PARSIS

A Particles System

for Daz Studio

V 1.0.2

By

ALVIN BÉMAR

TRADEMARK & COPYRIGHT NOTICE

Contents

Registering....1
Caution about uninstalling....1
INTRODUCTION....2
Basic Objects:....2
QuickStart....2
Basic Usage....3
Color of properties....3
CREATING OBJECTS....4
Source....5
Main settings group....5
Bouncing group....11
Gravity group....12
Kill particles group....12
Targeting group....14
Forces....16
Point force....16
Line force....17
Directional force....17
Torque force....18
Flow force....19
Internal force....19
Damping force....20
Force fields....20
Sinks....21
Obstacles....22
Mesher....23
How to use the materials with the mesher....27
ENGINE SETTINGS....28
....31
EFFECT OF THE SETTINGS....32
Dispersion of initial velocity....32
Blob geometric type....33
Any node geometry type....34
Forces....35
Force field....39
Targeting....40
Mesher....41
References....43
Licences:....44
Version 1.0.2....48
Version 1.0.1....48

PARSIS: A Particles System

Registering:

The plugin requieres to be registered in order to work. The process is the following:

1. On www.Daz3d.com website, go to *My Account.*
2. Click *Serial Numbers* and copy the serial number for the Parsis Product.
3. Next go into Daz Studio.
4. Go to *Help – About installed plugins* menu.
5. Scroll down to the PARSIS Product, paste the serial number and click OK, then restart Daz Studio to complete the process.
6. If PARSIS was correctly registered, you will see the *Create – New Parsis Object…* menu, and listed in the Windows – Panes (Tabs), the PARSIS one.

Caution about uninstalling:

If you have installed FLUIDOS II (any edition) and PARSIS plugins, and later you uninstall any of them, the other will be affected as they share some files. To solve the problem, uninstall both plugins and reinstall only the plugin you want to conserve.

As the content of both products are in separate folders, the content of one product are not affected by the uninstalling the content or the plugin files of the other.

INTRODUCTION

Basic Objects:

A) Parsis Source: This is the main. All particles get into the scene from a source. A scene can have multiple sources, each one with its own properties. Its particles can interact with all the Parsis objects in the scene (except another source's particles) or only with some of them. The source controls the production rate and speed of particles and their particular behavior. There is a maximum number of particles a source can produce in a simulation, the user can set this limit.

B) Obstacles: Any geometric object in the scene (except node instances) can be assigned as an obstacle. The obstacles, as their name indicates, can block the particles, by stopping or bouncing them, or only reduce their speed.

C) Forces: The Forces can control the velocity, and thus the trajectory of the particles by pushing or pulling them.

D) Force fields: These are geometric nodes that can trap or push out the particles. They can control the particle's trajectory very precisely. The user can choose any geometric object in the scene as a model for the force field.

E) Sinks: These objects eliminates any particle that goes inside them.

F) Mesher: This is the object that shows the particles in the scene. The user can choose between preview only or billboards (triangles) or instances nodes. The last ones allow the use of any geometric object (except another instance) as a geometric model for the particles.

G) Engine Settings: The general settings of the engine as the frames per second, frames to simulate, data folder, etc. can be set in the PARSIS pane, but they are not saved with the scene. On the other hand, the Engine Settings object can store the settings and it is saved with the scene.

QuickStart:

1. After registering the plugin, go to Daz Studio's main menu, select *Windows – Panes – PARSIS.* In the *Configuration* tab, go to the *Data folder* and browse a suitable folder for the baked files or the plugin. This folder will be memorized by Daz Studio for the next sessions.
2. Go to Daz Studio's main menu, select *Edit – Create – New Parsis Object….* Choose *New particles basic setup…*
3. The plugin will ask for the name of the source, the mesher, and the number of frames. You can simply click *Accept*.
4. When the plugin asks if you want to create an *Engine settings node*, click *No.*
5. Click the button *Start* in the *PARSIS* pane. After the plugin finishes, go to the *Objects* tab of the *PARSIS* pane and select the *PARSIS* Mesher in the *PARSIS objects in* the *scene* combo box.
6. Click the *Enable* property to set it *On* to see the particles.

