* one in the hierarchy takes precedence.

*Viscosity control*

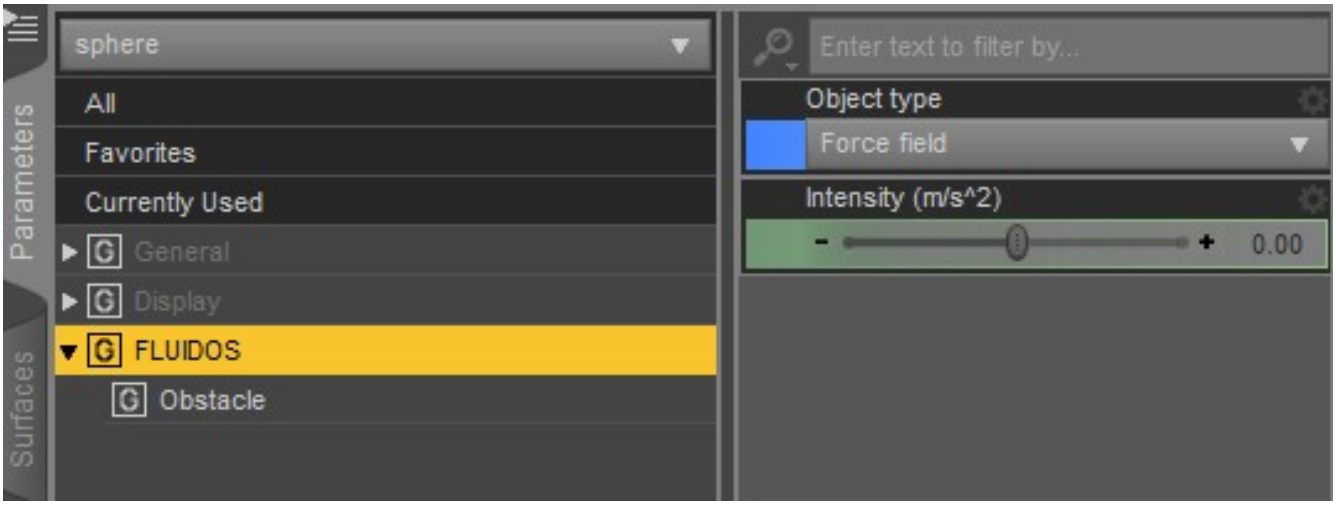



A viscosity control is an object with a Viscosity value that applies in all the space covered by the control.

The *Viscosity* values in a point in space are the sum of all the controls that overlap at this point, plus the general viscosity value (at Viscosity in Fluidos domain's properties). *Viscosity* parameter is animatable.

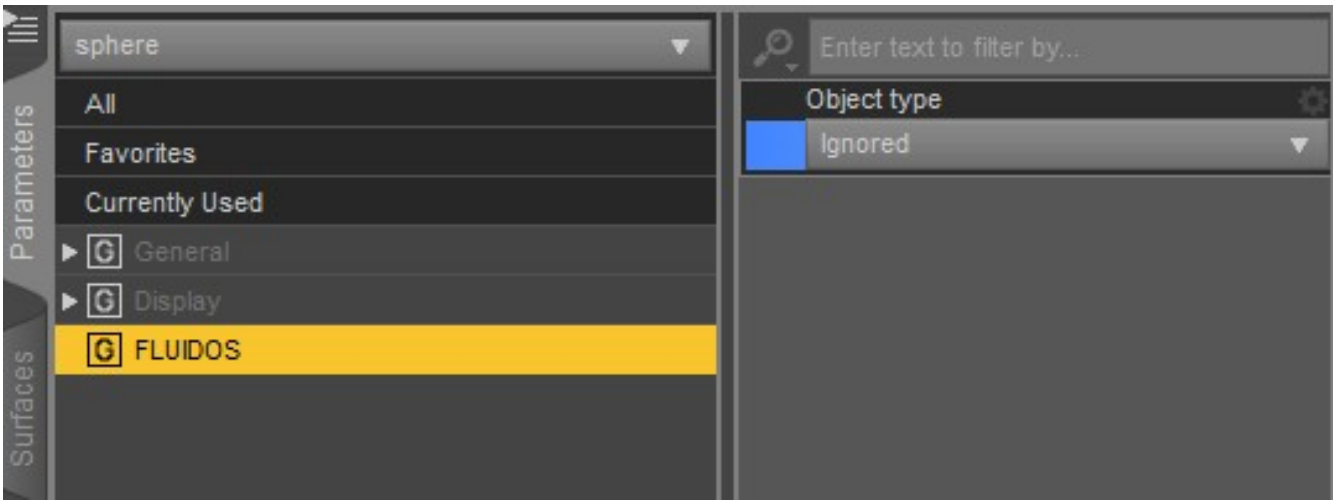
Negative values are allowed in the controls but the plugin will correct the final result to a point to zero viscosity if the sum of overlapped controls is negative.

*Force field*




*Force field*: this uses the geometry of the node to create a force field of the *Intensity* selected (animatable). This object doesn't act as an obstacle.

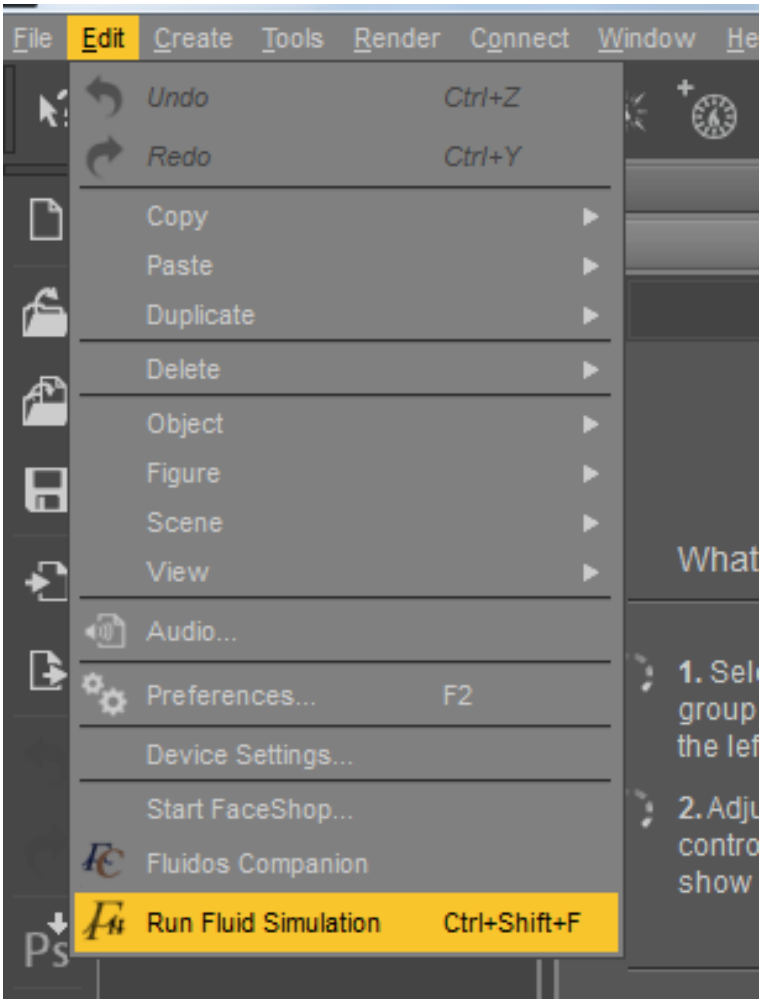
*Ignored*



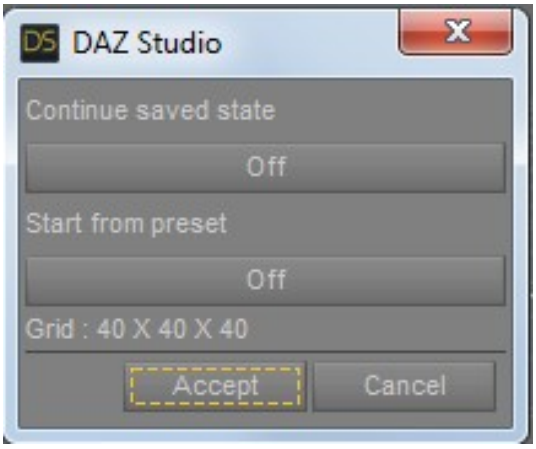

If *Ignored* is selected in *Object type* property, the object will be ignored by the simulator.

***Running a simulation***

To run a simulation, all the objects involved in it, must be parented to a Fluid Domain. Then select the Domain and go to the menu *Edit – Run Fluid simulation* (Ctrl+Shift+F is a shortcut). If no Domain is selected, the plugin will choose the existing one if there is only one, however, if there are more then one Domain in the scene the plugin will show a *dialog* for the user to choose the Domain.



The next dialog box then appears:




If there is a previous simulation baked in the selected folder, it can be resumed or extended for more frames by setting *ON* the *Continue saved state* button.

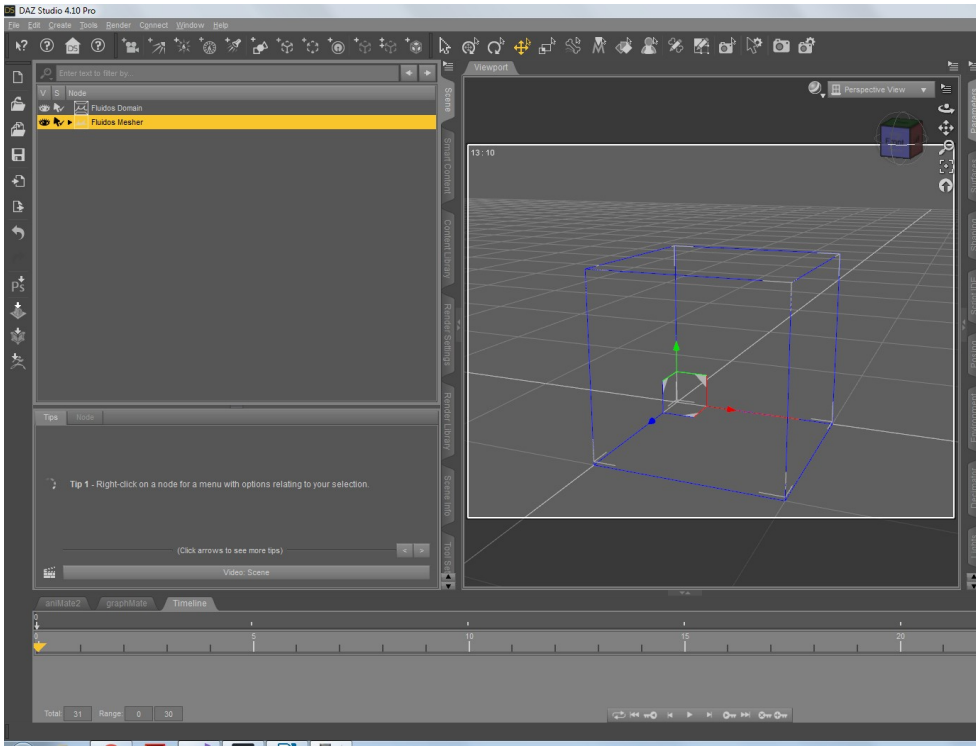
*Start from preset* will start a simulation from a preset, that is, from a simulation's *state* file. This file is simply an autosave.state file called by other name or moved to other folder. The difference between this option and the *Continue saved state* one is that the first one will start the simulation from frame 0, and the second one will start from the last calculated frame (For more information see *Appendix B* at p. 40)

The box tells the Grid size. The larger the grid, the slower the simulation and the bigger the fluids geometries.

Push *Accept* to run the simulation. The simulation can be canceled by clicking on the *Cancel* button of the Progress bar (or pushing the Escape key). As the engine is unresponsive when is calculating a frame, sometimes seems to be locked (in complex simulations), be patient.

**Visualizing the results of the simulation**

The Fluidos Mesher is the object that shows the results of the simulation:



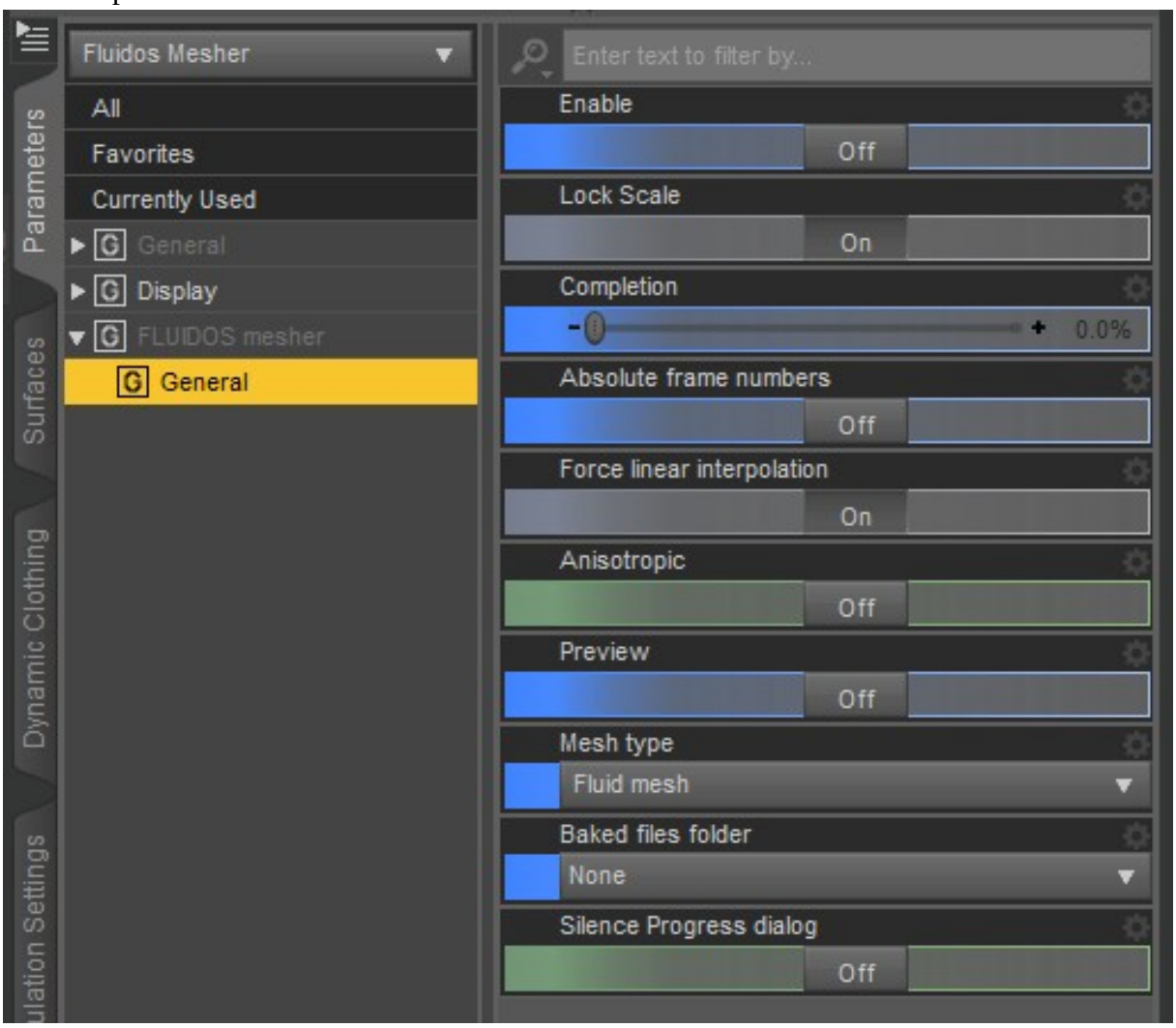

The user must insert the *Mesher* in exactly the same position as the Fluidos Domain. Its parameters are in three possible groups:

- General
- Particles
- Fire color temperature

There is no other limit for the number of *Meshers* in scene than the memory system.

**Note**: LITE edition can read Volumetric fire simulations, but cannot create or run them.

*General* parameters




*Enable*: Set *ON* this button to enable the modifier. If the simulation is extended (new frames calculated) and the Mesher is enabled, it must be disabled and re-enabled to update.

*Lock Scale*: set *ON* to prevent accidentally resizing the Mesher. The Mesher must not be resized if the user wants to see the results of the simulation matching the original set up of the scene. But, in a different case, setting this parameter *OFF* allows the access to scaling parameters

*Completion*: controls the animation of the fluid simulation. Set it to 0% at the beginning of your animation, and at 100% at the end. This is the only Mesher parameter that is animatable.

* *Absolute frame numbers*: When this property is enabled, the result displayed at each frame will coincide exactly with the result of the simulation at each frame. E. g., at frame 3 the result displayed will be the result of the simulation at frame 3, it cannot be displaced to another frame (unless the *Absolute frame numbers* option is Off, so the *Mesher* uses the Completion property instead).

*Force linear interpolation*: The default interpolation of the key frames in Daz Studio is nonlinear, but to get a correct synchronization of the simulation in the Mesher with the scene, a linear interpolation is needed. Setting ON this button forces a linear interpolation of the key frames of the Mesher (another option is to use another plugin like GraphMate).

*Anisotropic*: Set *ON* in order for the Mesher to read the anisotropic files if they exist (the isotropic and anisotropic files are saved in the baked files/ subdirectory). By default, the Mesher reads only the isotropic files. Of course, the isotropic or anisotropic files must be created previously by running a simulation.

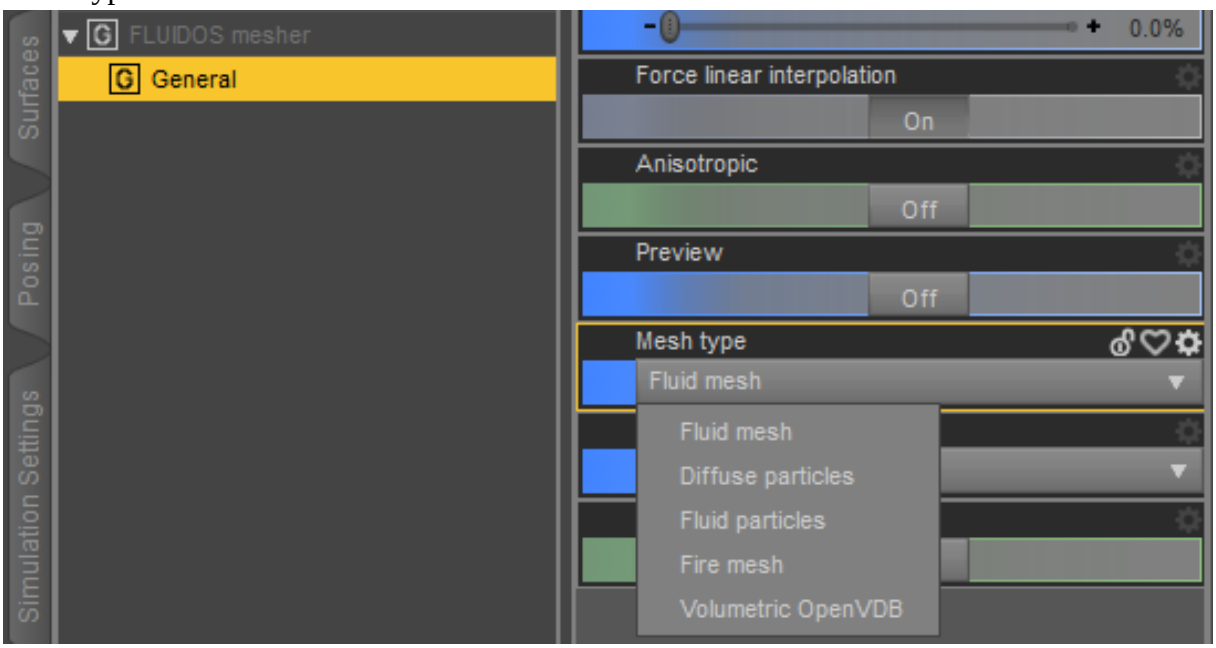
*Preview*: by default is set *ON*, so the Mesher shows a preview, (a low resolution version of the fluid simulation.) It is recommended to avoid as much as possible the deactivation of this option except during final rendering because extreme simulations can be very slow to get rendered in the viewport of Daz Studio. If *Anisotropic* is set *ON*, *Preview* is hidden because there is no preview generated for the anisotropic reconstruction.

*Baked files folder*: use this to select a folder of fluid simulation.

*Silence Progress dialog*: This property, if enabled, will hide the progress dialog that appears when a large polycount mesh or a high number of particles is reproducing.

> Normaly, when the user drags the timeline slider, the icon of the mouse pointer takes the form of a closed hand that changes to an open hand when the left button of the mouse is released. When the progress dialog is present, if the user drags the timeline slider, the releasing of the left button will not “open” the hand, and the mouse movements will cause more dragging of the slider. By clicking the left button again on the slider the hand icon opens again and the dragging ceases. The silencing of the Progress dialog avoids this locking of the dragging.

Mesh type

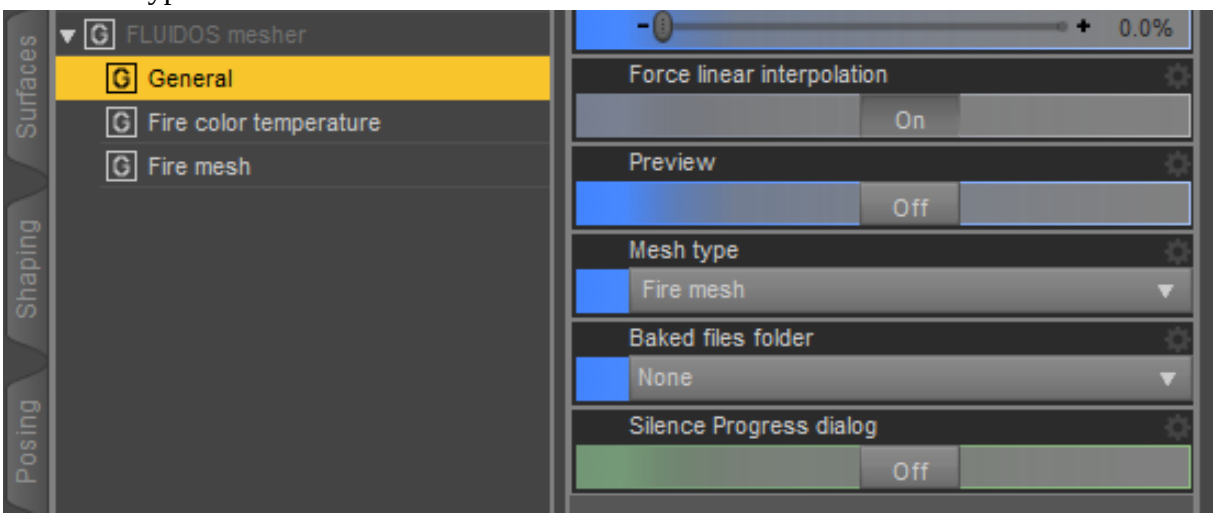



*Mesh Type*

There are four types*: Fluid*, *Fluid particles*, *Fire*, *Diffuse particles* and *Volumetric OpenVDB*. Each *Mesher* will show only a type of mesh. And to show a type of mesh, there have to be the correspondent files in the *Baked files folder.* Fluid mesh reads the .ply files with *anisotropic* or *preview* prefix or with no prefix. *Diffuse particles* mesh needs the .ply files with the *diffuse* prefix. *Fire mesh* needs the .vdb files with the *fluid* prefix. And the *Fluid particles* need the .ply files with the *mkparticles* prefix and optionally with the *mktemperatures* one. The fire simulations (Fluidos II complete edition only) will create the *fluid* .vdb files and the *mktemperatures* .ply files. Enabling the *Diffuse particles* in the domain will produce the *diffuse* files.