Basic Usage:

1. Create the source or sources.

2. Create additional desired objects: obstacles, forces, force fields, sinks, or engine settings.

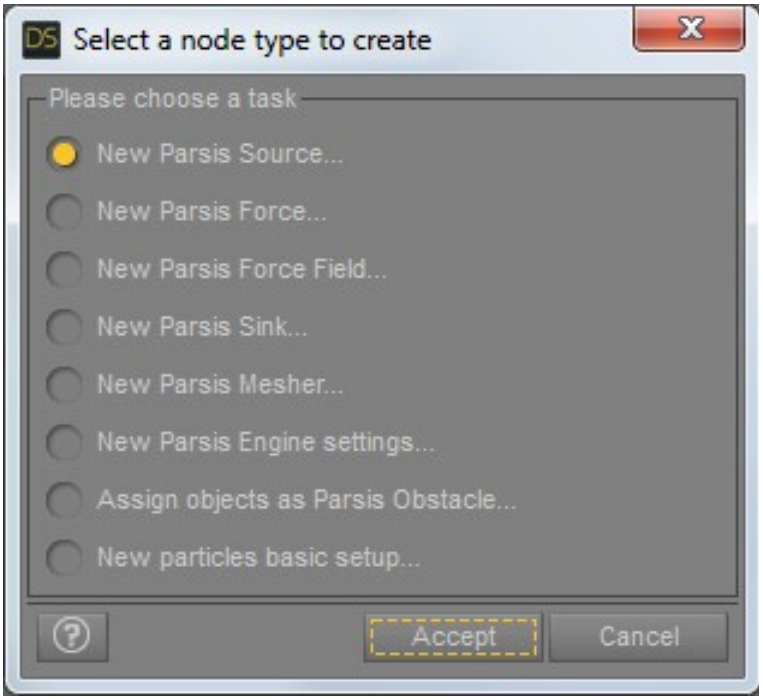
3. Set the data folder in the Engine settings object or the *PARSIS pane.* Run the simulation by clicking the *Start* button in the *PARSIS pane*.

4. Create a mesher and point to the same data folder of step 3. And enable it to see the particles in preview mode or using the desired mesh type.