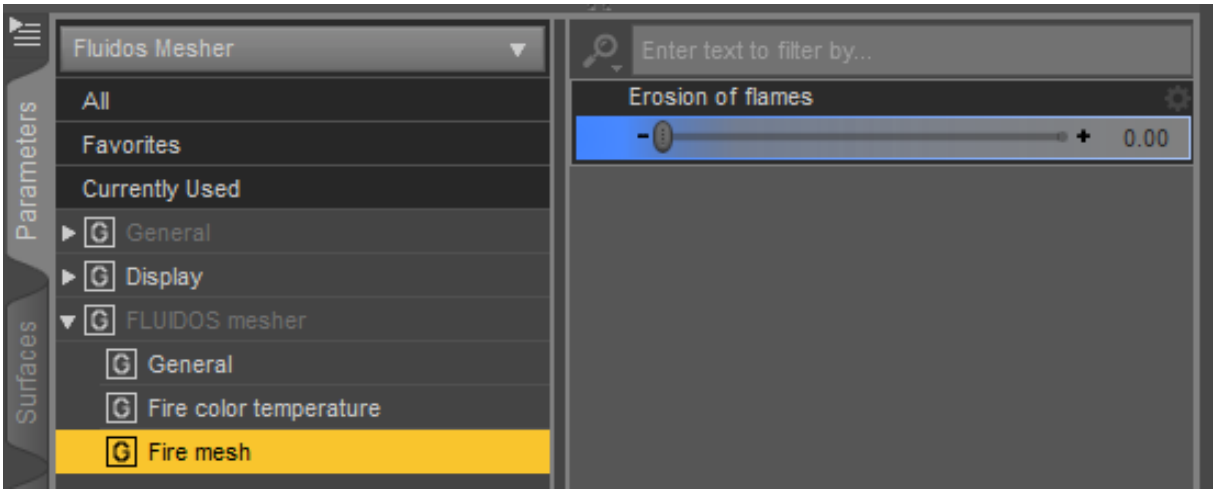
*Fluid Mesh*: this is a “solid”, closed mesh, it was the default in the *old Fluidos* plugin.

*Fire mesh* type



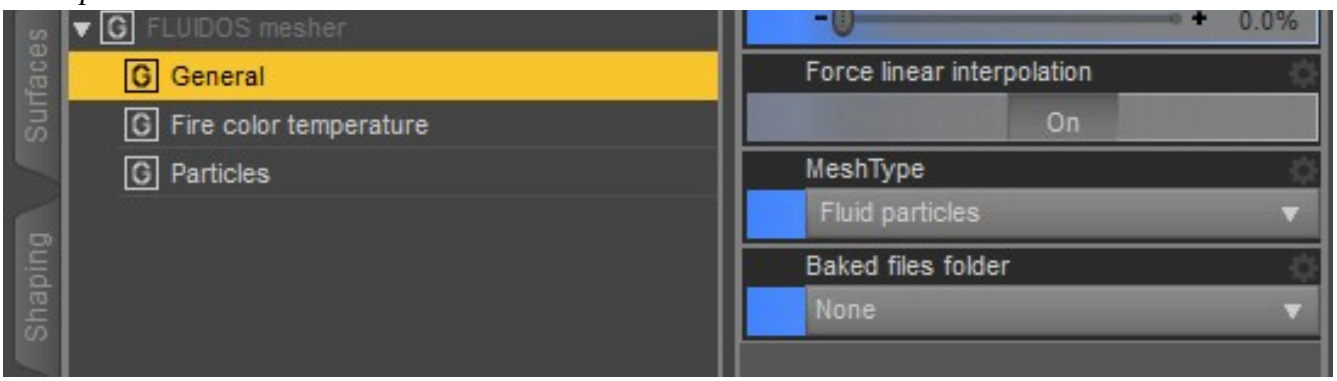

*Fire Mesh*: this is a "solid", closed mesh also, but it has associated temperatures data used by a *Fire shader,* thus the mesh regions have colors according to their temperatures.

*Fire mesh* group




*Erosion of flames*: The effect of this property is equivalent to the one of the *erosion* filter of the *Fluidos Domain*. The volume of the mesh is reduced by eroding it, that is, shrinking uniformly the mesh from the boundaries. It's animatable.

*Fluid particles*

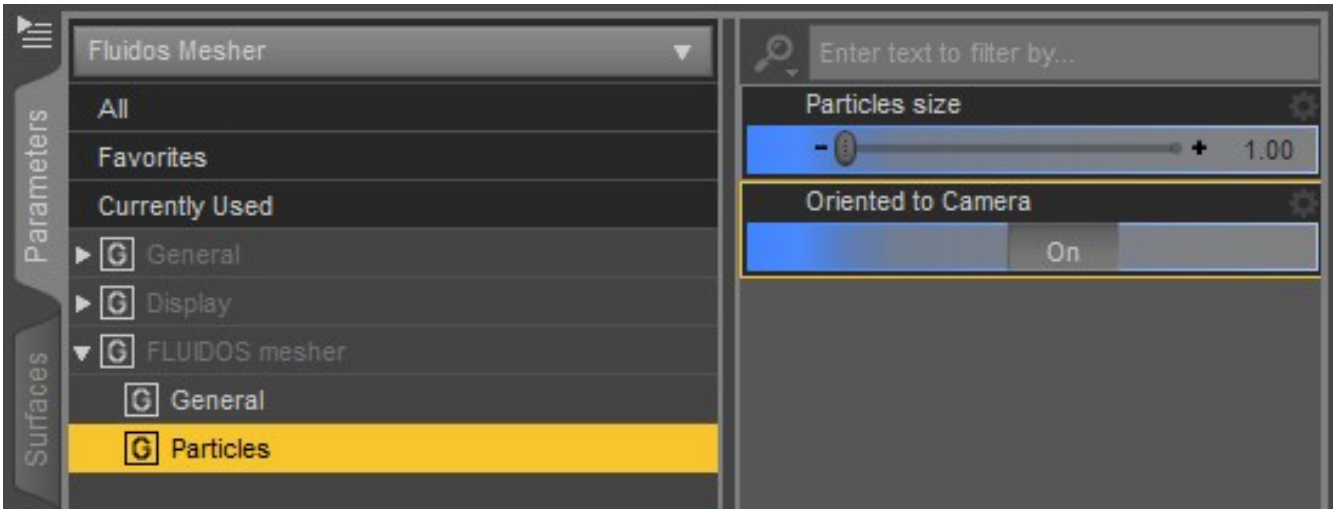



*Fluid particles mesh* will display triangles representing each fluid particle in the simulation. If *Volumetric fire* is selected in the domain, the temperatures for a *Fire particles* shader will be available, thus each particle will have a color according to its temperature. The *Preview* option is new since V2.2 By enabling it, the mesher will show only one of sixteen particles.

*Diffuse particles*:

The *Diffuse particles*, as in *Fluid particles mesh*, will show triangles one for each diffuse particle. The *Preview* option is new since V2.2 By enabling it, the mesher will show only one of sixteen particles.

*Particles* group properties

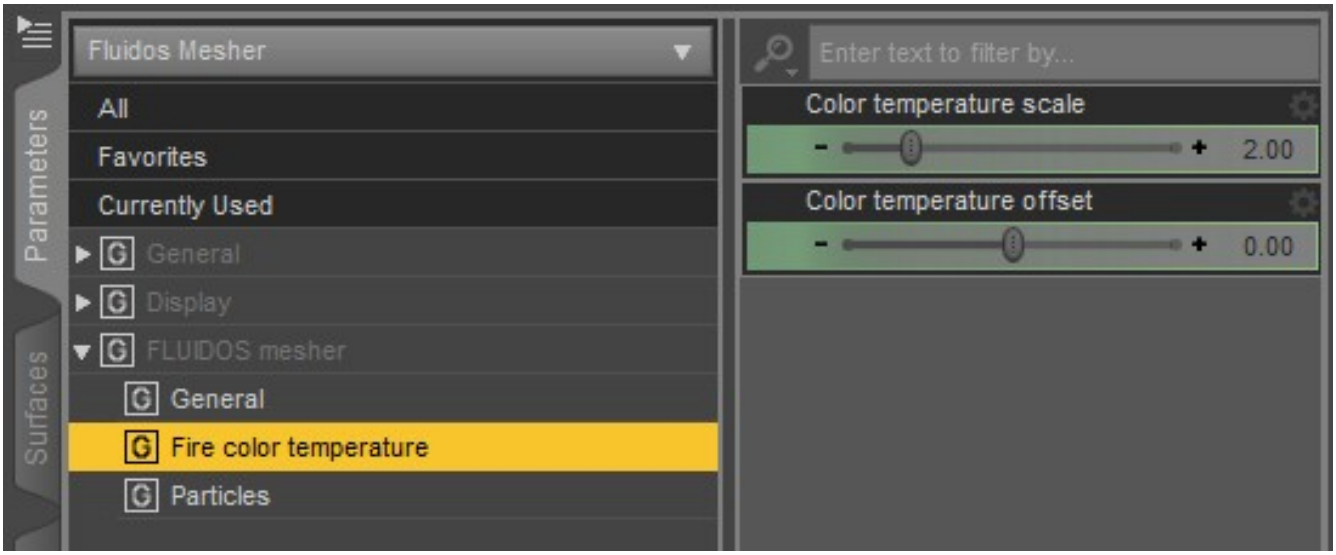


These properties apply to *Diffuse particles* and *Fluid particles.*

*Particles size*: the diffuse particles can be resized (the default value is 1.0), this is an animatable parameter. The particles are triangles that can be used as billboards.

*Oriented to camera*: this orients the particles to face the opposite direction in which the camera is oriented.

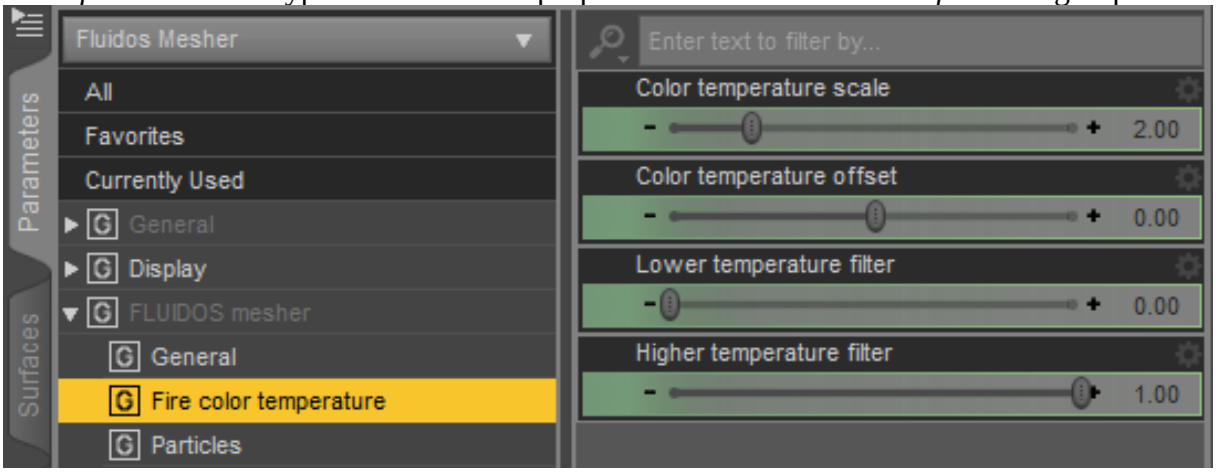
*Fire color temperature* properties group




These properties apply to *Fire mesh* and *Fluid mesh*.

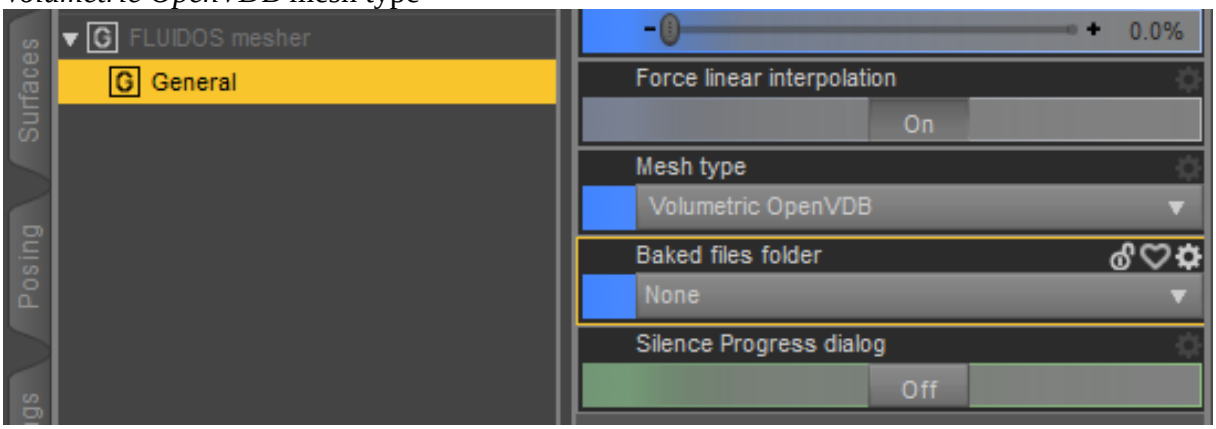
*Color temperature scale* and the *Color temperature offset.* The offset increase or decrease the minimum temperature among particles; the scale, increase or decrease the range of temperatures.
If you feel the fire is too reddish, increase the offset or the scale value; if it's too much whitish, decrease the offset or the scale.

*Fluid particles* mesh type have two more properties in the *Fire color temperature* group.

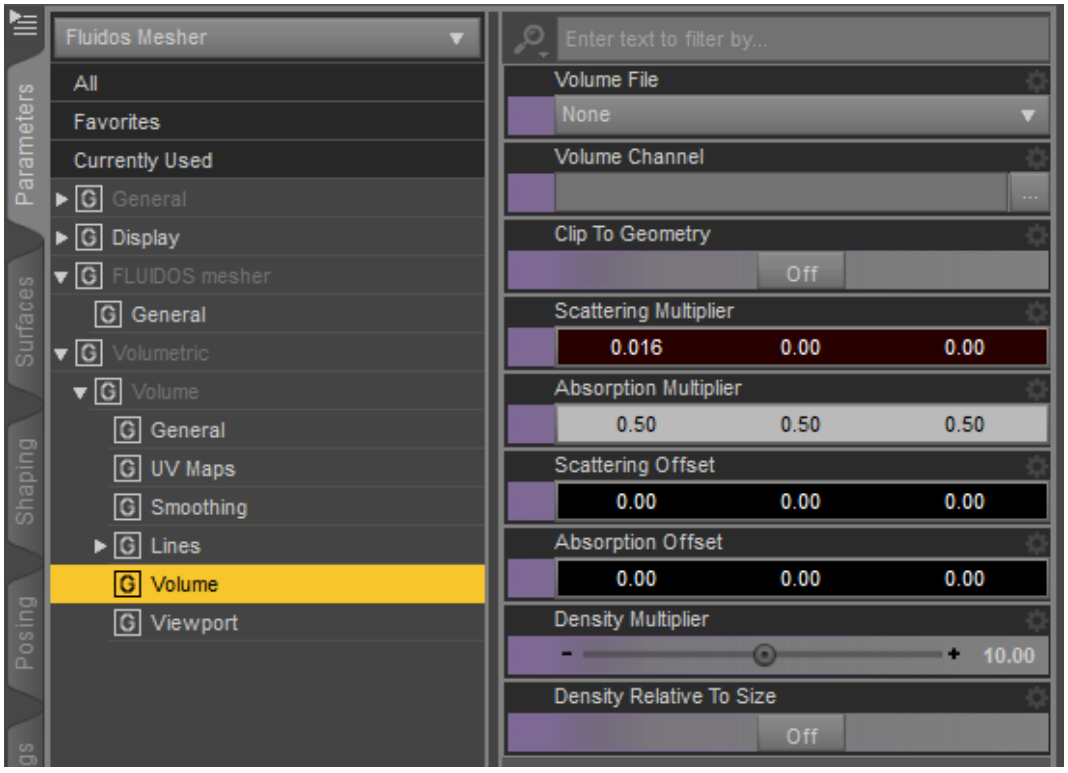


*Lower temperature filter* and the *Higher temperature filter*: These filters cut out the particles which temperature is outside the limits marked by them. The scale is relative to the maximum and minimum possible temperatures in fire simulations after the applying the *Color temperature scale*. These parameters are animatable.

*Volumetric OpenVDB* mesh type




This type allows the rendering of volumetric VDB sets. The user should apply one of the included *Fluidos II Smoke VDB* Iray shaders or any other volumetric VDB shader. The properties of the Mesher's *surface* are automatically copied to the properties of the Mesher as in the following image:




This way, the color and numeric *Surface* properties become animatable. The user can modify the properties of the Mesher or, better still, the properties of the S*urface* of the Mesher.
The user should not change the *Volume file* property, as the Mesher automatically will set the file corresponding to the current frame.

Preferably, change the *Mesh type* to *Volumetric OpenVDB* when the Mesher is disabled (*Enable* property is *Off*). Apply the shader before enabling the Mesher. Enable or re-enable the Mesher after changing a shader to re-synchronize.

This type of mesh requires the vdb files with prefix "*fogfluid*". This files are created in any type simulation (fluid, smoke) when the *Enable saving Fog-type VDB* property of the *Domain* is enabled.

References**:**


1. Guy L. Ryan. 2016. PIC/FLIP Fluid Simulation. http://rlguy.com/gridfluidsim/

2. Guy L. Ryan. 2017. FLIPViscosity3D https://github.com/rlguy/FLIPViscosity3D

3. Batty C and Bridson R. 2008. Accurate Viscous Free Surfaces for Buckling, Coiling and Rotating Liquids. In Proceedings of the 2008 ACM/Eurographics Symposium on Computer Animation.

4. Batty C. 2008. VariationalViscosity3D https://github.com/christopherbatty/VariationalViscosity3D

5. Bridson, R., & Müller-Fischer, M. 2007. Fluid simulation: SIGGRAPH 2007 course notes. In ACM SIGGRAPH 2007 courses.
6. https://www.researchgate.net/publication/234801901_Fluid_simulation_SIGGRAPH_2007_course_notes_Video_files_associated_with_this_course_are_available_from_the_citation_page

7. Bridson, R. (2016). Fluid simulation for computer graphics. 2nd ed. CRC Press.

8. Fedkiw, R., Stam, J., & Jensen, H. W. (2001, August). Visual simulation of smoke. In Proceedings of the 28th annual conference on Computer graphics and interactive techniques (pp. 15-22). ACM.

9. Feldman, B. E., O'brien, J. F., O'Brien, J. F., & Arikan, O. (2003, July). Animating suspended particle explosions. In ACM Transactions on Graphics (TOG) (Vol. 22, No. 3, pp. 708-715). ACM.

10. Kim, D. (2017). Fluid Engine Development. AK Peters/CRC Press.

11. K. Museth, “VDB: High-Resolution Sparse Volumes with Dynamic Topology”, ACM Transactions on Graphics 32(3), 2013. Presented at SIGGRAPH 2013.

Acknowledgements

I wish to thank *Imago* from Daz Studio forums for his help with beta testing and his very valuable suggestions.

Licences:

About the engine simulation fluidsiIIm.dll and libfluidsimII.dynlib:

The engine simulation contains a modified version of **GridFluidSim3d** licensed under zlib license, by Ryan L. Guy. The engine uses some modified codes of FLIPViscosity3D licensed under MIT license as:

Copyright (c) 2017, Ryan L. Guy
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so,
subject to the following conditions:
The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

**OpenVDB 9.1.0**

Copyright (c) 2012-2022 DreamWorks Animation LLC

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at https://mozilla.org/MPL/2.0/.

https://github.com/AcademySoftwareFoundation/openvdb

**IlmBase**

Modified BSD License:
Copyright (c) 2006-2019 OpenEXR a Series of LF Projects, LLC. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

TBB (Threading Building Blocks)

Copyright (c) 2005-2019 Intel Corporation
Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License.
You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0
Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

## ZLIB

Copyright (C) 1995-2017 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

## Blosc

BSD License

For Blosc - A blocking, shuffling and lossless compression library

Copyright (C) 2009-2018 Francesc Alted <francesc@blosc.org>
Copyright (C) 2019-present Blosc Development team <blosc@blosc.org>

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name Francesc Alted nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## Boost

Boost Software License - Version 1.0 - August 17th, 2003

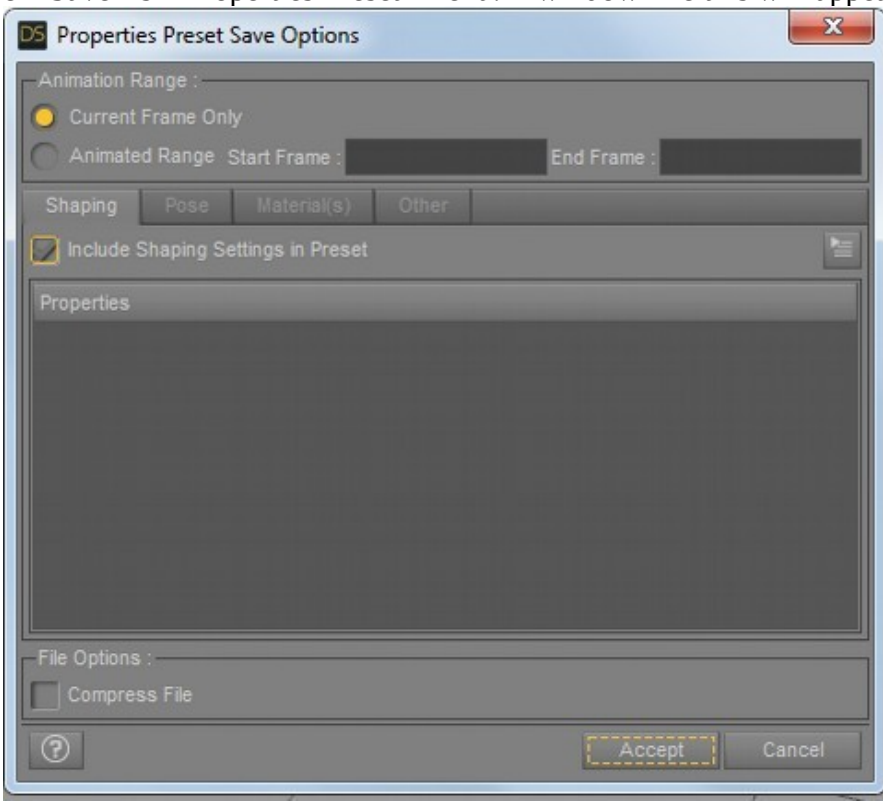
Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