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

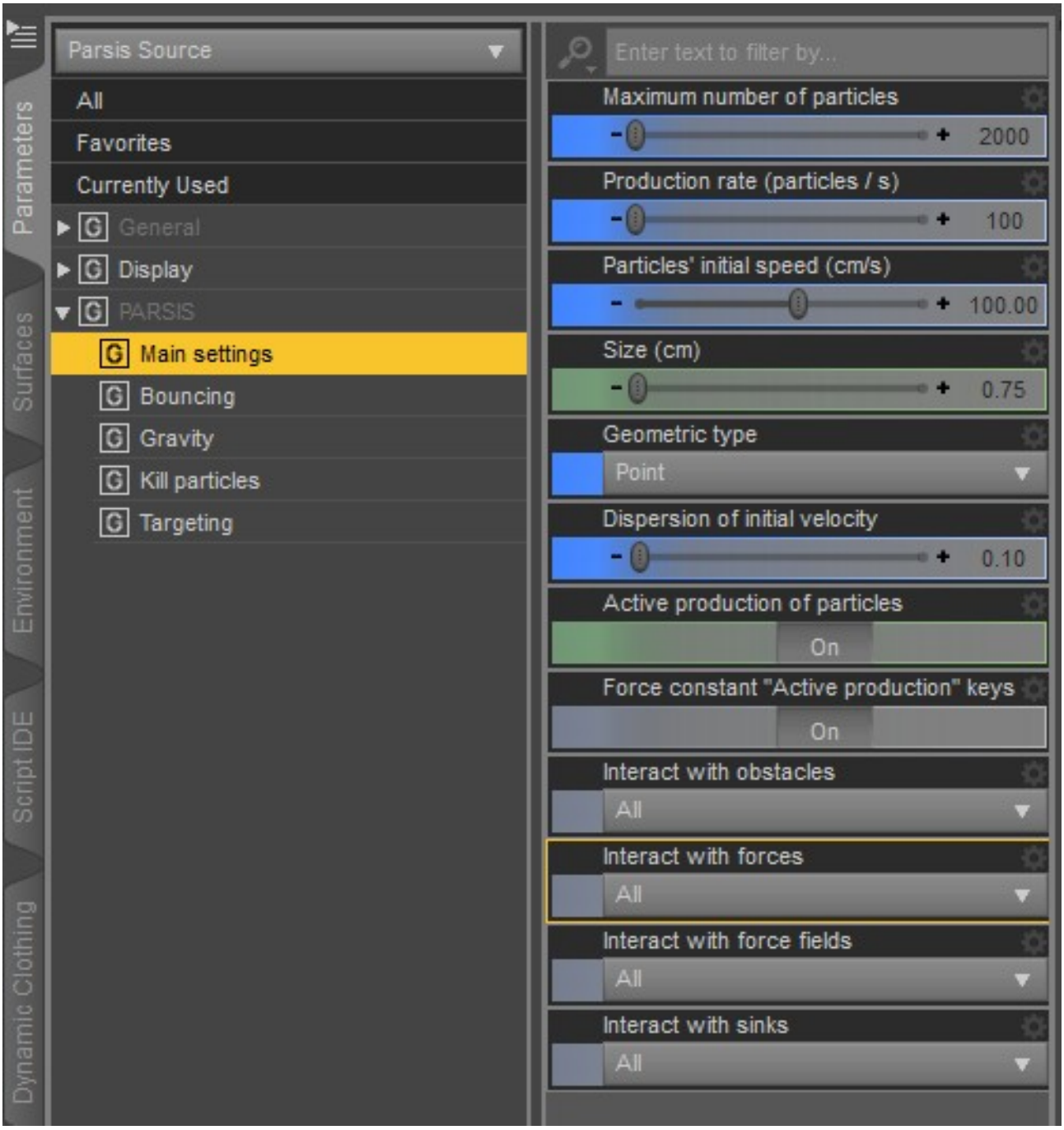
CREATING OBJECTS

To create *PARSIS* objects, go to the Daz Studio main menu, *Create – New Parsis object…*



You can add as many of the objects as you want in a scene.

***Source*:**



The PARSIS parameters of the sources are grouped into five sets: *Main settings*, *Bouncing*, *Gravity*, *Kill particles* and *Targeting.*

***Main settings group*:**

Maximum number of particles: This is the maximum number of particles of the source that will exist at a time. The source will not emit any particle unless the number of particles decreases (by destroying them by a sink, aging, etc.)

Production rate: This is the number of particles that will be emitted per second. It's animatable.

Particles' initial speed: This is the speed of the particles in centimeters per second at the moment of their emission. It's animatable.

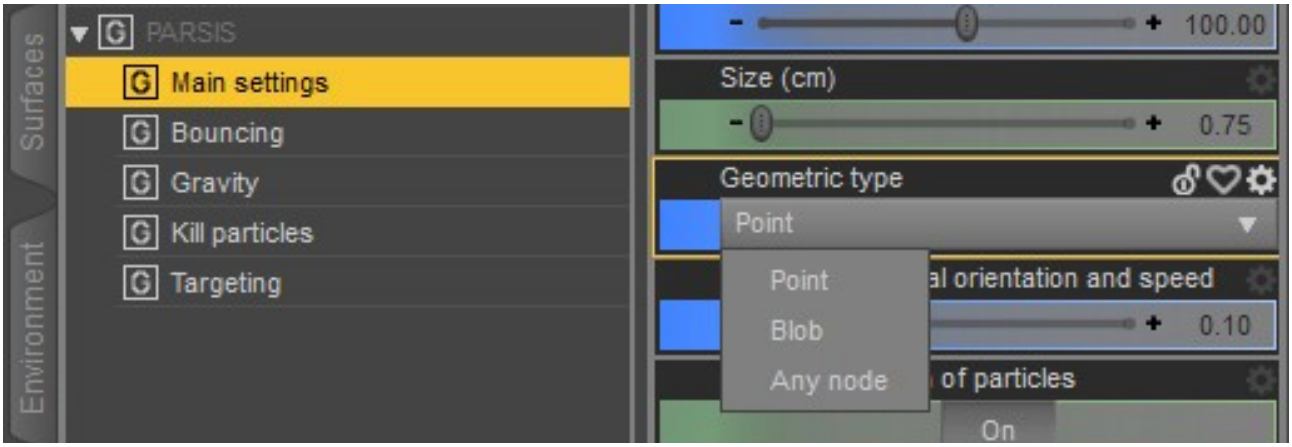
Particle size: This is the size of the particles in centimeters. This value is an approximation as the exact size depends on the object chosen as a representation of the particle. It's animatable.

Dispersion of initial velocity: This property controls the variability of velocity (the speed and orientation) of the particles at their emission. The value is the standard deviation. It's animatable.

Active production of particles: If this property is set to *Off*, the emission of particles is suspended. It's animatable.

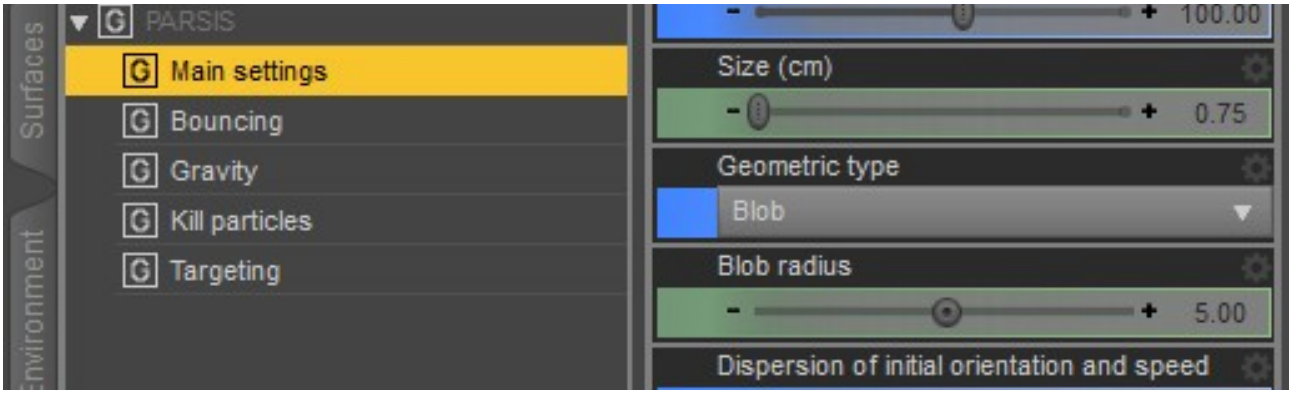
Force constant "Active production" keys: If this property is set to *On*, the keyframes of *Activate* are treated as a constant interpolation: if the user adds a first keyframe as *On* and the next as *Off* for the *Activate* parameter, the plugin will add an *Off* key just a frame before the last one. This way, the source is active until the user sets it to *Off*.

Geometric type: This controls the form of the emitter. It can be a *point*, *blob*, or *any node*.



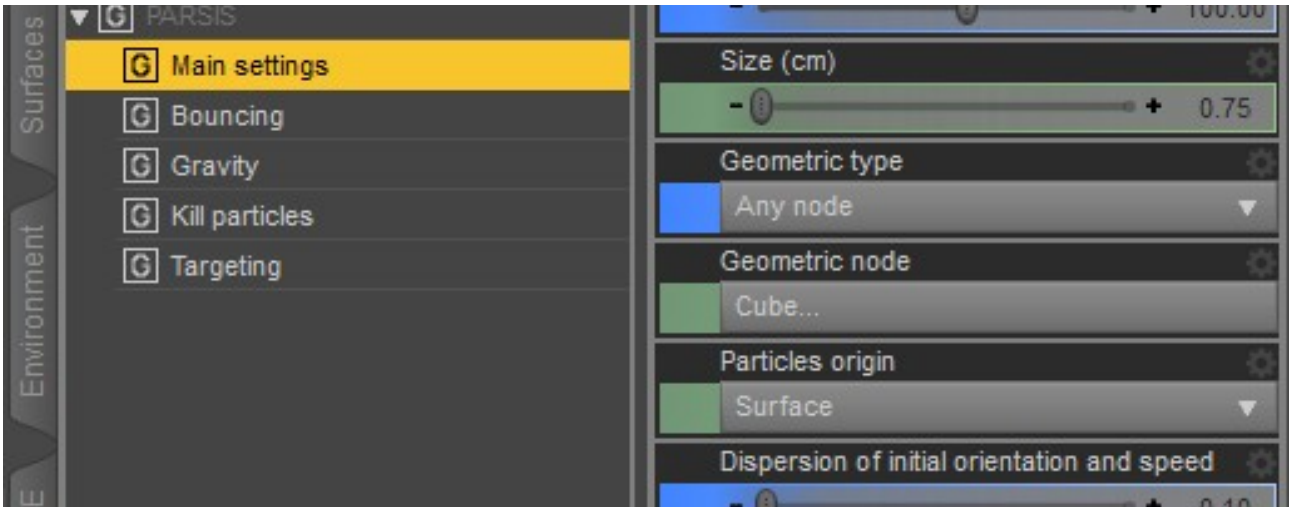
If the user selects *Point* (the default), the particles will be emitted from a point in the space (the position and orientation are controlled by the user using the control properties in the usual *Transform* group, or by moving and rotation the source in the viewport as usual.