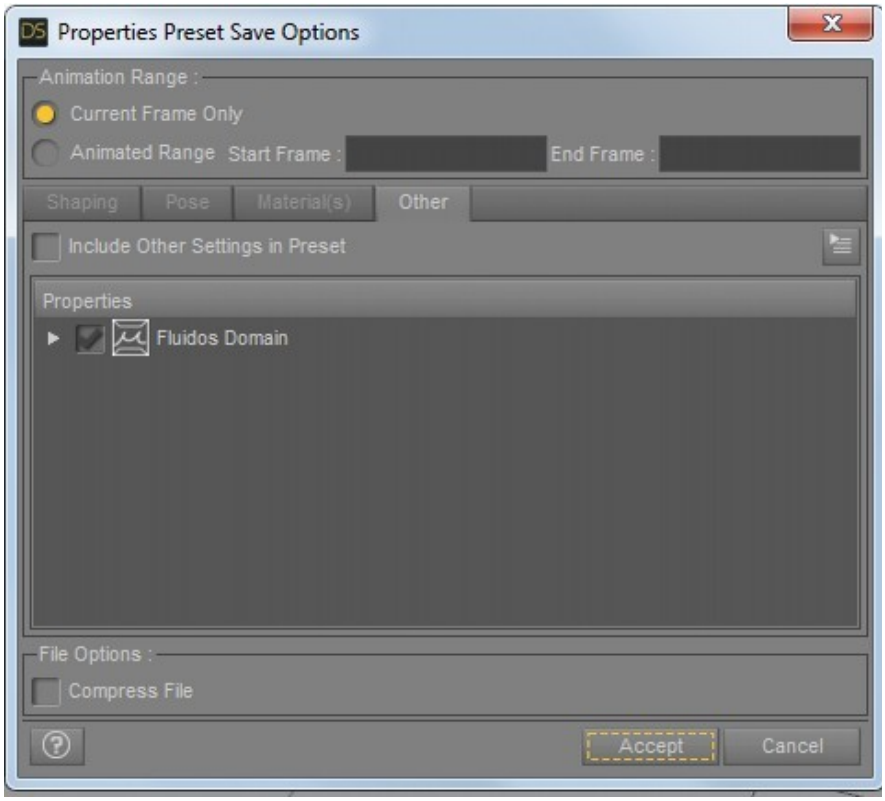
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

**APPENDIX A: How to create a Properties preset for the Fluidos Domain.**

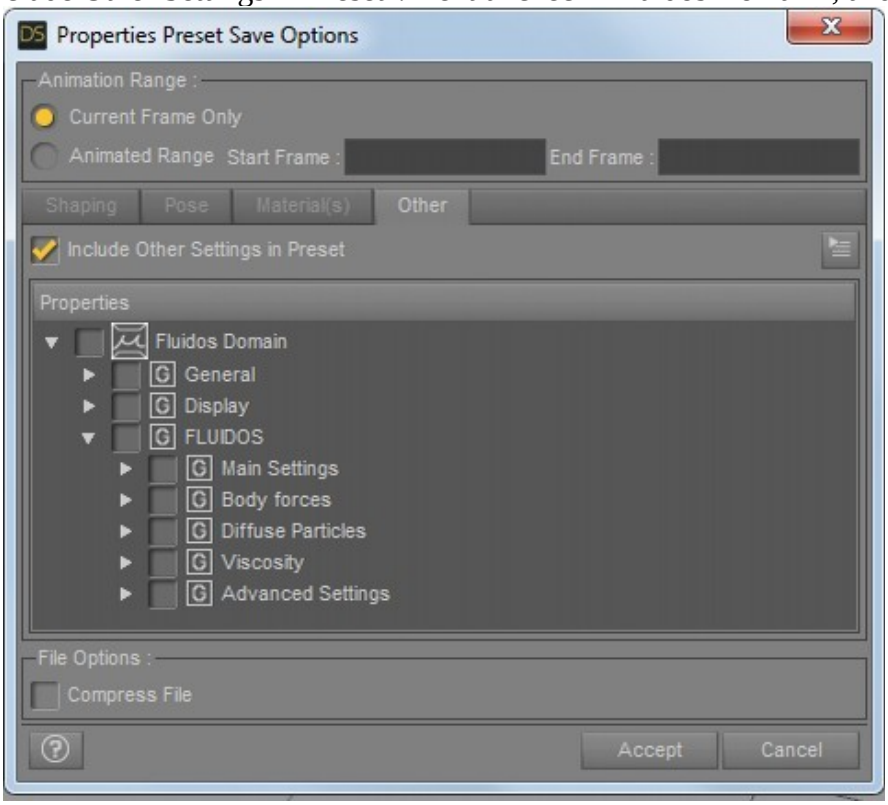
1. Select the Domain. Set the Baked Folder you always use.
2. Go to "File – Save As – Properties Preset" menu. A window like this will appear:



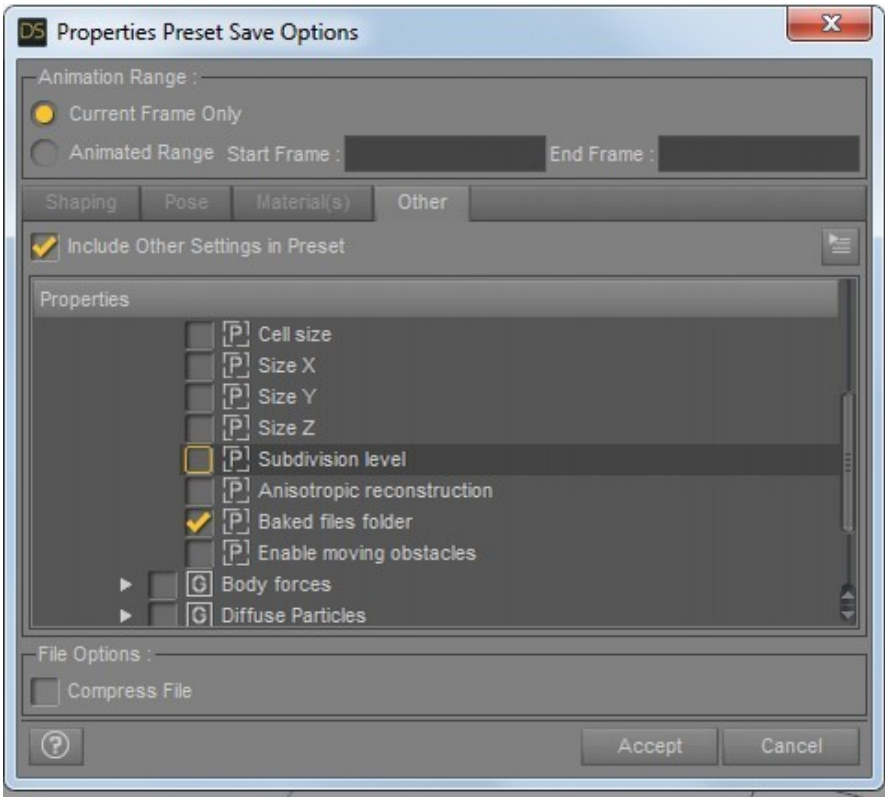

3. Uncheck "Include Shaping Setting in Presets". Do the same in Pose and Material(s) tabs.
4. Next, go to the Other tab.




5. Check "Include Other Settings in Preset". Next uncheck "Fluidos Domain", and expand.




6. Expand "Main Settings" and check "Baked files folder”



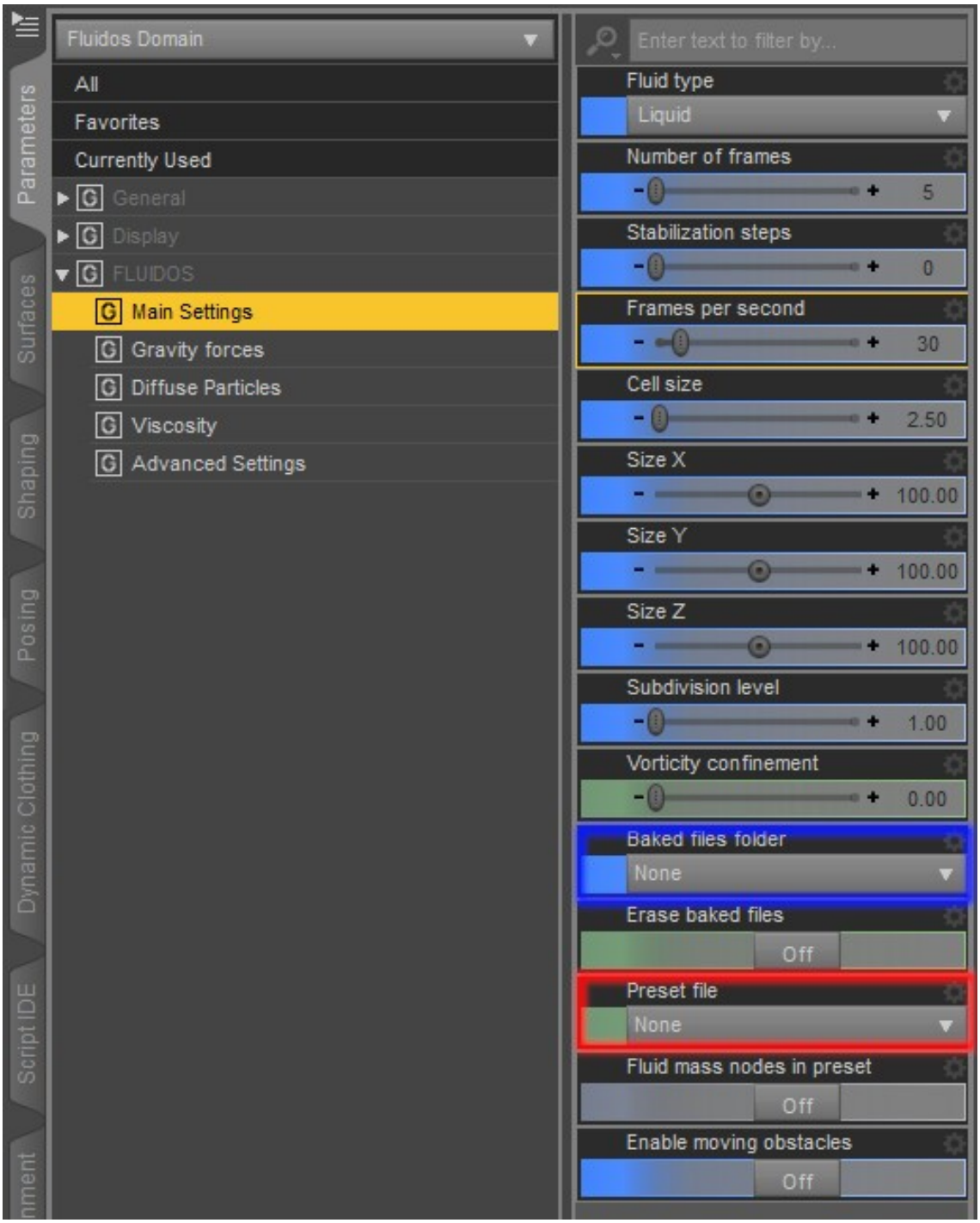

7. Click on Accept.
8. You have now a Properties preset only for the Baked folder. Applicable to any domain or mesh you wish.

You can choose other properties besides the Baked files folder. The same property preset can be used for Domains and Meshes. If you add an only-Domain-property to your preset, when you apply this to a Mesher, the only-Domain-property will be ignored.

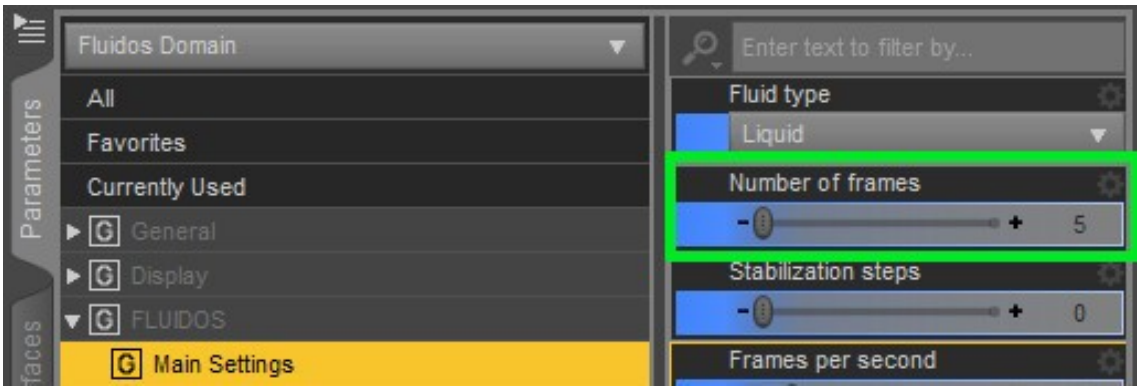
**APPENDIX B: How to use the preset examples included in Fluidos.**

To run the preset example scenes included in Fluidos package:

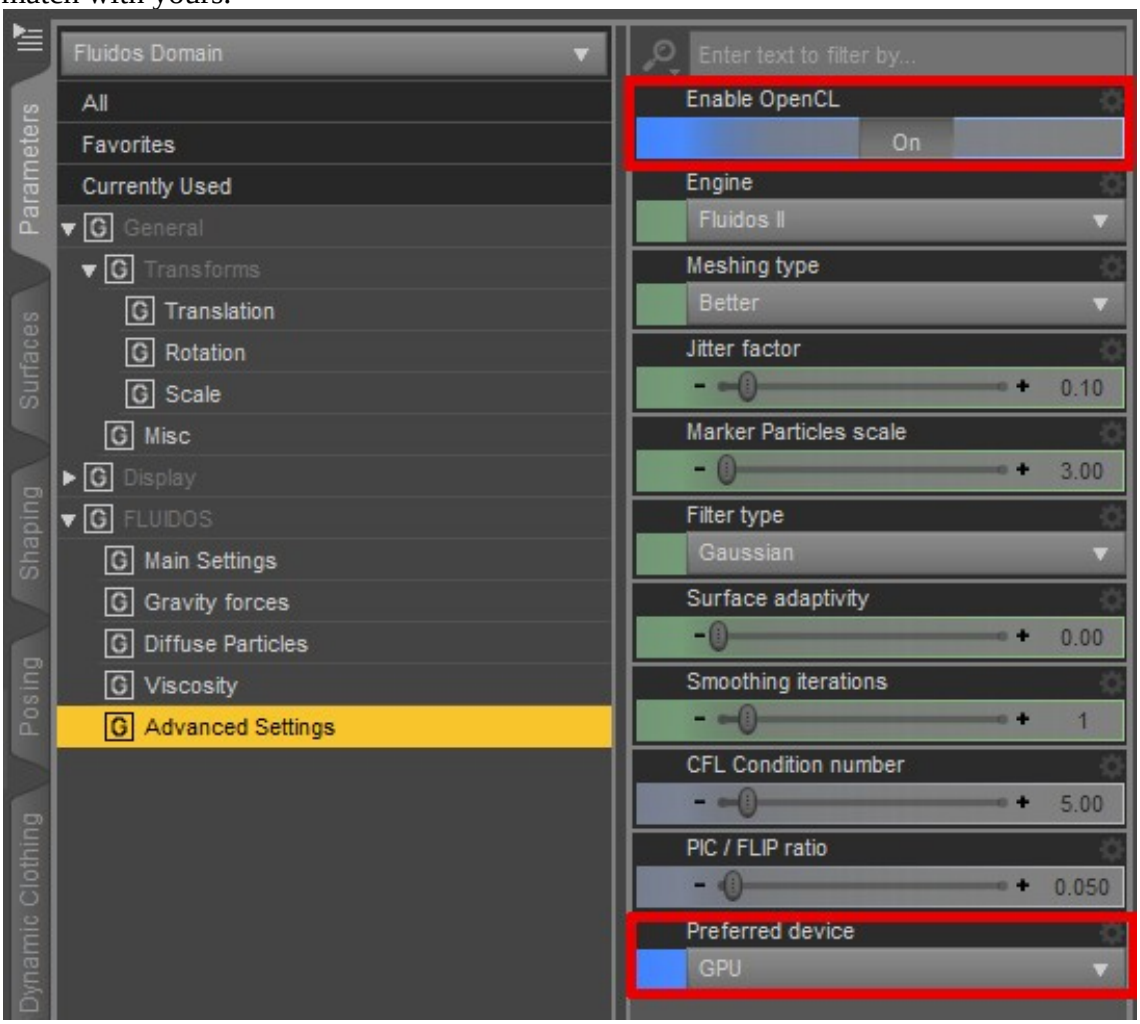
1. Load the preset example scene. You can add now obstacles, forces, sinks or sources. New fluid mass objects will be ignored if you run a preset.
2. Select the Fluidos Domain in scene.
3. In *Main Settings – Baked files folder,* browse to the preferred folder for simulation files. Do the same with the Fluidos Meshers in scene.



4. Then, browse for the *Preset File*. This is a .state file.




5. You can set up the number of frames you wish to simulate. The preset contains the data of a premade simulation, thus, in most cases it is unnecessary a large number of frames. Choosing 0 will produce the frame zero. Do not forget to set *Completion* to 100 % at the last frame you require, in all the Fluidos Meshers present in scene.
6. Check if the *Preferred device* and *Enable OpenCL* properties are the usual ones for your system, because the preset example scenes included in Fluidos have choices that could not match with yours.

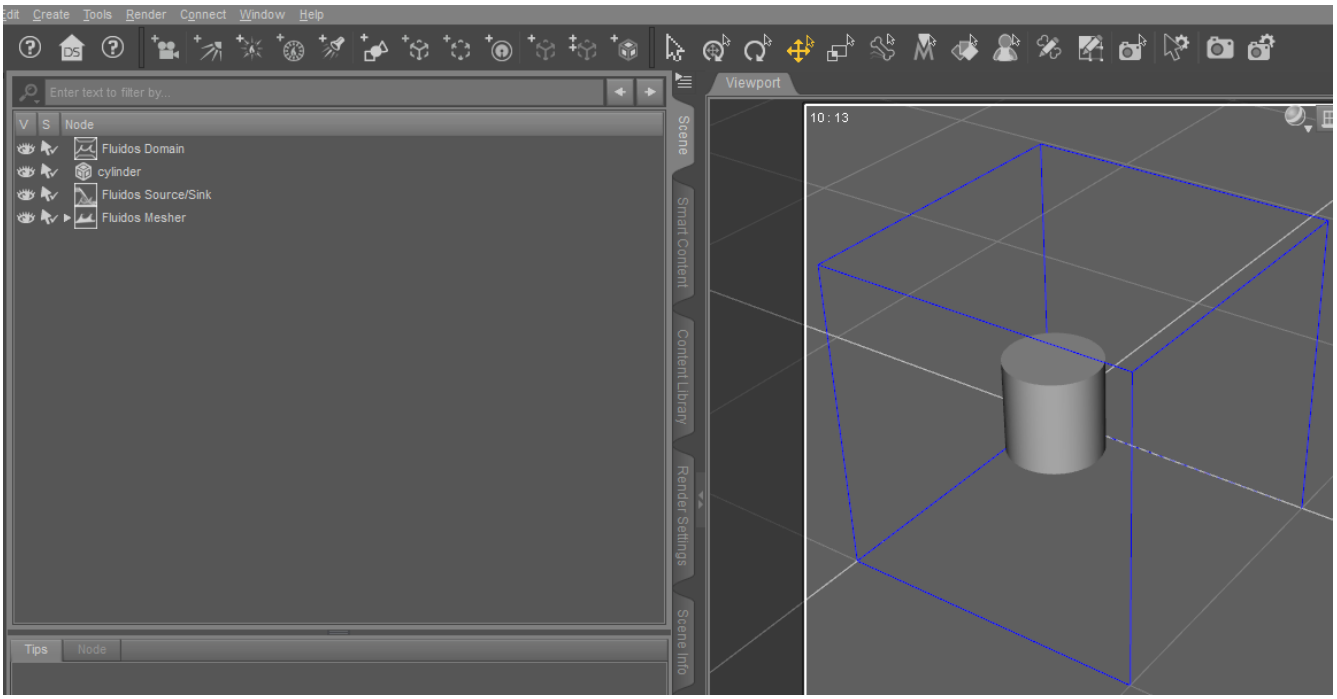


7. When running a simulation, set ON *Start for preset.*

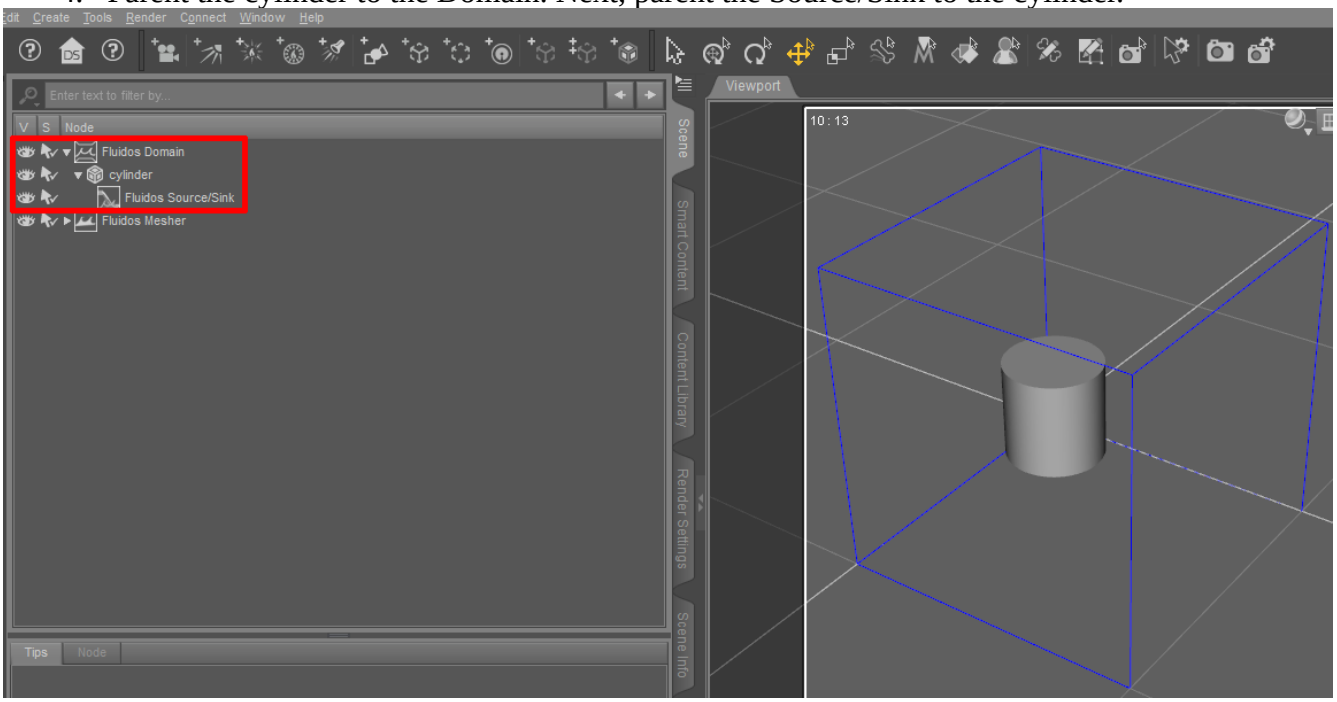
*Note:* When running a preset simulation, the user can change any original object node and any property. The only exceptions are: Cell Size, Size X, Size Y, Size Z of Fluidos Domain Main Settings. The user cannot add new fluid masses unless they set ON in *Allows initial fluid masses in preset* at Main Settings (*New* option since version 1.3), however the original fluid masses will also be activated (but they can be deactivated by setting their *Object type* property to *ignored*).