If the user selects *Blob*, it appears an extra option, the *Blob radius.*



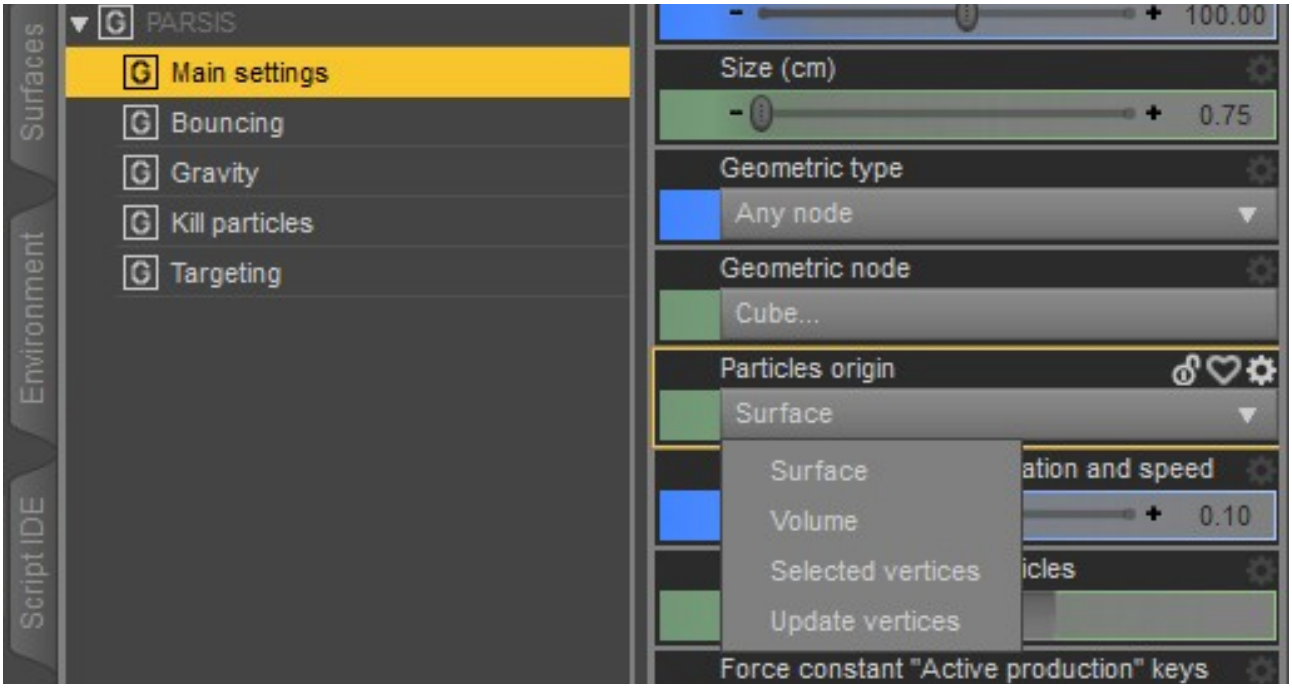
The *blob* type sources emit the particles from the inside of a sphere of radius controlled by *Blob radius* (in centimeters). The position origin and speed are controlled by the *Dispersion of initial orientation and speed*, but not the orientation, as this is parallel to the radius.

The *Any node* type allows choosing any geometric node in the scene.



Geometric node: This points to the geometric node that will be the particles' emitter. The user can select the emitter by parenting the source to a valid geometric node or by clicking this property to select a node. The node should have a closed mesh, but some open meshes can work as well.

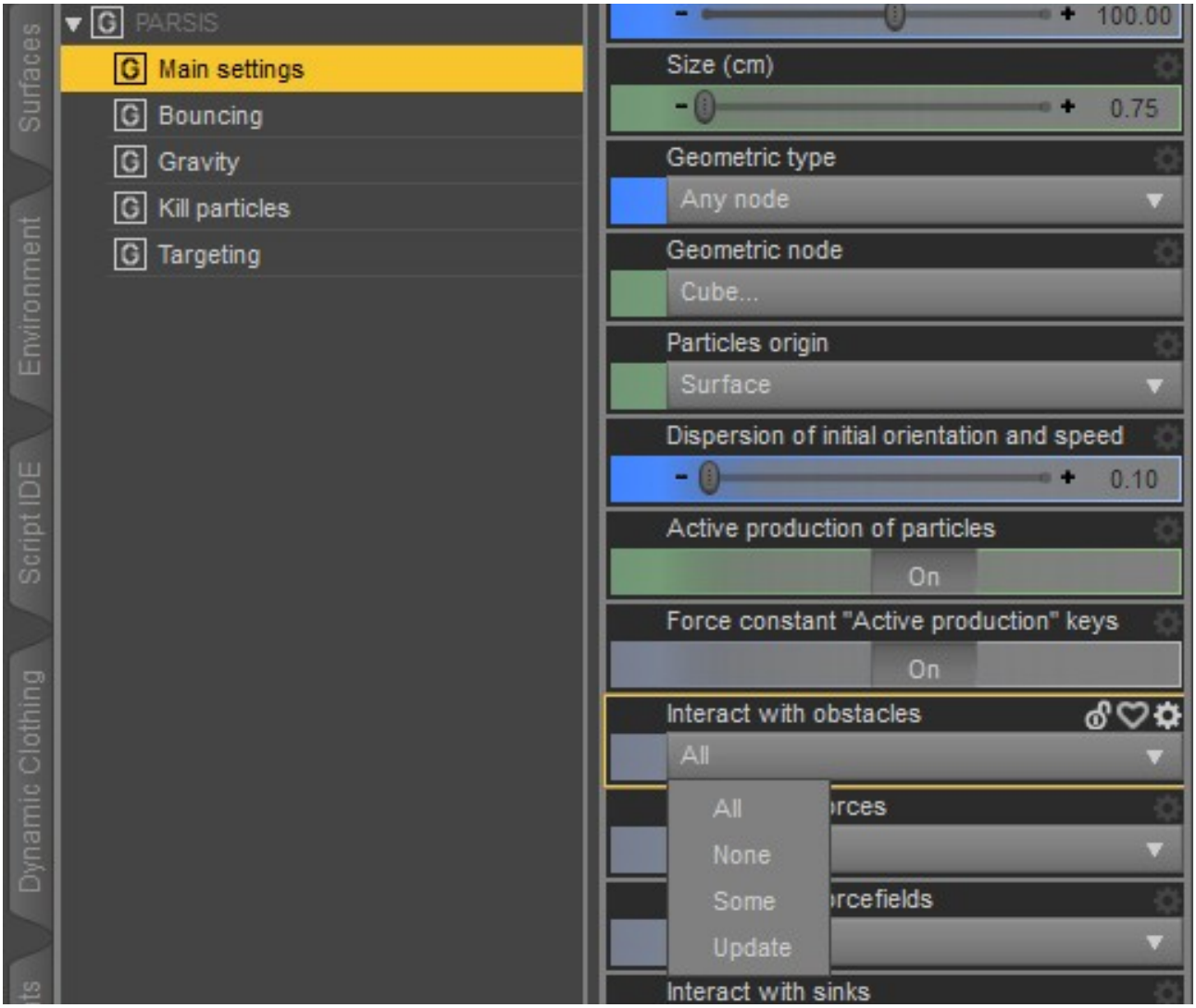
Particles origin: The origin of the particles in space in a geometric node can be: *Surface*, *Volume*, or *Selected vertices.*