## APPENDIX C: Example of use of Any Node Geometric Type Source

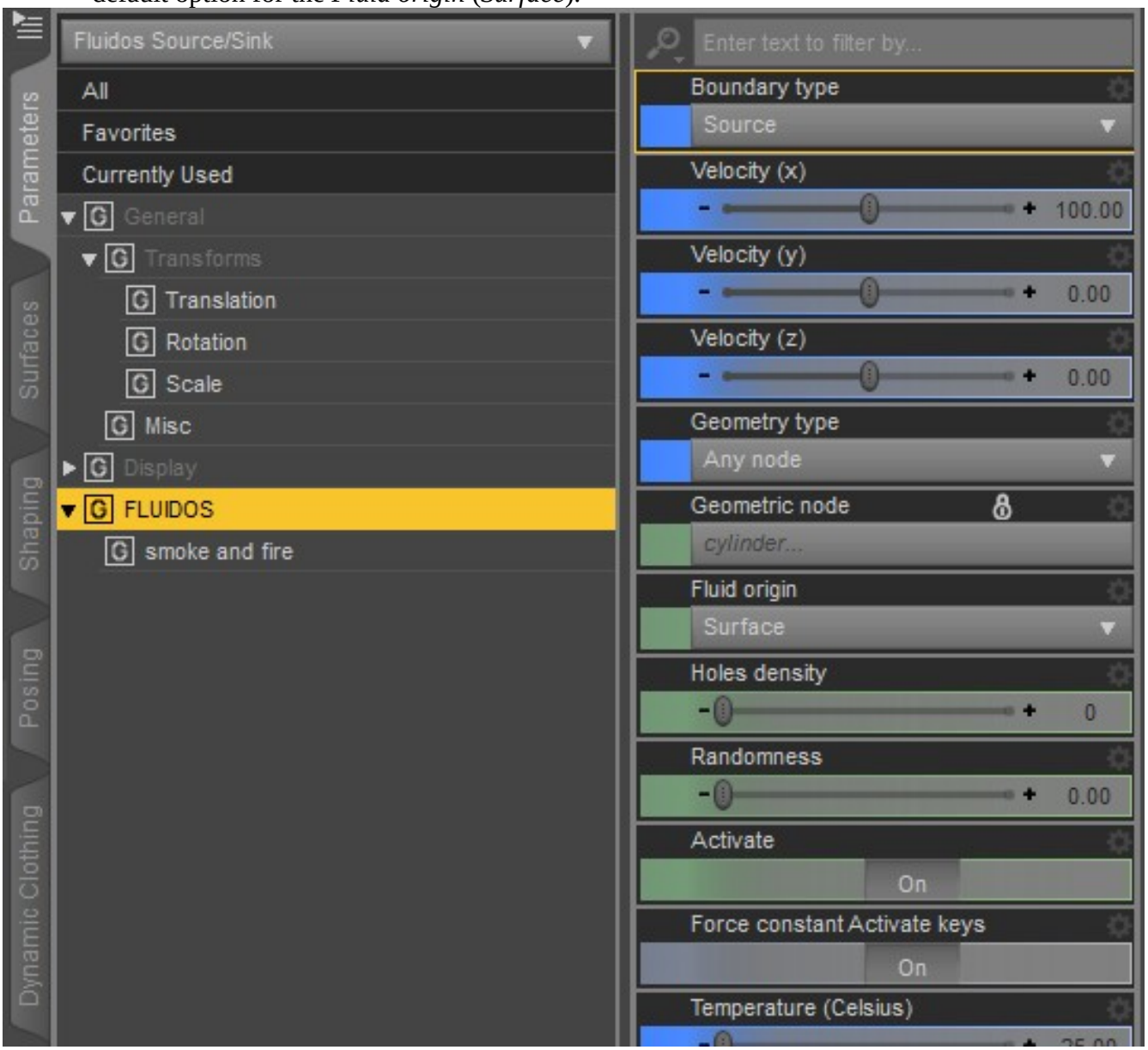
1. Add a Fluidos Domain and a Mesher to a new scene. Set up the Mesher to visualize from 0 to 30 frame as usual.
2. Create a cylinder primitive (0.3 m length and 0.3 m diameter). Move the cylinder inside the space of the Domain.
3. Create a Fluidos Source Sink.



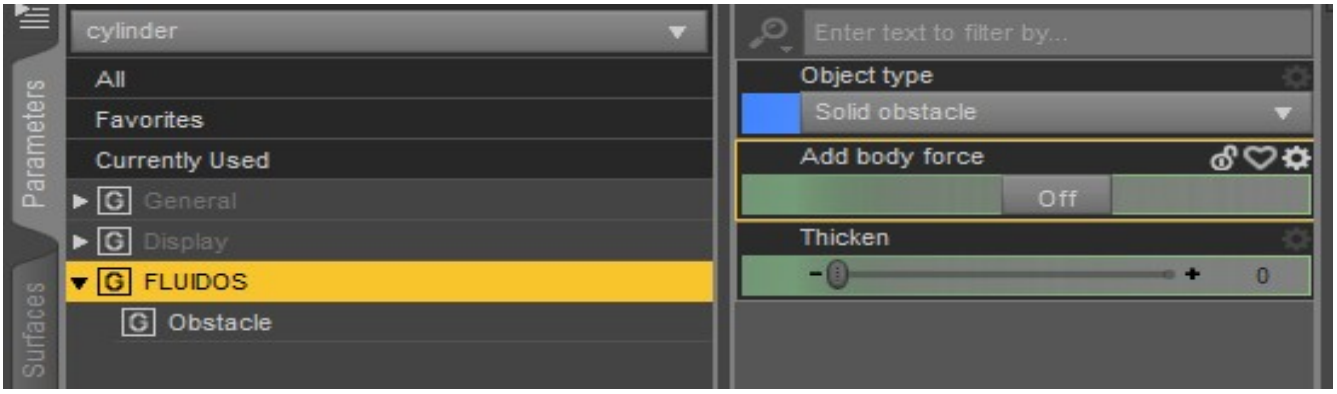

4. Parent the cylinder to the Domain. Next, parent the Source/Sink to the cylinder.



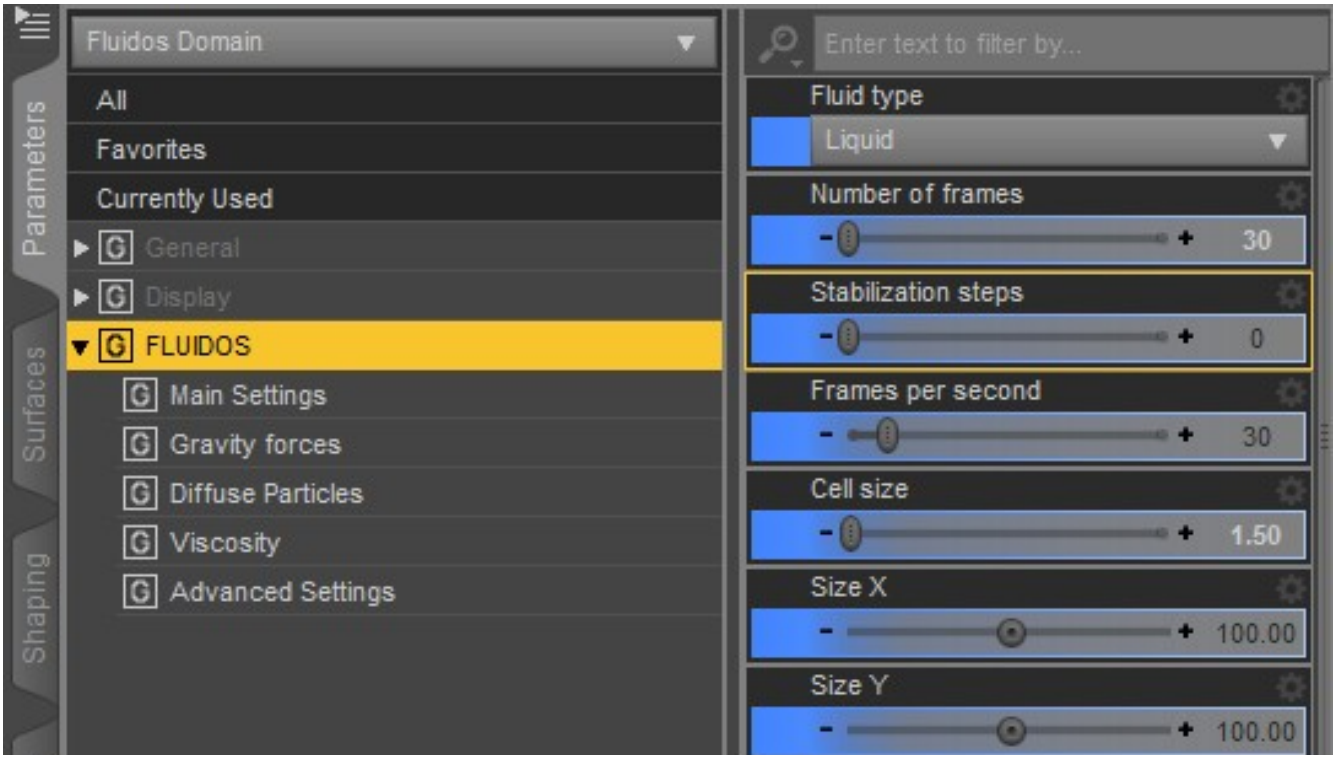
5. In Geometric type property of the Source/Sink, choose *Any node*. The *Geometric node* property box will show the name of the node that will act as a Source. Set the *Velocity (x)* to 0.0. Let the default option for the *Fluid origin* (*Surface*).



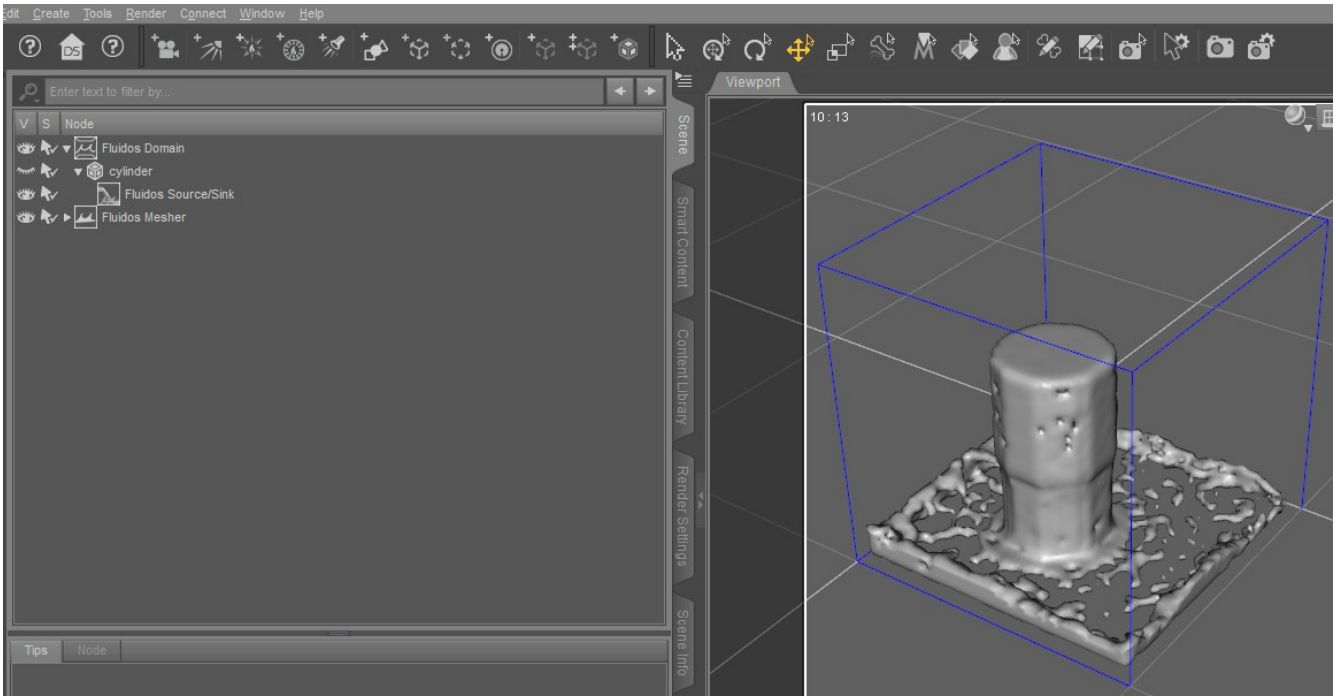

6. Let the default values of the properties of the Cylinder primitive.




7. In Fluidos Domain *Main Settings* change the *Number of frames* to 30 and the *Cell size* to 1.5.




8. Run the simulation, and enable the Mesher (disable Preview too) to see the results. Next image shows the frame 30.




*Note*: You can change the *Fluid origin* in the Source to *Volume*, but this case, you have to change the *Object type* property of Cylinder to *Ignored* (*Viscosity control* and *Fluid mass* would work too) to avoid the object to block the flow.

# APPENDIX D: Effects of the settings

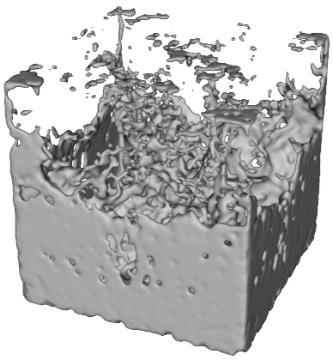
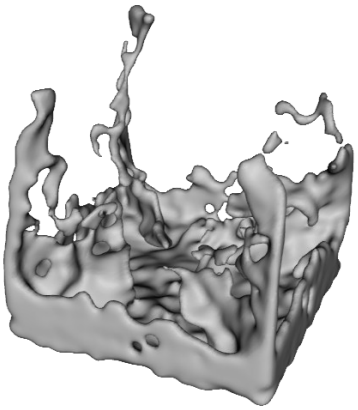
For the following images, it was used the Fluidos II *Scene 1* for liquids. For smoke, it was used the *smoke example*, and for Volumetric fire, the *fire example* one.

### *General properties of Fluidos Domain*

**Main settings**

*Cell size*

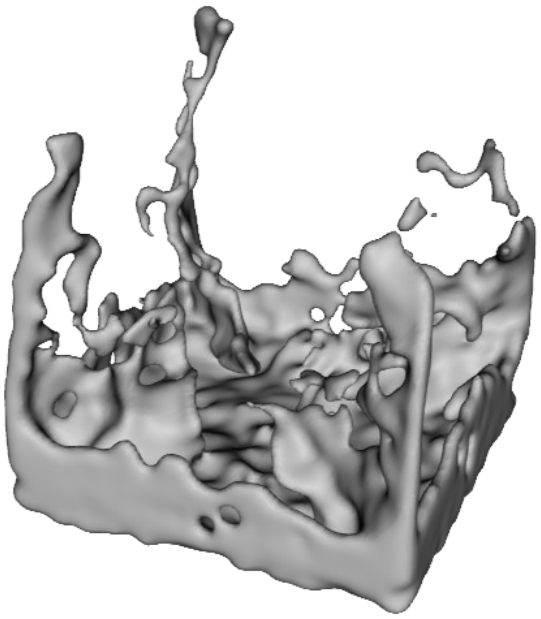
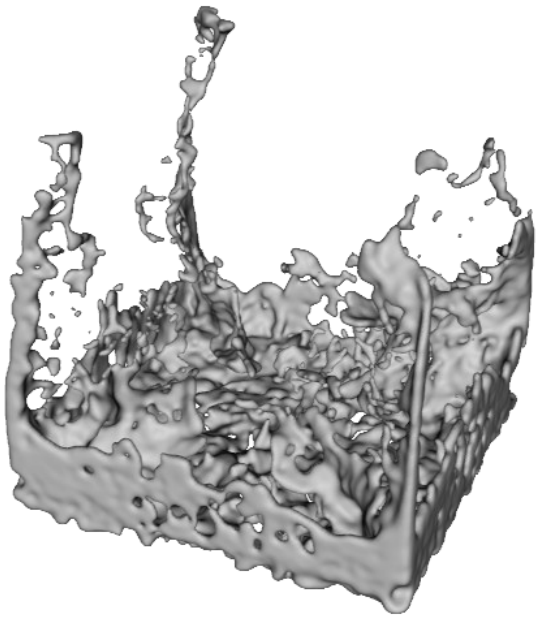
This property controls the resolution of the simulation.

| Cell size | | |
|---|---|---|
| 1.0 | 2.0 | 4.0 |
|  |  | |

Decreasing the cell size will give more details and more accurate results, but the calculations will take a longer time and the geometries will have more polygons. When the cell size is reduced by half, the calculation speed, the RAM consumed and the number of polygons in geometries will increase approximately eight times.

*Subdivision Level:*

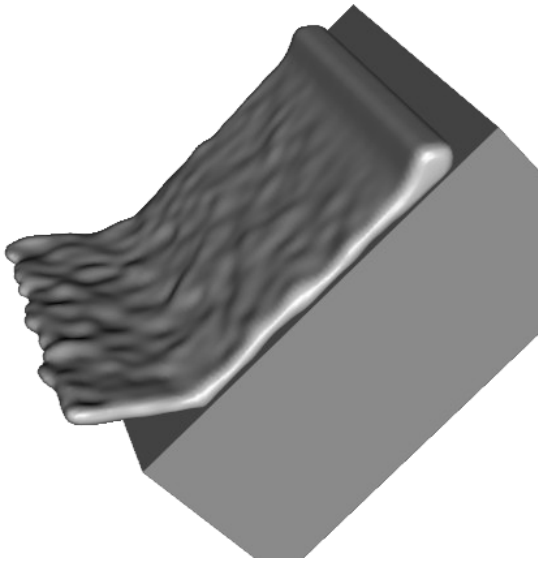
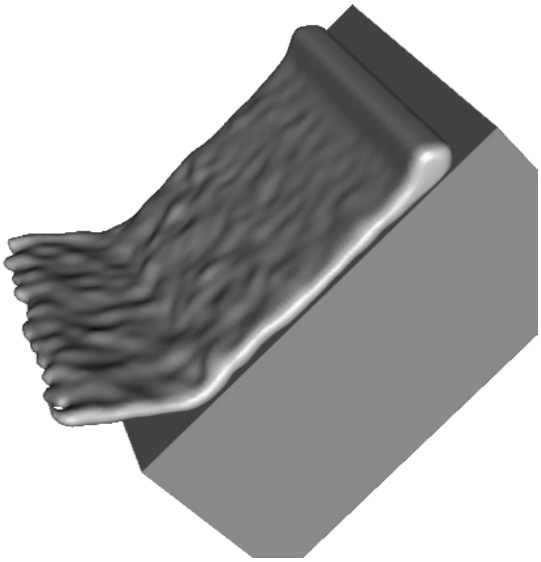
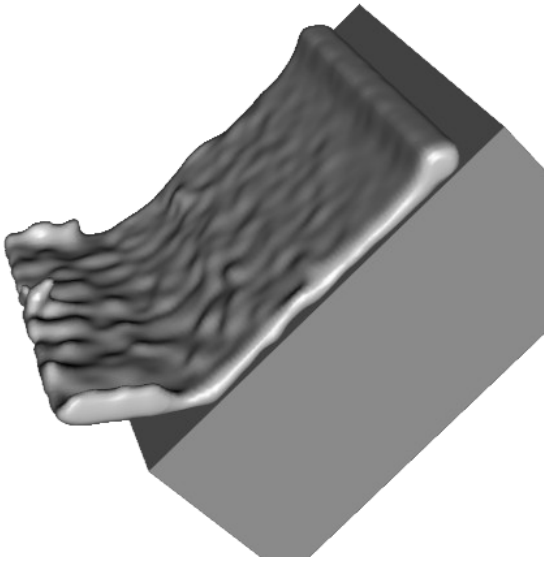
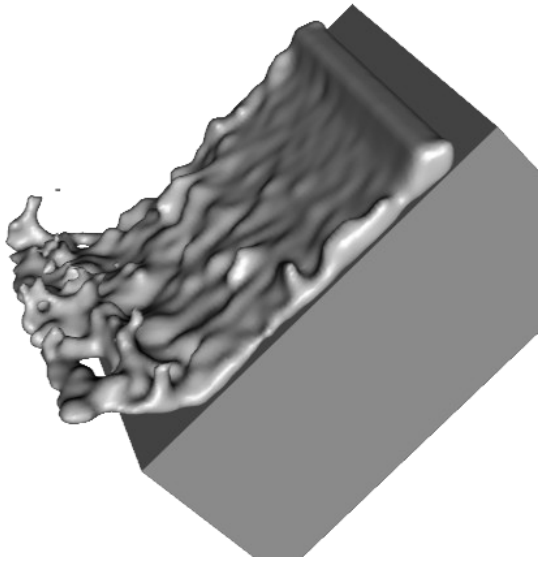
This property can take any numeric value between 1 and 3, decimals included.

| Subdivision level | |
|---|---|
| 1.0 | 2.0 |
|  |  |

The effect of the subdivision level is to give more detail without changing the cell size in the simulation. But the number of polygons will be increased.

*Vorticity confinement*

The vorticity confinement is a technique that enhances the vorticity in a fluid simulation.

| Vorticity confinement | |
|---|---|
| 0.0 (default) | 1.0 |
|  |  |
| 10.0 | 50.0 |
|  |  |

Vorticity confinement in Smoke

| Vorticity confinement – smoke particles | |
|---|---|
| 0.0 (default) | 1.0 |
| | |
| 10.0 | 50.0 |
| | |

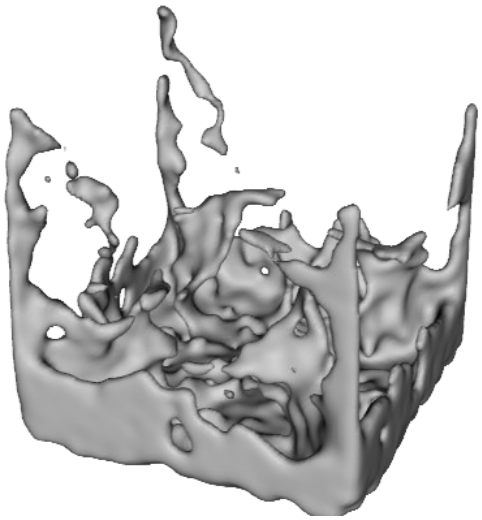
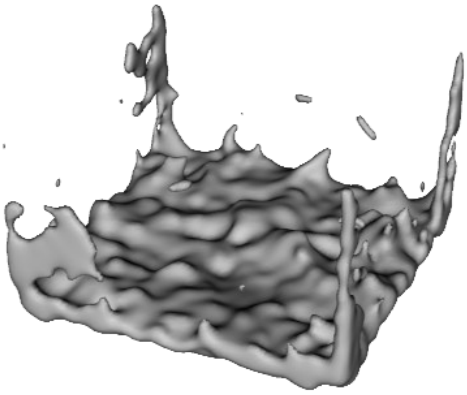
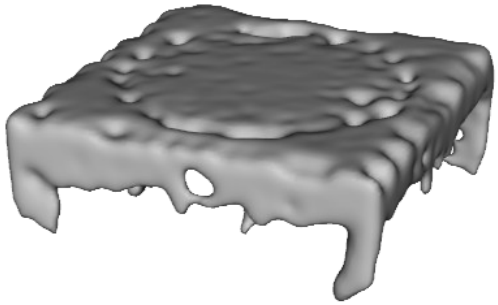
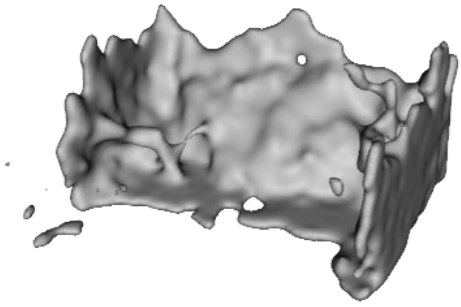
*Remove isolated particles:*

This property, when enabled, tells the Engine to remove the particles that are isolated, that is those that do not have any neighbor particle in the same cell of the simulation grid.

| Remove isolated particles – smoke particles | |
|---|---|
| Disabled | Enabled |
| | |

*Gravity forces*

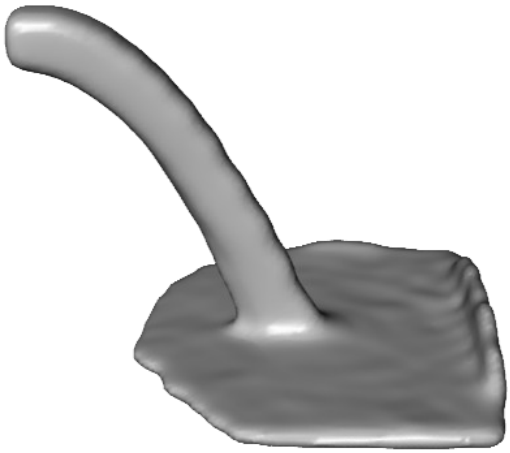
Gravity forces are constant forces that simulate gravity.

| Gravity forces ($m/s^2$) | |
|---|---|
| x = 0.0 , y = 0.0 , z = 0.0 | x = 0.0 , y = – 9.81 , z = 0.0 (default)  |
| x = 0.0 , y = – 20.0 , z = 0.0  | x = 0.0 , y = +9.81, z = 0.0  |
| x =0.0 , y = –9.81 , z = – 9.81  | |

**Viscosity subgroup**

*Viscosity*

This property simulates a viscosity effect. The greater the viscosity, the greater the resistance to flow.

| Viscosity | |
|---|---|
| Viscosity disabled – frame 30 | Viscosity = 2.0 – frame 30 |
| |  |
| Viscosity = 10.0 – frame 30 | Viscosity = 2000.0 – frame 300 |
| | |

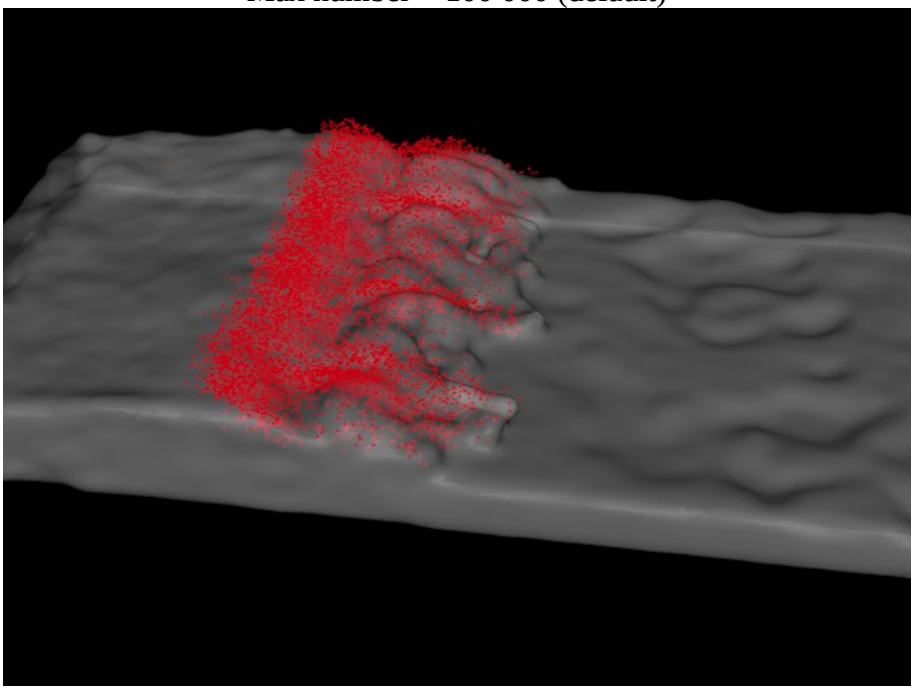
**Diffuse particles subgroup**

*Max number of diffuse particles*

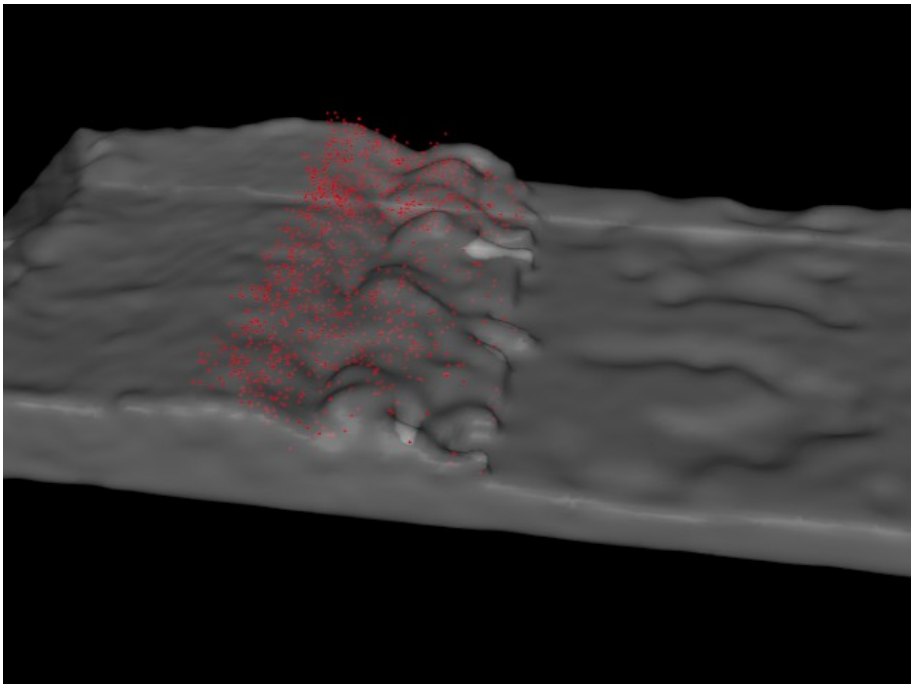
This property controls the upper limit of the number of diffuse particles.

Max number of diffuse particles comparison
All the other properties have the default values

Max number = 100 000 (default)

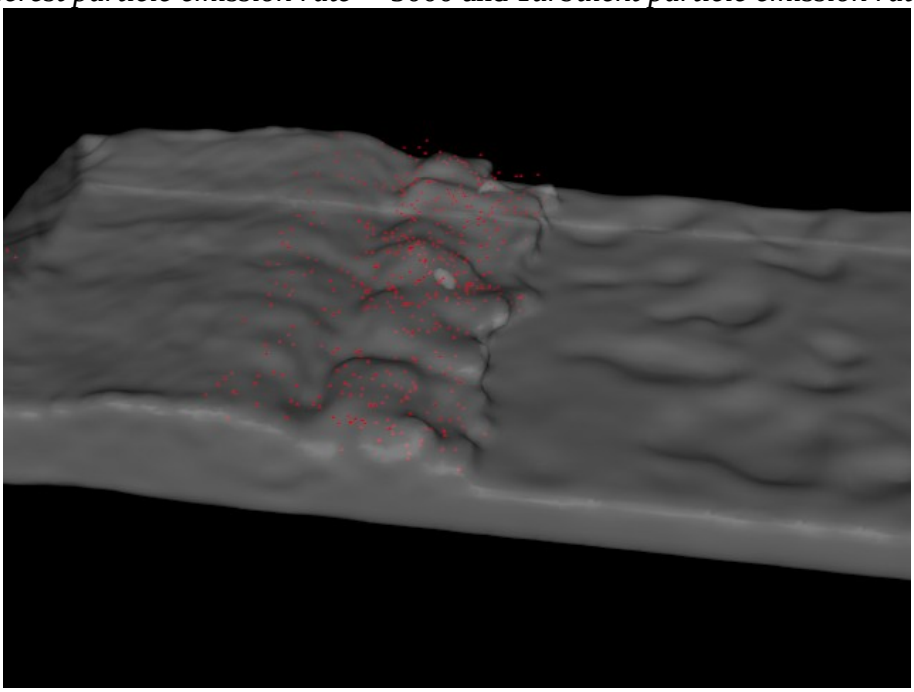
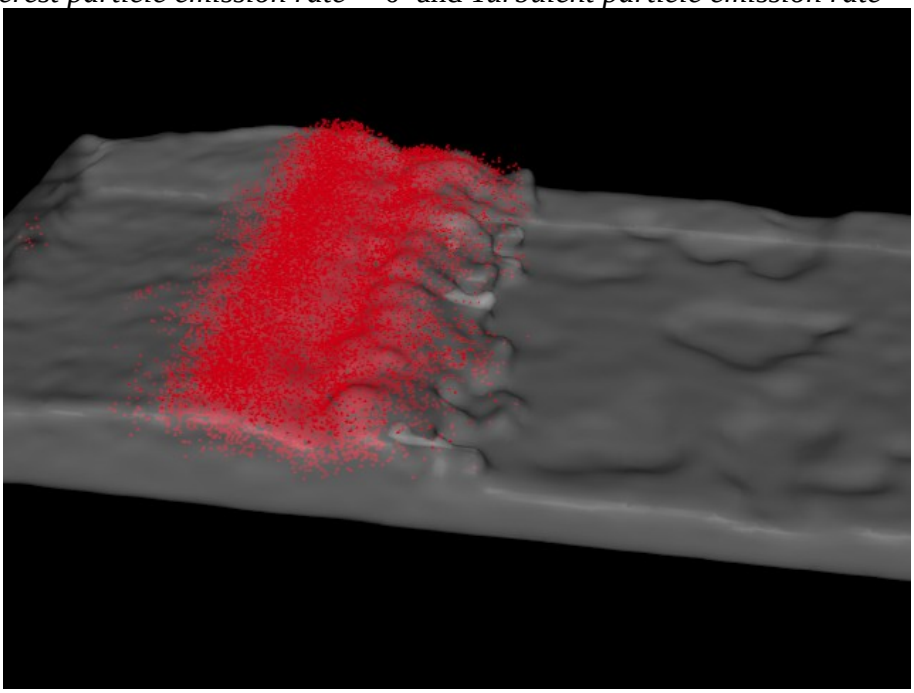


Max number = 10 000



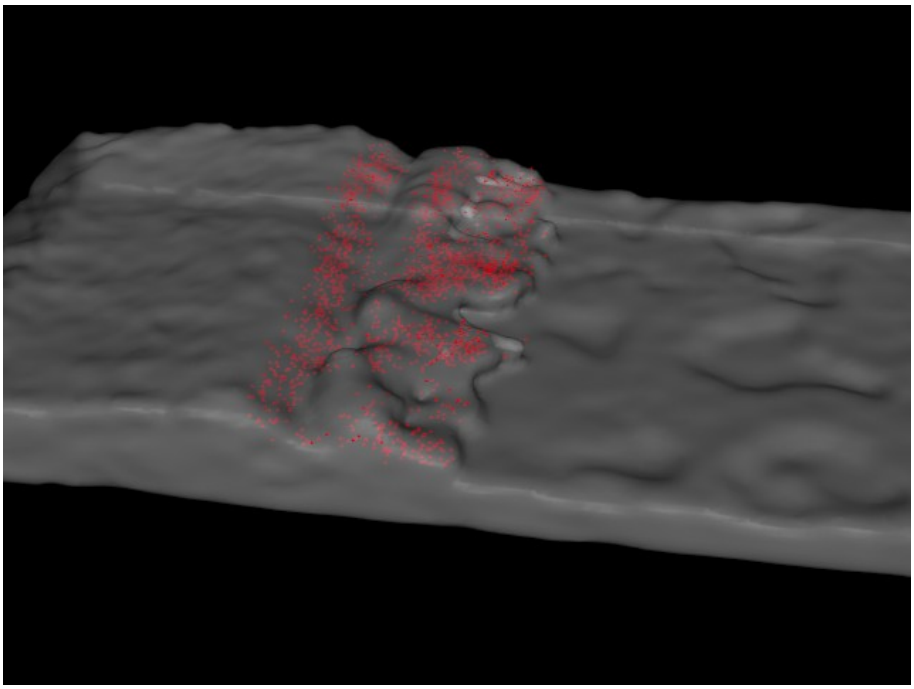
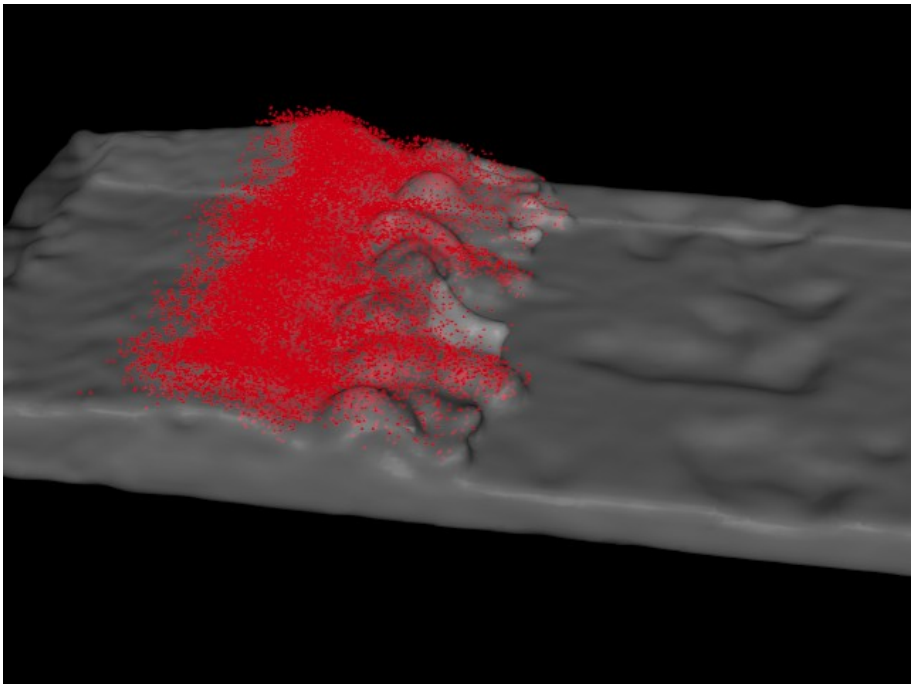
*Wavecrest particle emission rate* and *Turbulent particle emission rate*

The *Wavecrest particle emission rate* property controls the rate of emission of particles on a wave crest (emitted only from the fluid surface). Oh the other hand, the *Turbulence particle emission rate* controls the rate of emission of particles from the inside of a fluid due to turbulence.

| Wavecrest and turbulent particle emission rates comparison<br>All the other properties have the default values |
| --- |
| *Wavecrest particle emission rate* = 5000 and *Turbulent particle emission rate* = 0<br>  |
| *Wavecrest particle emission rate* = 0 and *Turbulent particle emission rate* = 5000<br>  |

*Max lifetime of a particle*

This property controls the time the particle could be in the scene before disappearing. The units are seconds. Bubbles take longer time than foam and this one longer than spray.

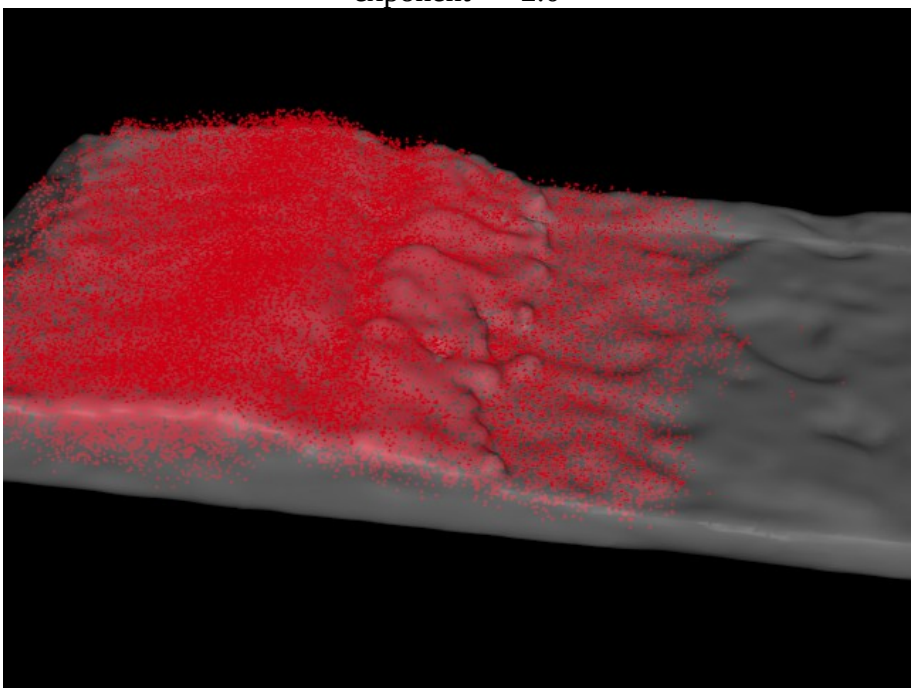
| Max lifetime of particle comparison<br>All the other properties have the default values |
| --- |
| Max lifetime = 0.2 s<br>  |
| Max lifetime = 10 s<br>  |

*Scaling exponent*

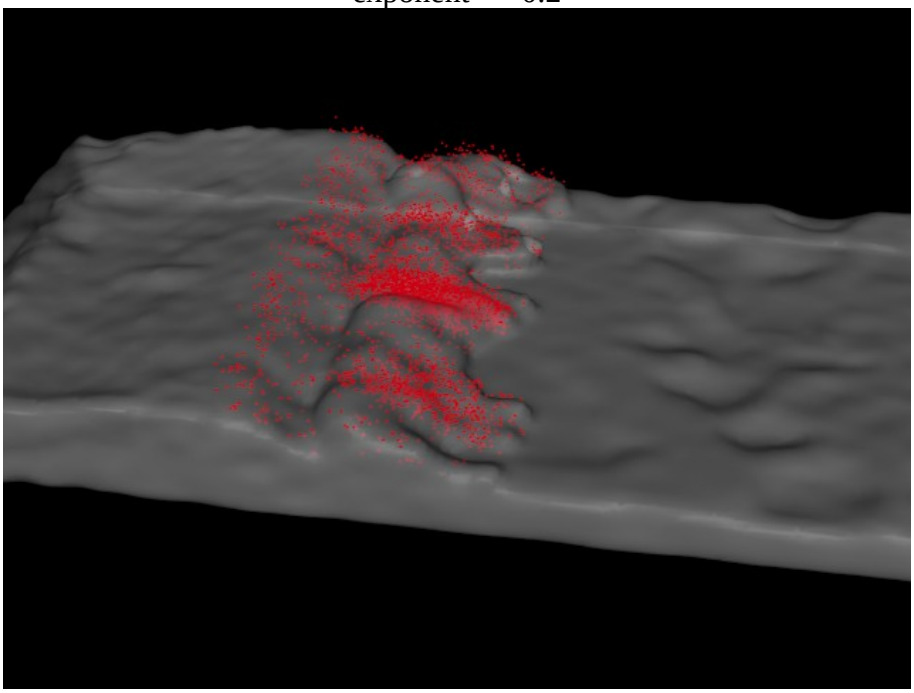
This property is intended to tune the distribution of the diffuse particles. By default, the particles will be emitted from enough large scenes, e. g. a turbulent waterfall will emit them, but a falling water drop will not. Changing the *scaling exponent* could simulate a greater or smaller scene scale.

Scaling exponent comparison
All the other properties have the default values

exponent = –2.0

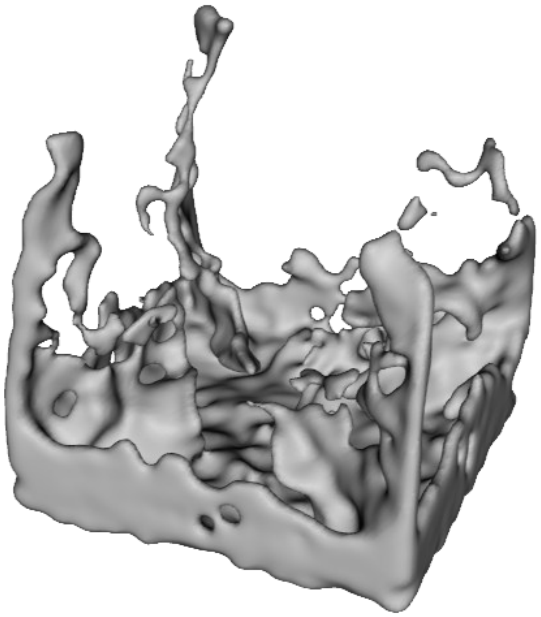
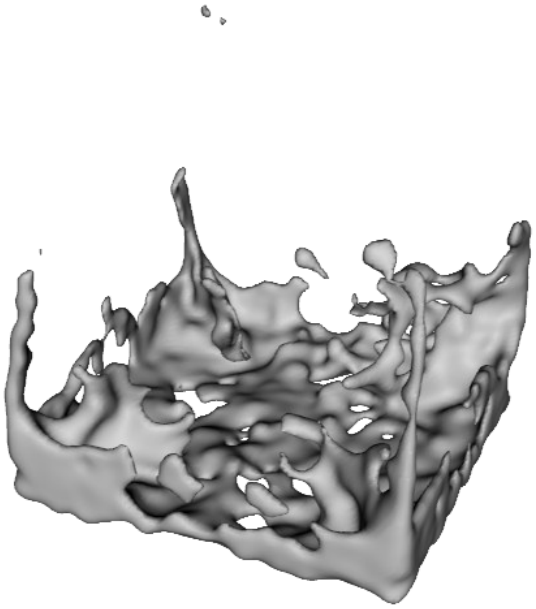


exponent = +0.2

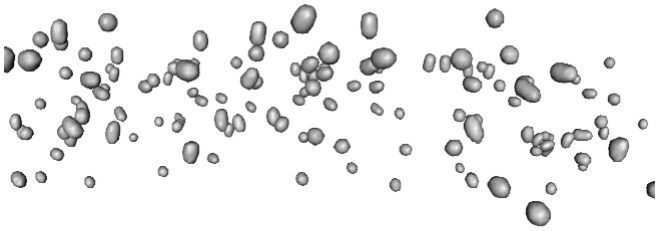
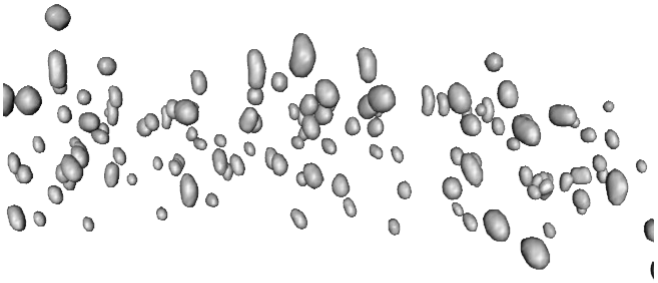


**Advanced setting subgroup**

*Meshing type: Fast, Better, Better-Trail and None*

| Meshing type | |
|---|---|
| Better | Fast |
|  |  |

The fast meshing, as its name says, is faster than the standard (*Better*) meshing type, but is less sensitive to the changes in *Marker Particle scale. Better* meshing can accept any positive number, with decimals, but the *Fast* will ignore decimals and some integer numbers as well.

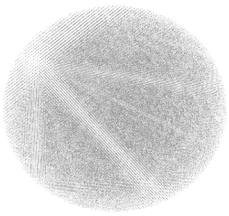
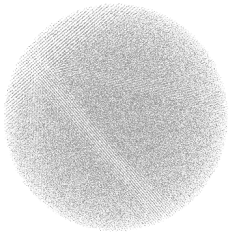
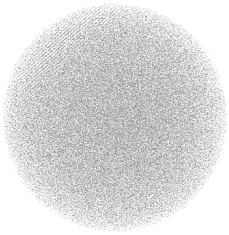
| Better | Better-Trail |
|---|---|
|  |  |
| Filter: Mean, 1 iteration | |

The trail filter elongates a little the mesh in the direction of the particle velocity.