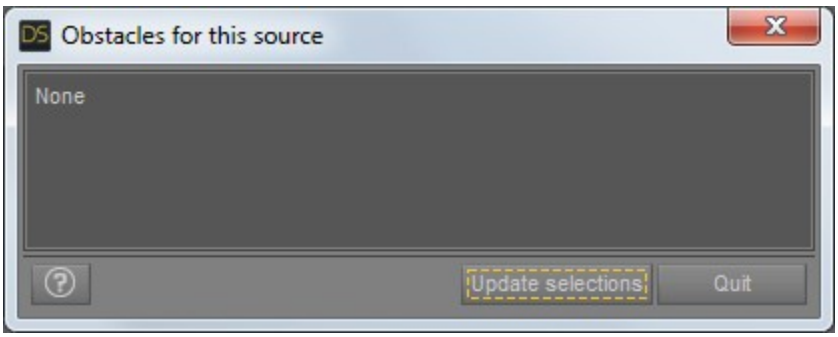
By choosing *Surface*, the particles will be emitted from the surface of the geometric node. If *Volume* is selected, the particles will be emitted from all the space inside the geometric node.

But if *Selected vertices* is chosen, the particles will be emitted around the selected vertices in the geometric node. The vertices can be selected using the Daz Studio *Geometric Editor Tool* (using the option *Vertex selection*, not *Polygon selection.* The vertices *must be* selected before choosing this option, but the vertices can be changed after that. The option *Update vertices* must be applied after changes in the selection of vertices to synchronize the changes with PARSIS.

Interact with obstacles: This property controls what *obstacles* in the scene can interact with the particles emitted by *this specific* source.



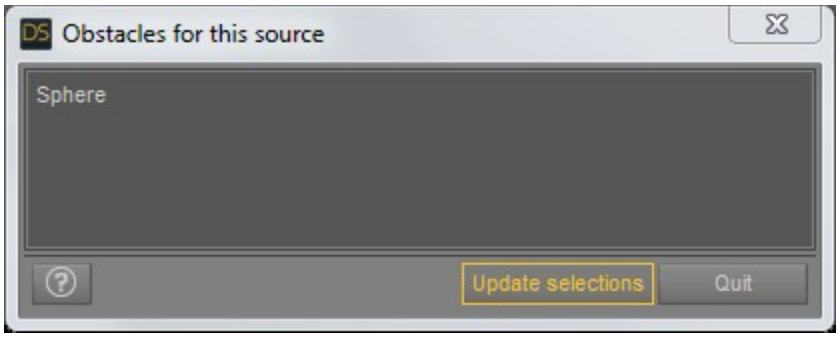
Selecting *All* (the default) will allow all the objects in the scene to interact with the particles. Selecting *None* will avoid any object to interact with the particles. If *Some* is selected, a box will appear showing the currently allowed obstacles (or showing *None*)



Clicking on *Update selections* will open a dialog for selecting the obstacles. The list will show only the objects assigned as obstacles (see *Obstacles*). The user can select more than one obstacle.

If the checkbox *Unassign not selected nodes* is checked, only the objects selected in this dialog will be allowed; any previously allowed obstacles will be ignored.

When the user clicks on *Accept*, the dialog closes and the first box shows the currently allowed obstacles for this source.



To change the allowed obstacles, the user must select the option *Update* in the menu. This way, the dialogs for selection will be opened again.