If the user selects *None* as meshing type, the plugin will run the calculation, but won't write any file besides the .state file. This option could allow faster calculations when the first frames are not needed.

*Jitter factor:*

The *jitter factor* controls the arrangement of the particles in cells grid when they are created in the simulation (inside *Fluidos sources* and initial fluid mass). The higher the value, the more random the distribution.

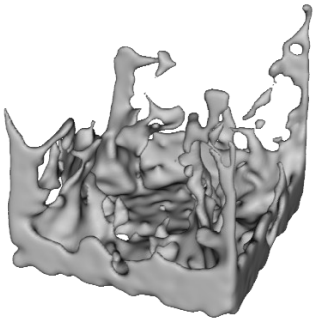
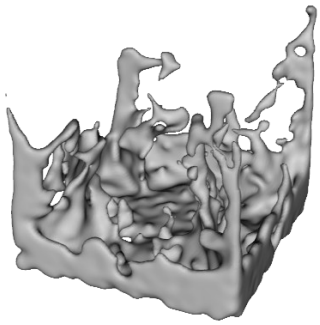
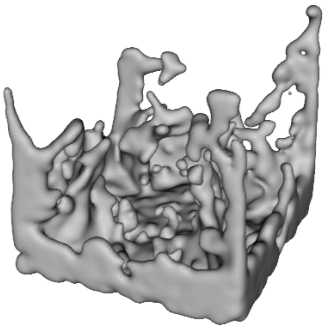
| Jitter factor – particles<br>Frame 0  Scene 1 | |
|---|---|
| 0.0<br> | 0.1<br>  |
| 0.5<br>  | 1.0<br>  |

Examples with Smoke simulation:

| Jitter factor – smoke particles | |
|---|---|
| 0.0 | 0.1 |
| | |
| 0.5 | 1.0 |
| | |

*Marker Particle Scale*.

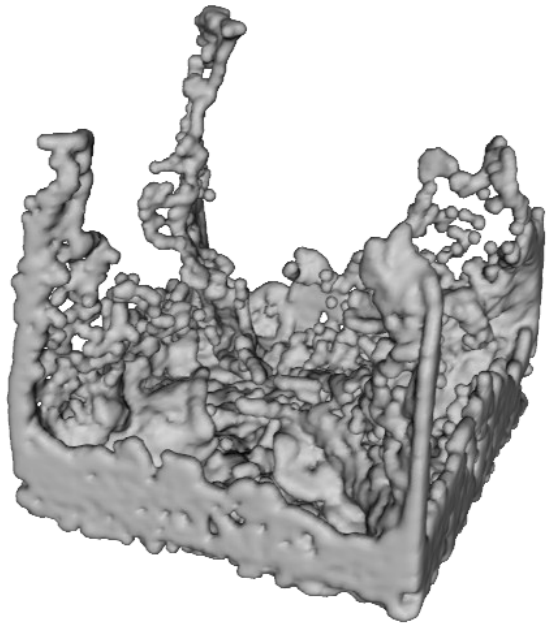
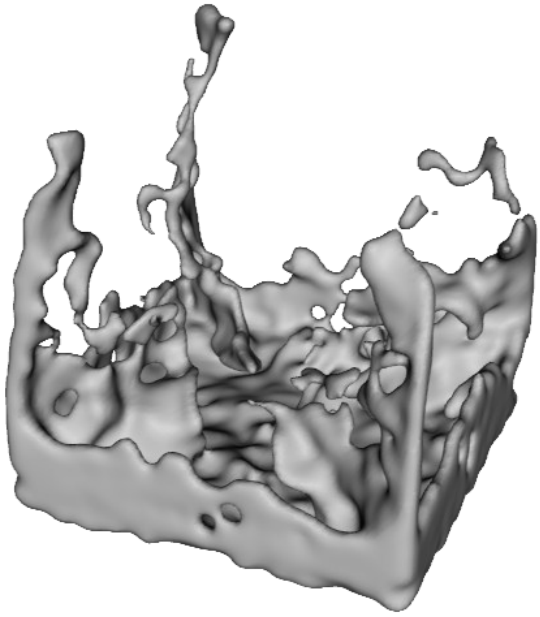
The plugin creates the fluid mesh from the particle's positions. As the particles themselves are points and have no dimensions, the plugin reads the Marker Particle scale value to assign a "diameter" to each particle. So, the larger the value, the "thicker" the fluid mesh will be.

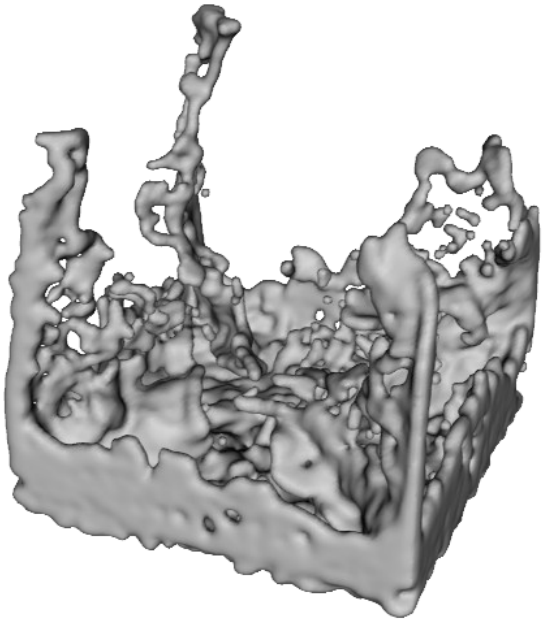
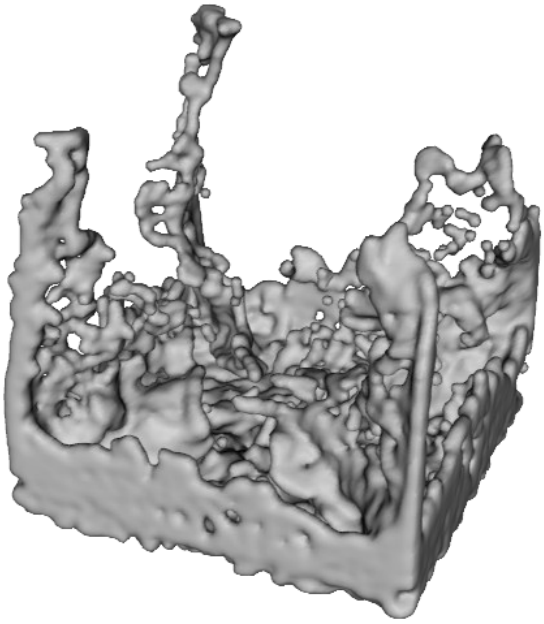
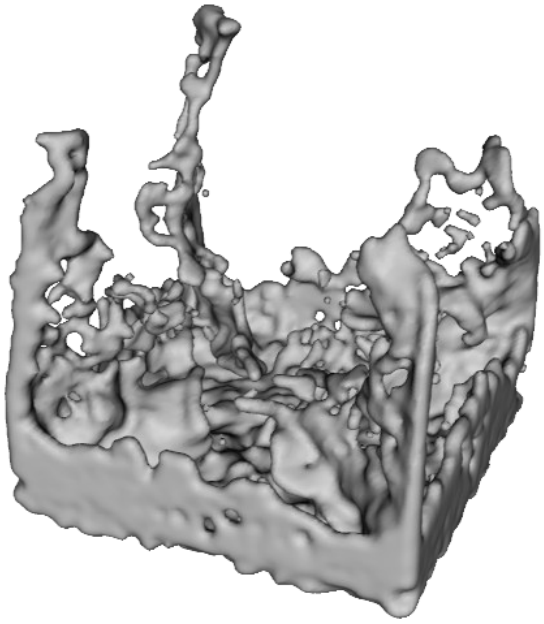
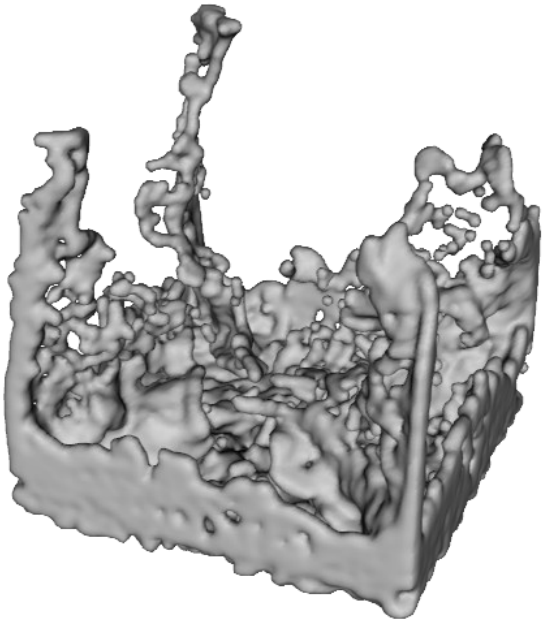
| Marker Particle scale | | |
|---|---|---|
| 1.0 | 3.0 | 10.0 |
|  |  |  |

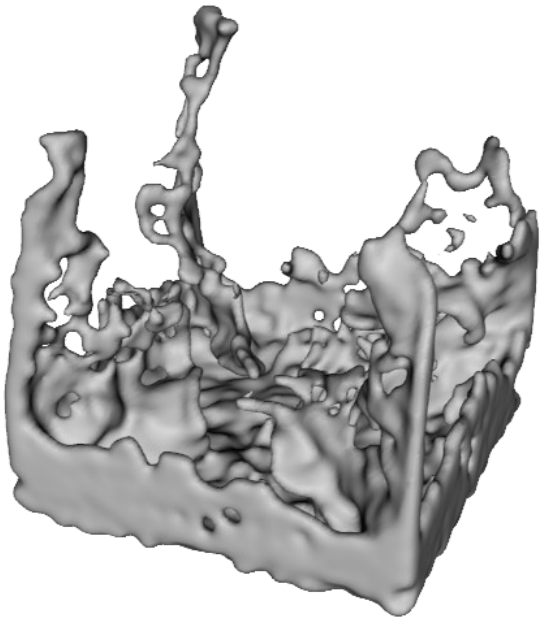
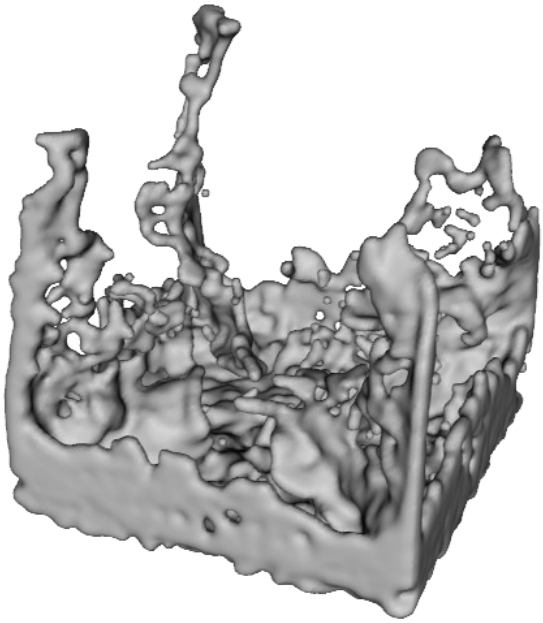
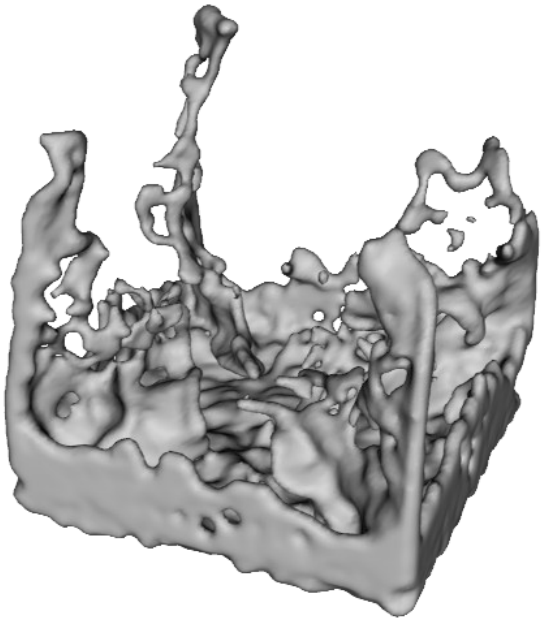
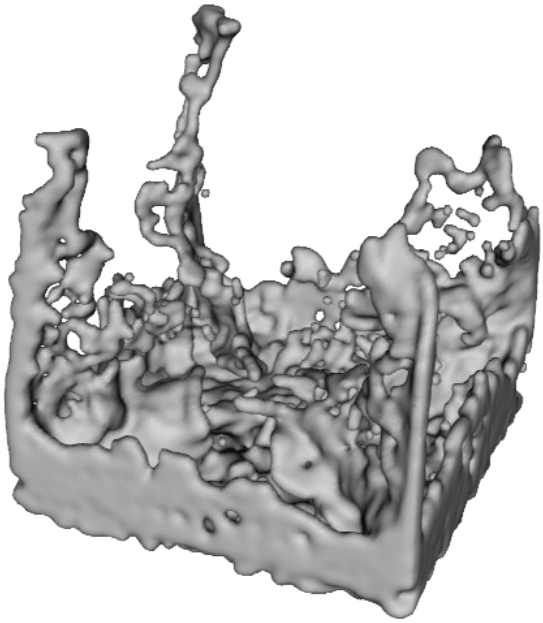
*Filters*

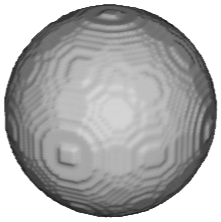
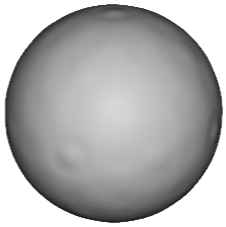
Fluidos II applies the chosen OpenVDB filter to the fluid mesh after the simulation calculations. The default is the gaussian filter.

The actual differences between filters are the softness degree and the sharp edges preservation.

| **Filters (one iteration)** | |
|---|---|
| No filter | Gaussian |
|  |  |
| | |

| Filters (one iteration) | |
|---|---|
| Mean | Laplacian |
|  |  |
| Median | Mean Curvature |
|  |  |

| Filters (two iterations) | |
|---|---|
| Mean | Laplacian |
|  |  |
| | |
| Median | Mean Curvature |
|  |  |

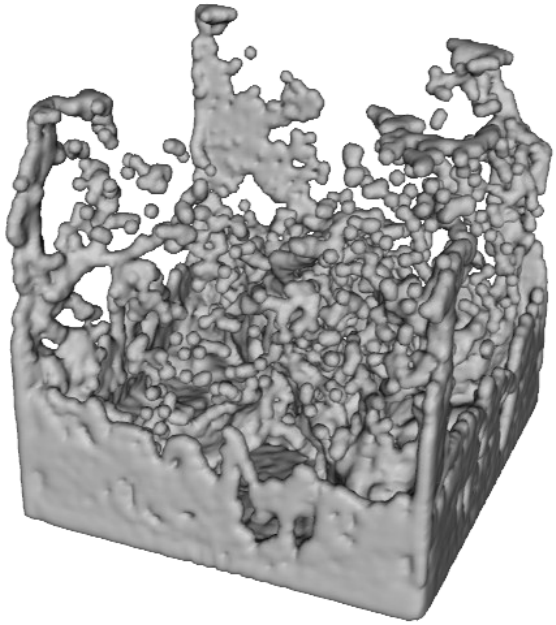
| Filter, one iteration, frame 0 of Scene 1 | | |
|---|---|---|
| No filter | Gaussian | Mean |
|  |  | |
| Laplacian | Median | Mean Curvature |
| | | |

*Dilate* and *Erode filters*

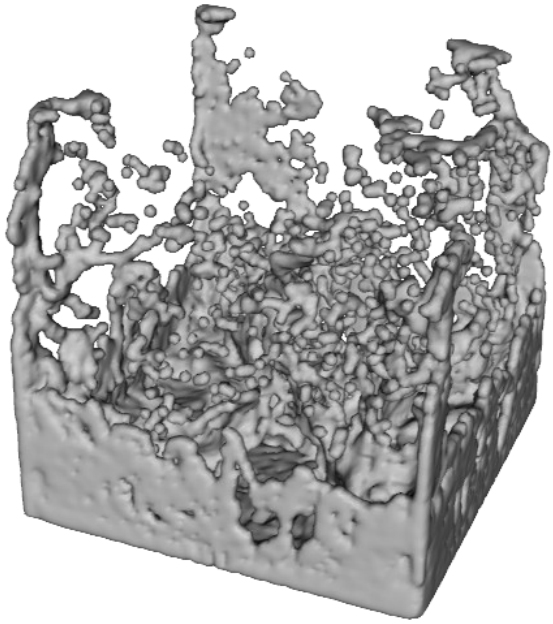
. The first one will tend to fatten the final mesh, whereas the latter will produce the opposite effect. They are very fast filters.

**Dilate and erode filters**

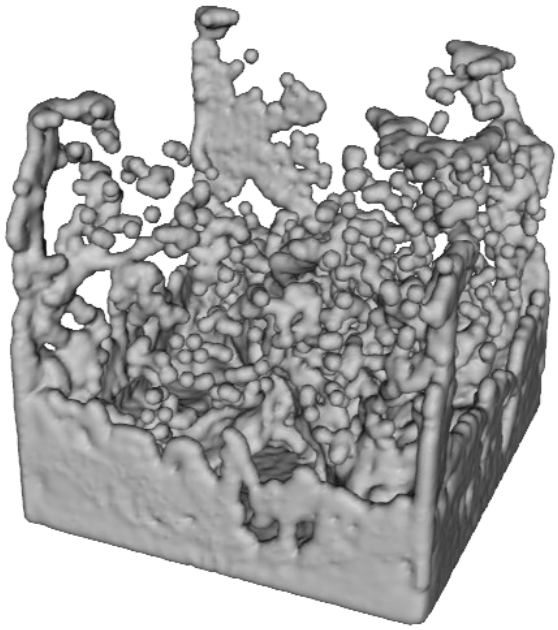
No filter



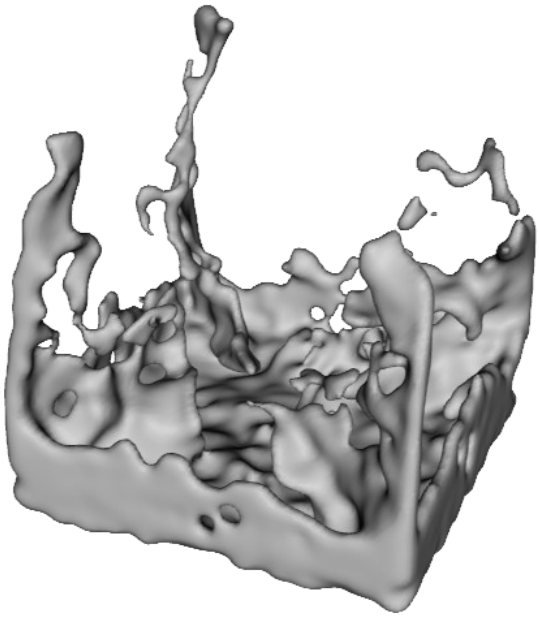
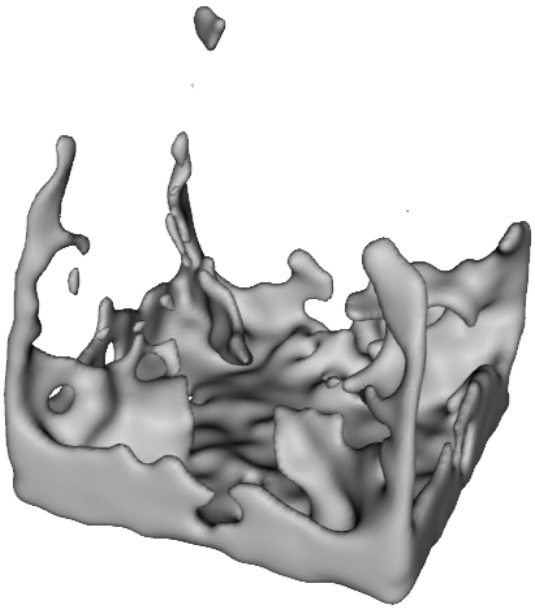
Erode (5 iterations)



Dilate (5 iterations)



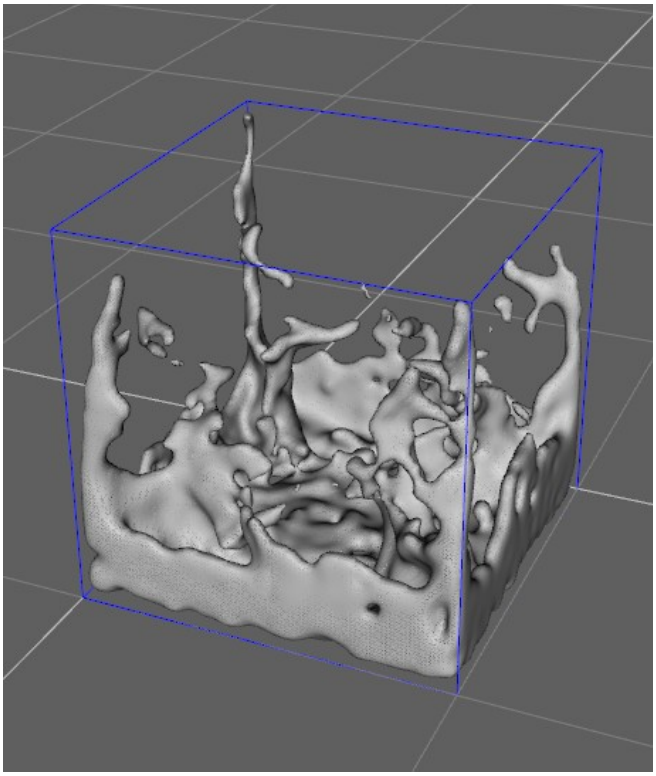
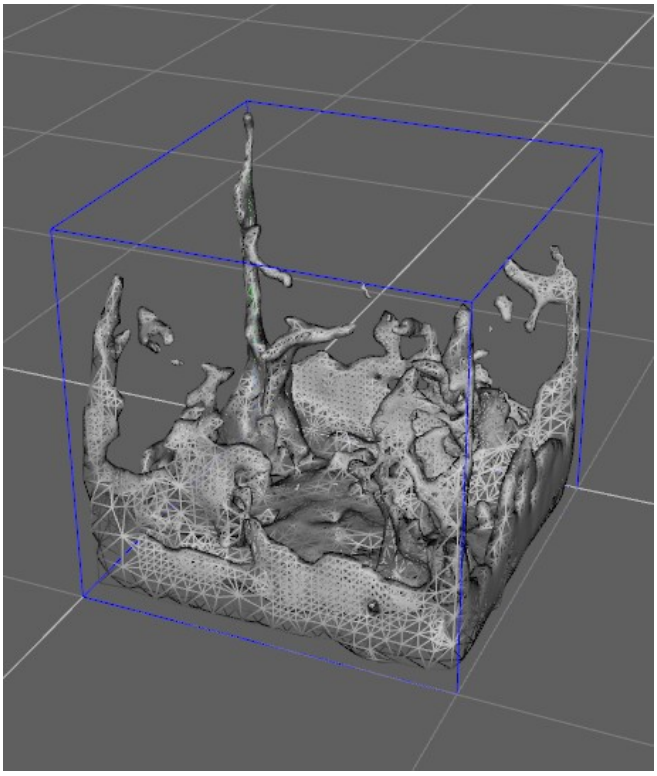
*Iterations*

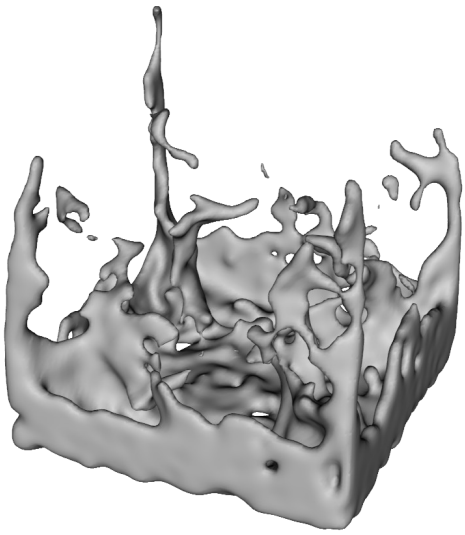
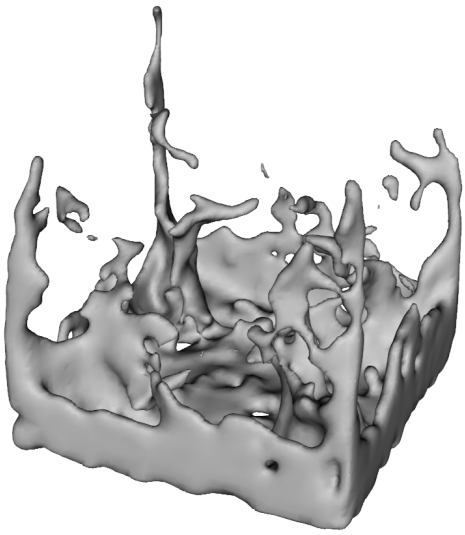
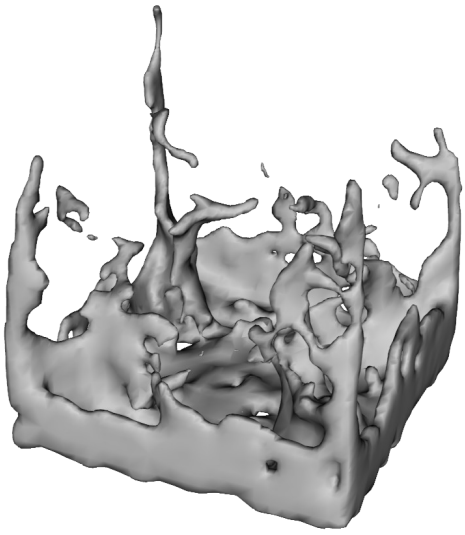
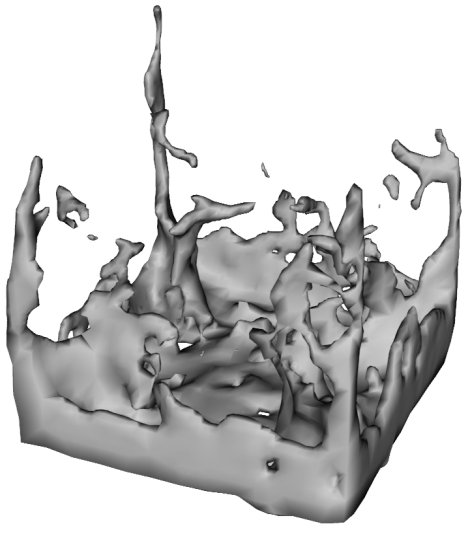
| Gaussian Filter | |
|---|---|
| One iteration | Two iterations |
|  |  |

The more iterations, the smoother, but the details will be lost, and the running time will be longer.

*Surface adaptivity*

The *surface adaptivity* controls the number of polygons in the generated fluid mesh. Automatically, the areas with less detail will be meshed with fewer polygons than the highly detailed areas.

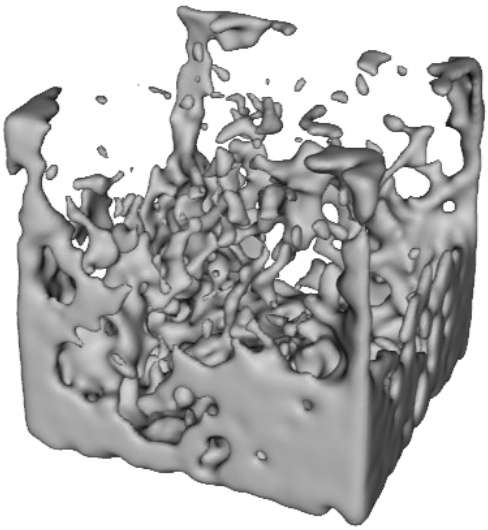
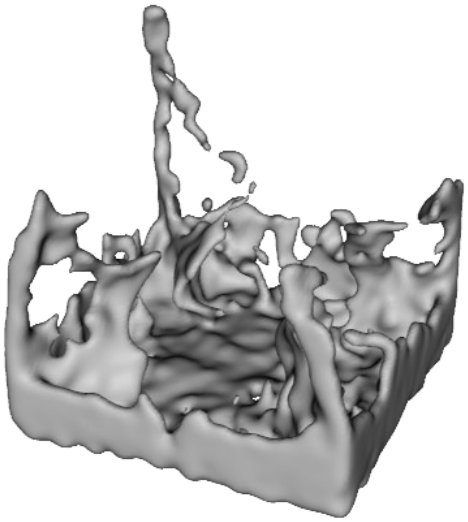
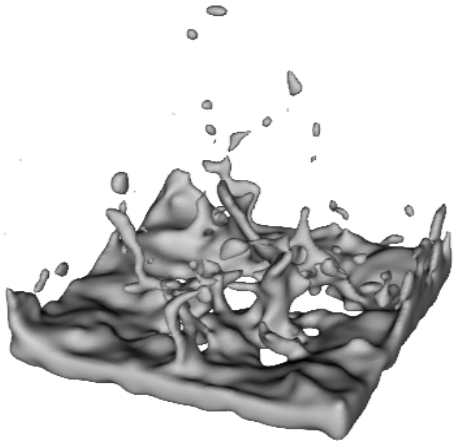
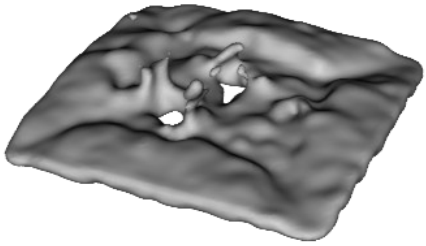
| Surface adaptivity<br>Viewport – Lit Wireframe | |
|---|---|
| 0.0 | 1.0 |
|  |  |

| Surface adaptivity - renderized | |
|---|---|
| 0.0  | 0.1  |
| 0.5  | 1.0  |

Too high values of *Surface adaptivity* could generate coarse surfaces.

*PIC/FLIP ratio*

The *PIC/FLIP ratio* has an effect similar to that of the *Viscosity*. Increasing it, the liveliness of the fluid simulation is reduced.

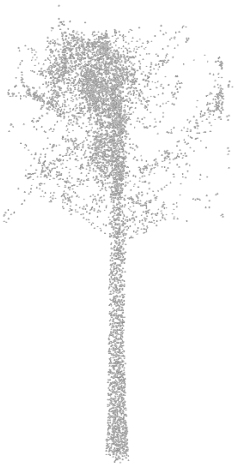
| *PIC/FLIP ratio* | |
|---|---|
| 0.0 | 0.05 (default) |
|  |  |
| 0.5 | 1.0 |
|  |  |

**Smoke specific properties**

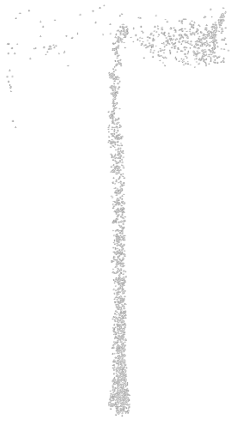
For *smoke*, the following images are created with *Fluid particles Mesh type* to show the finer details of the results of the simulation. But for smoke and fire could also be used the *Fluid mesh* (and the *Fire mesh* for fire).

1. *Smoke temperature* (*at Fluidos source or fluid mass*).

The buoyancy of smoke (and Volumetric fire) depends on the differences in temperature between the ambient and the fluid itself. The greater the difference the greater the buoyancy.

| Smoke temperature (Ambient temperature is 25ºC)<br>Boussinesq method | | |
|---|---|---|
| 100 ºC | 150ºC | 500ºC |
| |  | |

2. *Beta buoyancy factor*

| Beta buoyancy factor comparison | | |
|---|---|---|
| 0.5 | 1.0 (default) | 2.0 |
| | |  |

The *Beta buoyancy factor* affects the buoyancy in Smoke simulations.

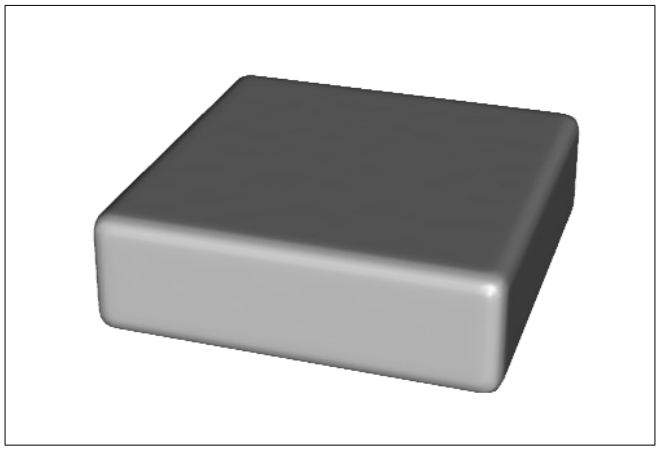
*3. Ambient temperature*

| Ambient temperature while maintaining the smoke temperature as constant (250ºC) | | |
|---|---|---|
| 25 ºC | 125 ºC | 200 ºC |
| 250 ºC | 300 ºC | |

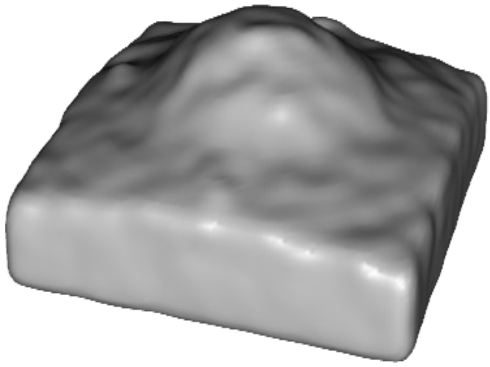
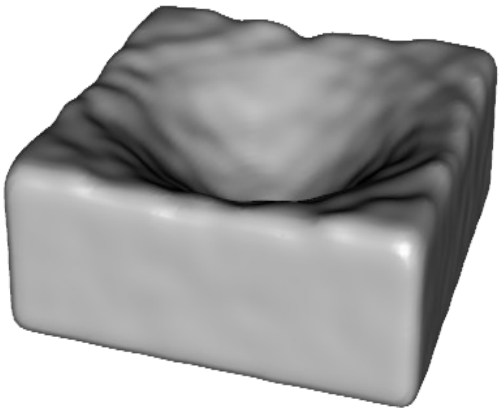
The *Ambient temperature* in smoke simulation only has an effect if there is a difference in temperatures in relation to the smoke one.

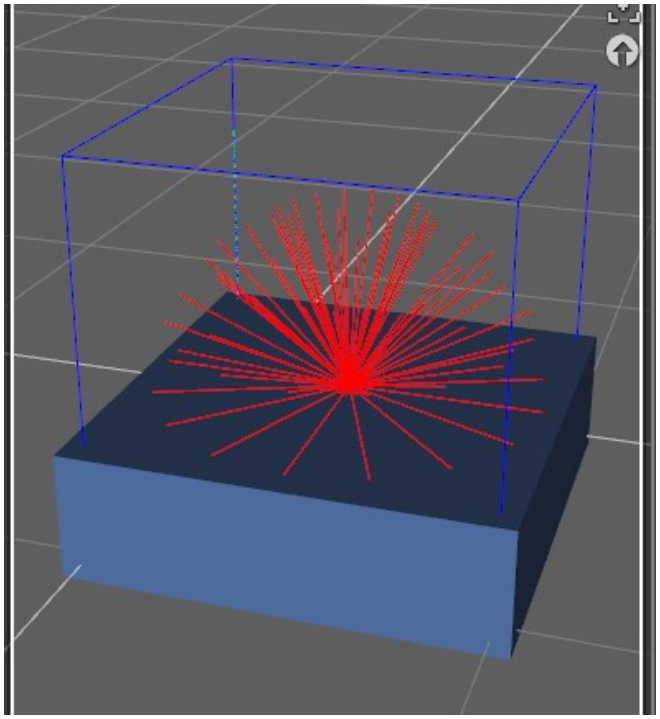
## Forces

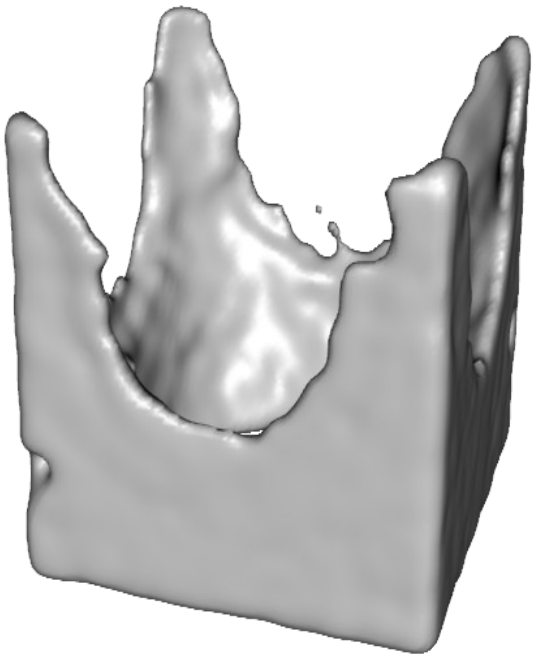
No force

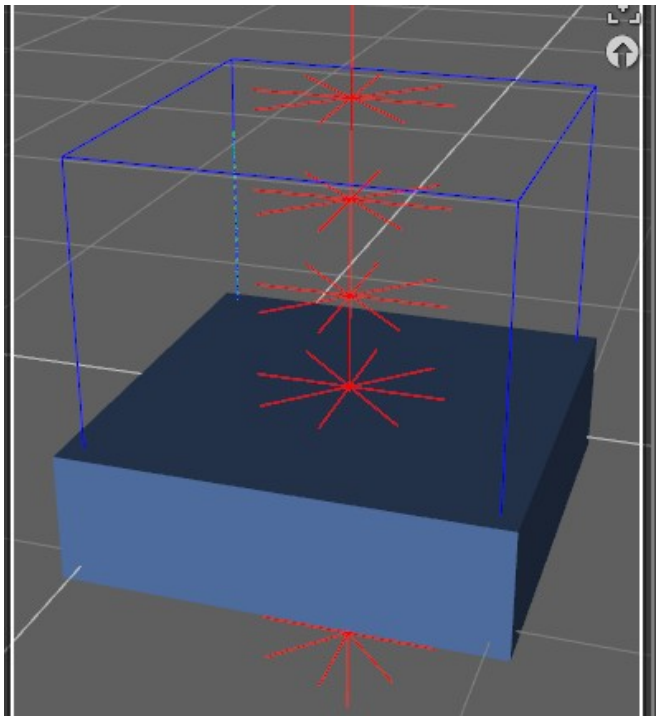


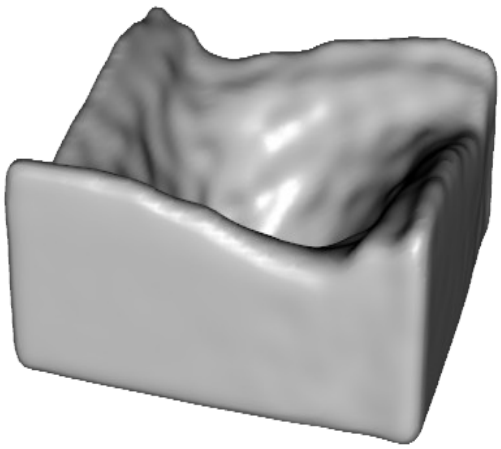
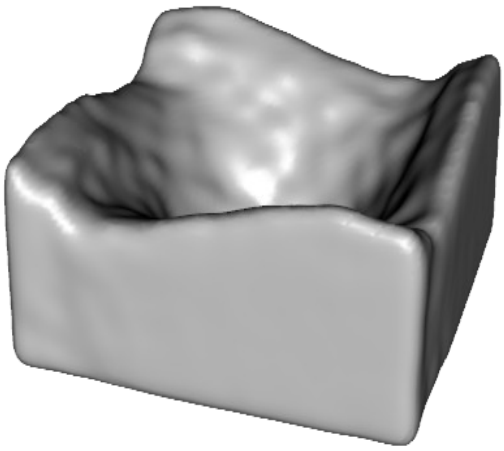
In the following screen snapshots that show the Daz Studio scenes, the *forces* are in red, the *fluid mass* is in blue.

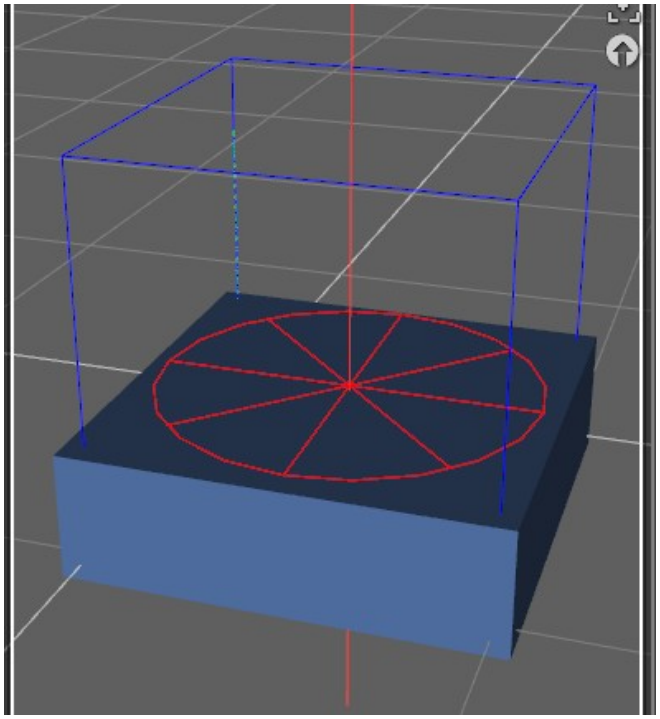
| Point Force | |
|---|---|
| 250 | - 250 |
|  |  |

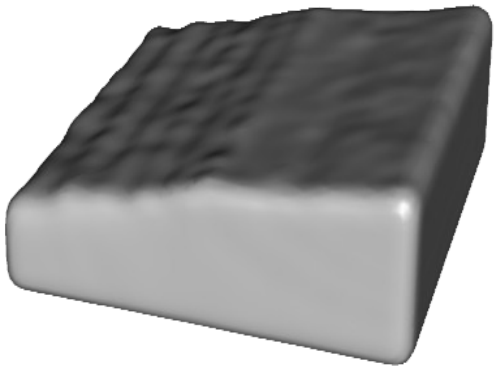
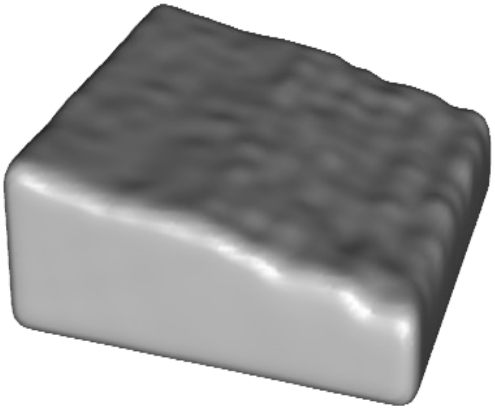


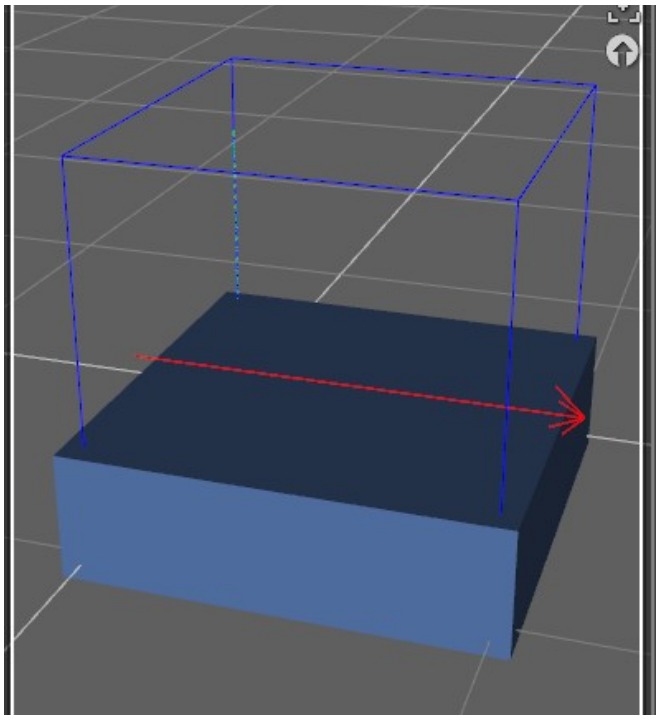
| Linear Force | |
|---|---|
| 50 | - 50 |
| |  |

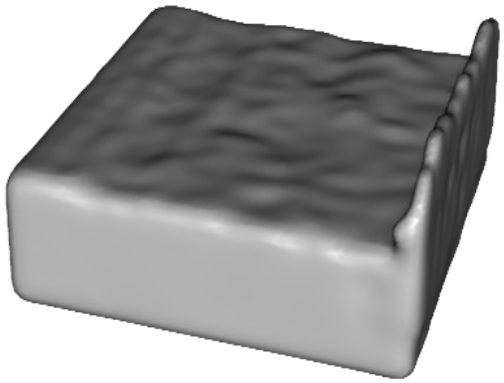
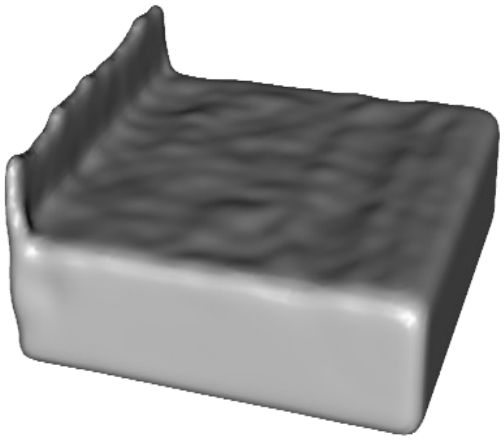


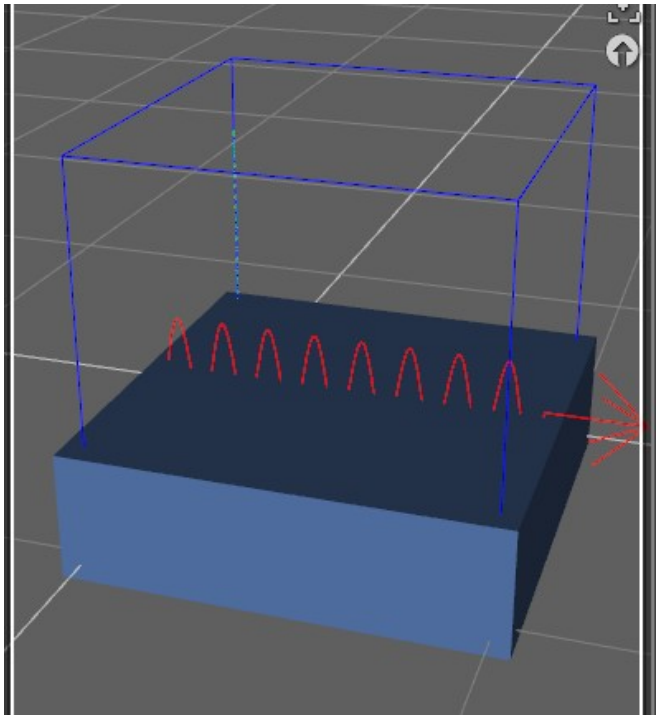
| Torque Force | |
|---|---|
| 5.0 | - 5.0 |
|  |  |

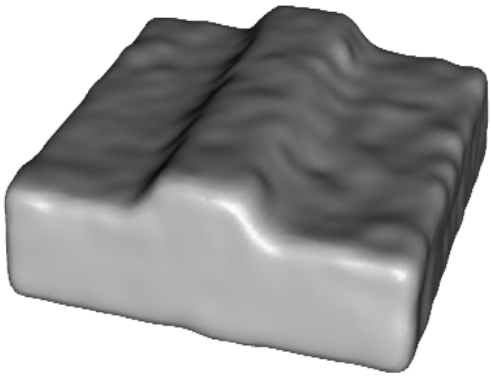
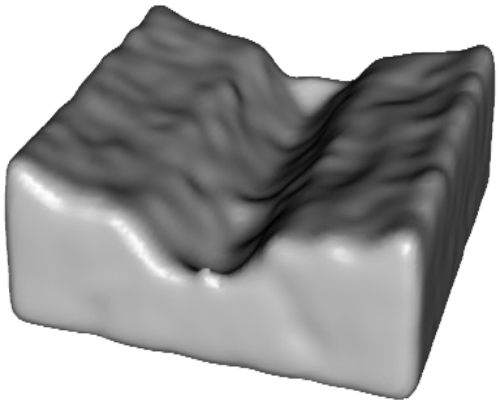


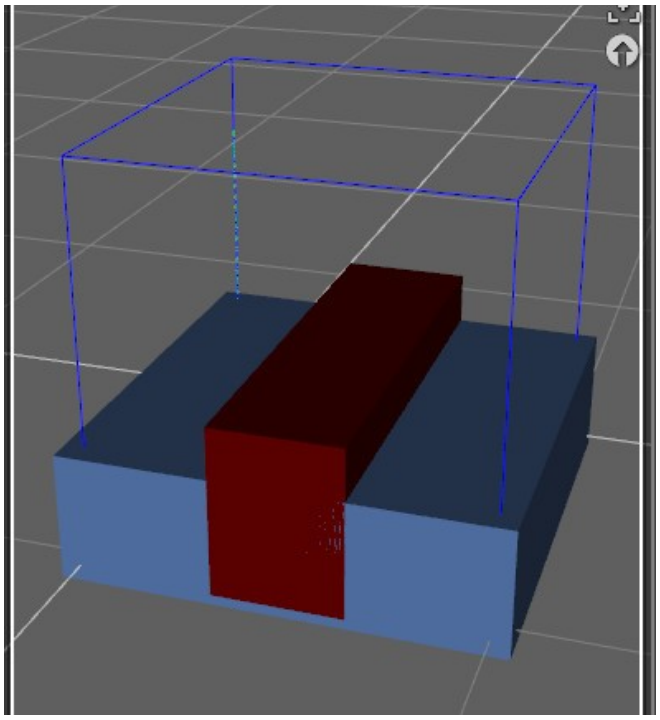
| Directional Force | |
|---|---|
| 1.5 | - 1.5 |
|  |  |



| Flow Force | |
|---|---|
| 0.5 | - 0.5 |
|  |  |



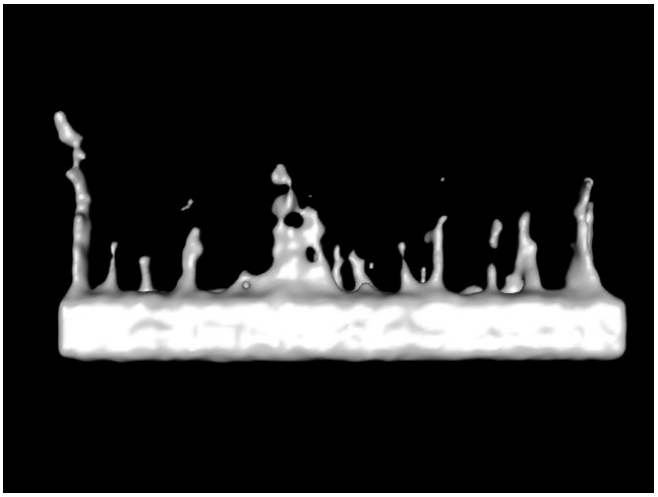
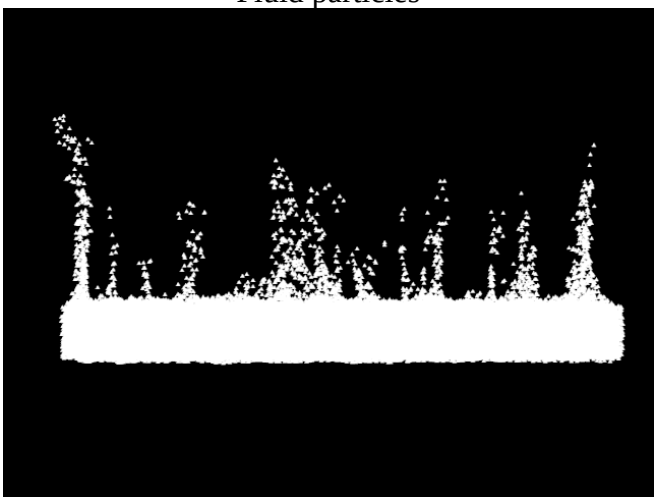
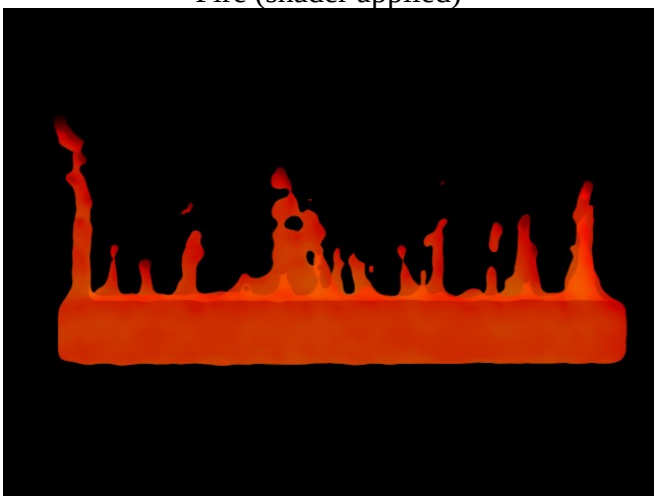
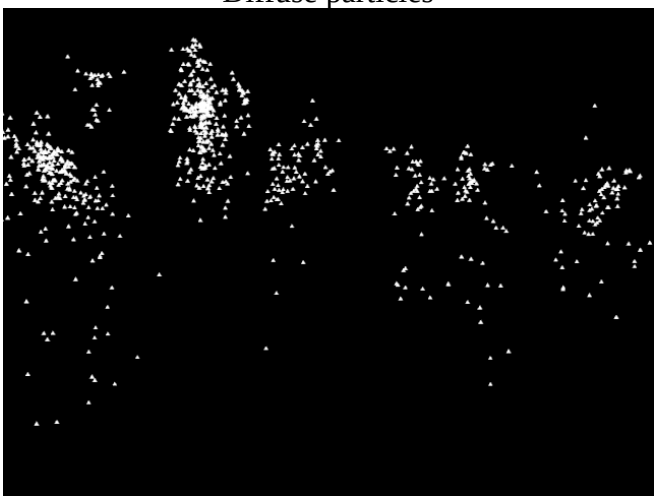
| Force field | |
|---|---|
| 10.0 | - 10.0 |
|  |  |



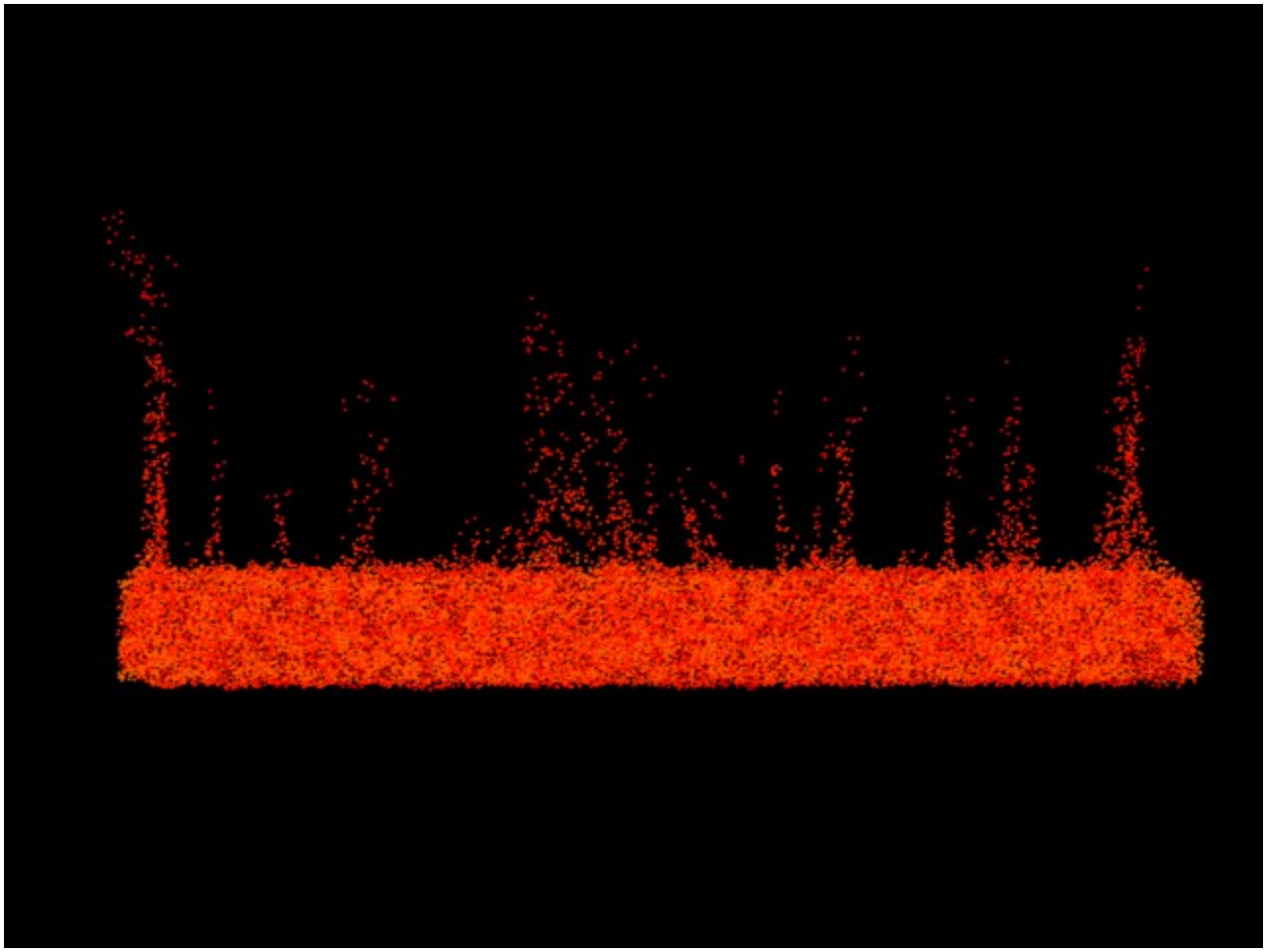
*Mesh type*

There are five types: *Fluid*, *Fluid particles*, *Fire*, *Diffuse particles* and *Volumetric OpenVDB*. The *Fluid mesh* is a “solid”, closed mesh. The *Fire mesh* is also a “solid” mesh, but it has associated temperatures data used by a special shader. The *Fluid particles mesh* and the *Diffuse particles mesh* are triangles representing each particle. *Fluid particles* mesh, just as the *Fire mesh*, has associated temperature data used by a special shader.

**Note**: LITE edition does not support Volumetric fire simulations, but it can read files generated by the Fluidos II complete edition.

<table>
<tr><td colspan="2">Mesh type comparison</td></tr>
<tr><td>Fluid<br>  </td><td>Fluid particles<br>  </td></tr>
<tr><td>Fire (shader applied)<br>  </td><td>Diffuse particles<br>  </td></tr>
</table>

Fluid particles mesh with a fire shader:



*Meshers for smoke:*

For rendering a smoke simulation, it could be used a *Fluid* or a *Fluid Particle mesh type* or both. That is, the user can add two meshers to the scene and select a *Fluid mesh type* for the first and a *Fluid particle* mesh for the other.

*Meshers for fire:*

For rendering a volumetric fire simulation, it could be used a *Fluid Fire* or a *Fluid Particle mesh type* or both. Besides those, the user could add also a *Fluid mesh* and apply a smoke shader to it.

It is advisable to the user to try the possible combinations to achieve the best possible effect.

The *Fire mesher* and the *Fluid particle* have two properties to adjust the color temperature of flame: *Color temperature scale* and the *Color temperature offset.* The offset increases or decreases the minimum temperature among particles; the scale, increases or decreases the range of temperatures.

If you feel the fire is too reddish, increase the offset or the scale value; if it's too much whitish, decrease the offset or the scale.

*Erosion of flames*:

This mesher's property apply only to the *Fire mesh* type. It "thins" the flames by uniformly eroding the boundaries.

| *Erosion of flames* comparison | |
|---|---|
| 0.0 | 0.3 |
| | |

| Comparison between *Fluid particle mesh* and *Fluid mesh* in a smoke simulation | | |
|---|---|---|
| *Particles mesh* | *Fluid mesh* | *Volumetric OpenVDB* |
| | | |

This is an example of *Fluid Particles mesh*

For small scale scenes, it could be better to use a *Fluid mesh* for smoke simulations. For large scale scenes, the use of *Fluid particles mesh* give better details.

**Note 1**: Increase the size of the particles in the mesh until you get a satisfactory result. If needed, disable the limits of the *Particles size* in the mesher to set any size you wish.

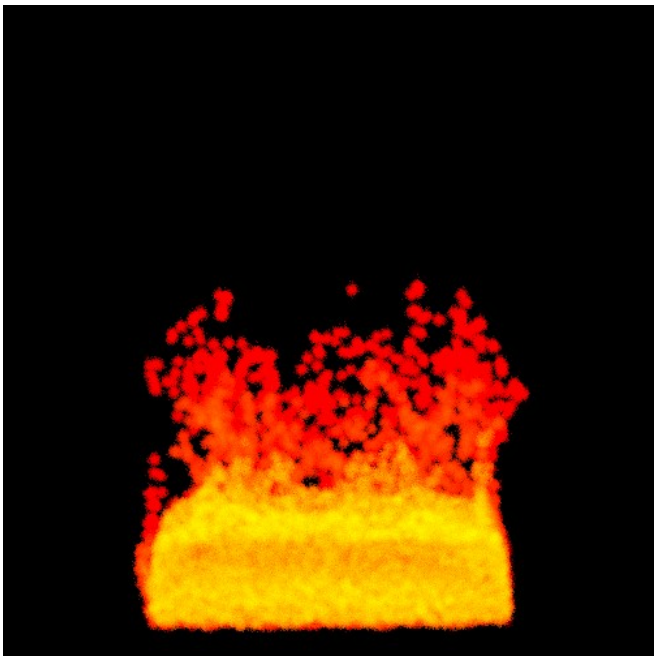
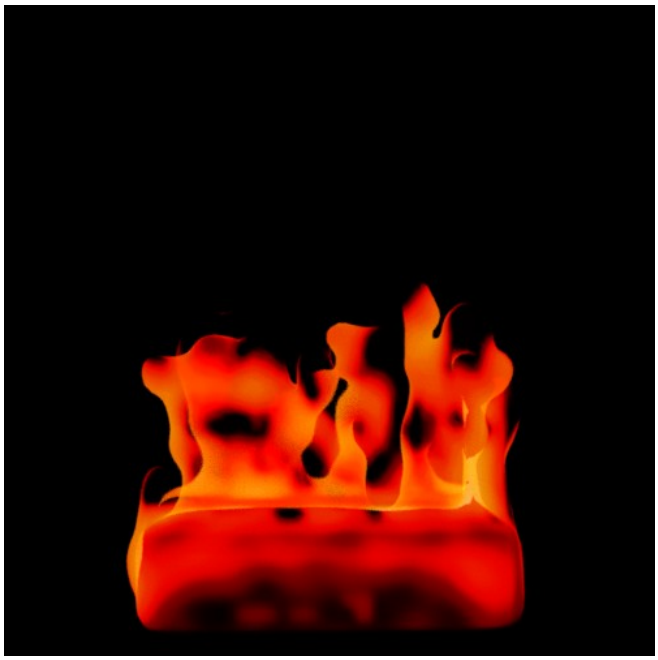
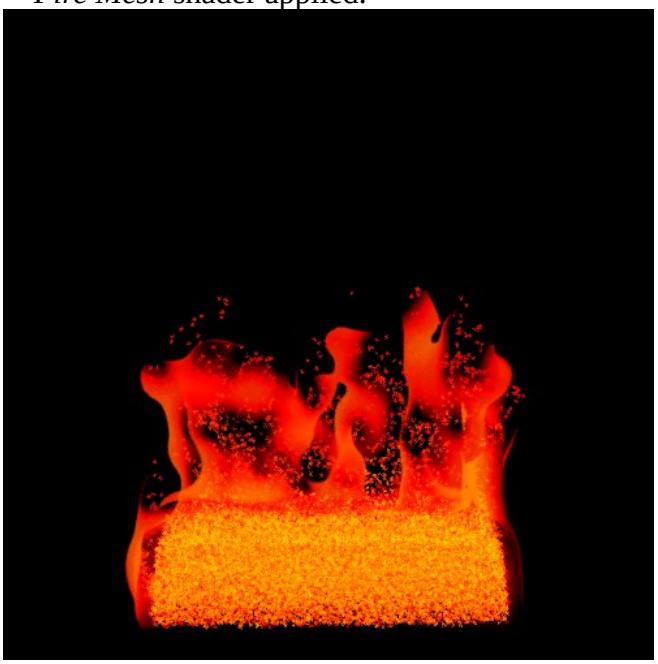
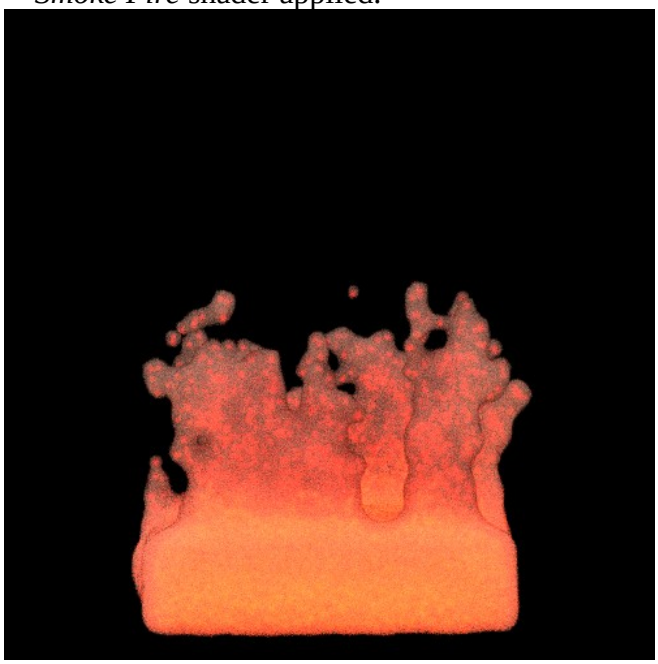
**Note 2**: In Fluidos II content there are opacity shader utilities to adjust the opacity of the particles and fluid mesh shaders.

## Shaders

There are some shaders included with Fluidos II for Iray and 3Delight engines. They are intended to be applied to meshers according to their mesher type.

| Shader type | Mesh type |
|---|---|
| Fluidos II Diffuse particles | *Diffuse particles* |
| Fluidos II Fire particles | *Fluid particles* |
| Fluidos II Fire Mesh | *Fire mesh* |
| Fluidos II Fire Mesh Backdrop | *Fire mesh* when there is only a backdrop in the background of a fire image. |
| Fluidos II Smoke | *Fluid mesh* |
| Fluidos II Smoke Fire | *Fluid mesh*<br>This is useful when using two meshers, one with *Fluid mesh* mesh type and the other with *Fluid particles* mesh type in fire simulations. |
| Fluidos II Smoke Particles | *Fluid particles* |
| Fluidos II Smoke VDB | *Volumetric OpenVDB* |

| Comparison between meshers and shaders in a smoke simulation | | |
|---|---|---|
| *Fluid particles* mesh type in a Mesher with a *Fluidos II Smoke Particles* shader applied.<br>Particle size: 70 | *Fluid mesh* type in a Mesher with a *Fluidos II Smoke* shader applied | *Volumetric OpenVDB* type in a Mesher with a *Fluidos II Smoke* VDB shader applied |
| | | |

| Comparison between meshers and shaders in a volumetric fire simulation | |
|---|---|
| *Fluid particles* mesh type in a Mesher with a *Fluidos II Fire particles* shader applied.<br>Particles size:14<br>  | *Fluid mesh* type in a Mesher with a *Fluidos II Fire Mesh* shader applied.<br>  |
| Two meshers enabled:<br>*a. Fluid particles* mesh type in a Mesher with a *Fluidos II Fire particles* shader applied. Particles size:4<br>*b. Fluid mesh* type in a Mesher with a *Fluidos II Fire Mesh* shader applied.<br>  | Two meshers enabled:<br>*a. Fluid particles* mesh type in a Mesher with a *Fluidos II Fire particles* shader applied. Particles size:8<br>*b. Fluid mesh* type in a Mesher with *Fluidos II Smoke Fire* shader applied.<br>  |

New in Version 2.3

- The library OpenVDB was updated from v. 6.1.0 to v. 9.1.0. Many simulations will be faster, in consequence.
- The behavior of moving solids was improved (The *CFL Condition number,* however must be low, less than 2.0)
- It can use a range of frames for simulation instead of the usual starting at frame 0. You should select the Current frames range in the *Fluidos Domain*, and *Absolute frame numbers* in the *Fluidos Mesher.*
- Fixed a bug that happened when the user try to continue a simulation of a *Narrow band* (*Particles distribution in set in the Domain*) if the mass of fluid is huge: the results were not correct.

New in Version 2.2

- *Fluid particles* and *Diffuse particles* mesh types of meshers now allow the Preview option.
- There is a new type of mesher: *Volumetric OpenVDB*.
- New Domain property: Vorticity confinement
- New properties of the Domain: *Enable saving Fog-type VDB* and *Remove isolated particles*.
- New property of the meshers of type Fluid particles: *Lower temperature filter and the Higher temperature filter*.
- New properties of meshers: Silence Progress dialog and *Erosion of flames*.
- Multiple Filters.