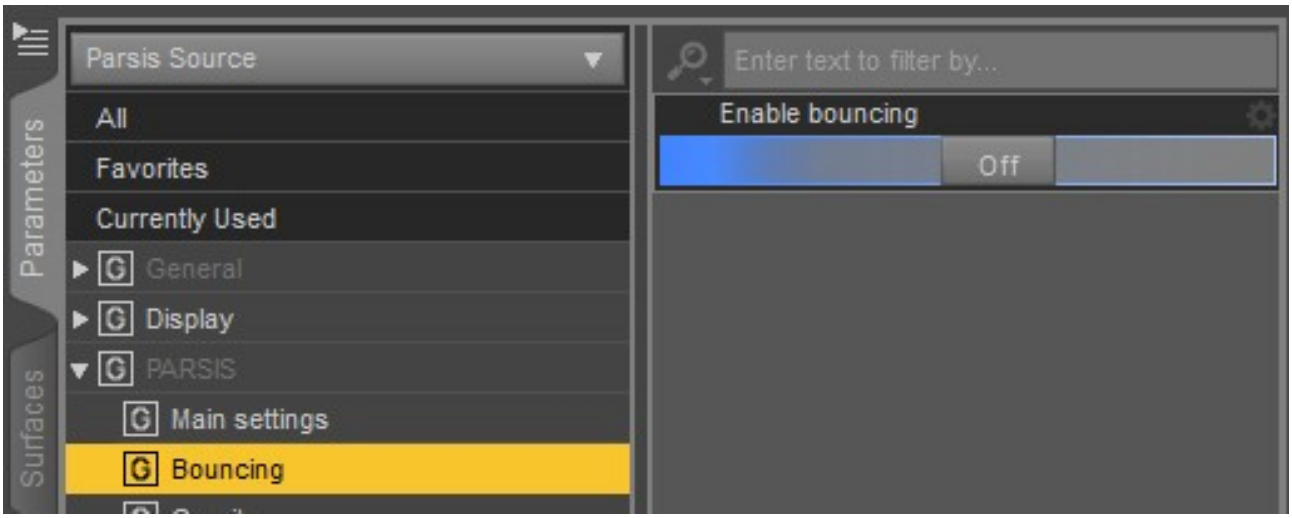
Interact with forces: This property controls what *forces* in the scene can interact with the particles emitted by *this specific* source. The procedure for choosing is as the *Interact with obstacles* one.

Interact with force fields: This property controls what *force fields* in the scene can interact with the particles emitted by *this specific* source. The procedure for choosing is as *Interact with obstacles* one.

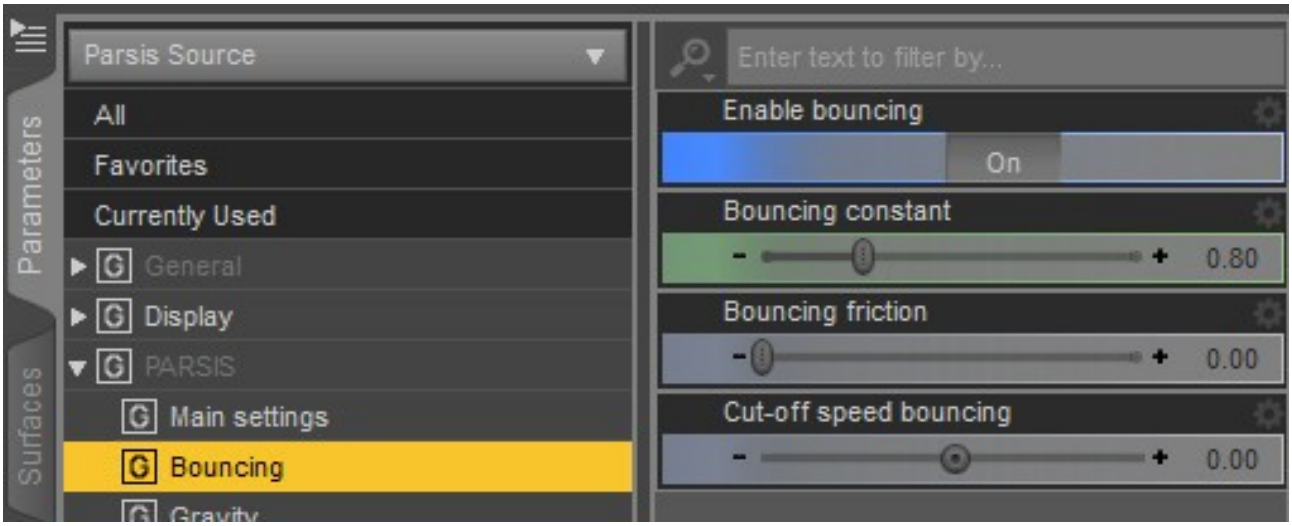
Interact with sinks: This property controls what *sinks* in the scene can interact with the particles emitted by *this specific* source. The procedure for choosing is as *Interact with obstacles* one.

Bouncing group:

Enable bouncing: if this is selected, the particles will bounce against any allowed obstacle.



When bouncing is enabled, three extra properties will be shown:

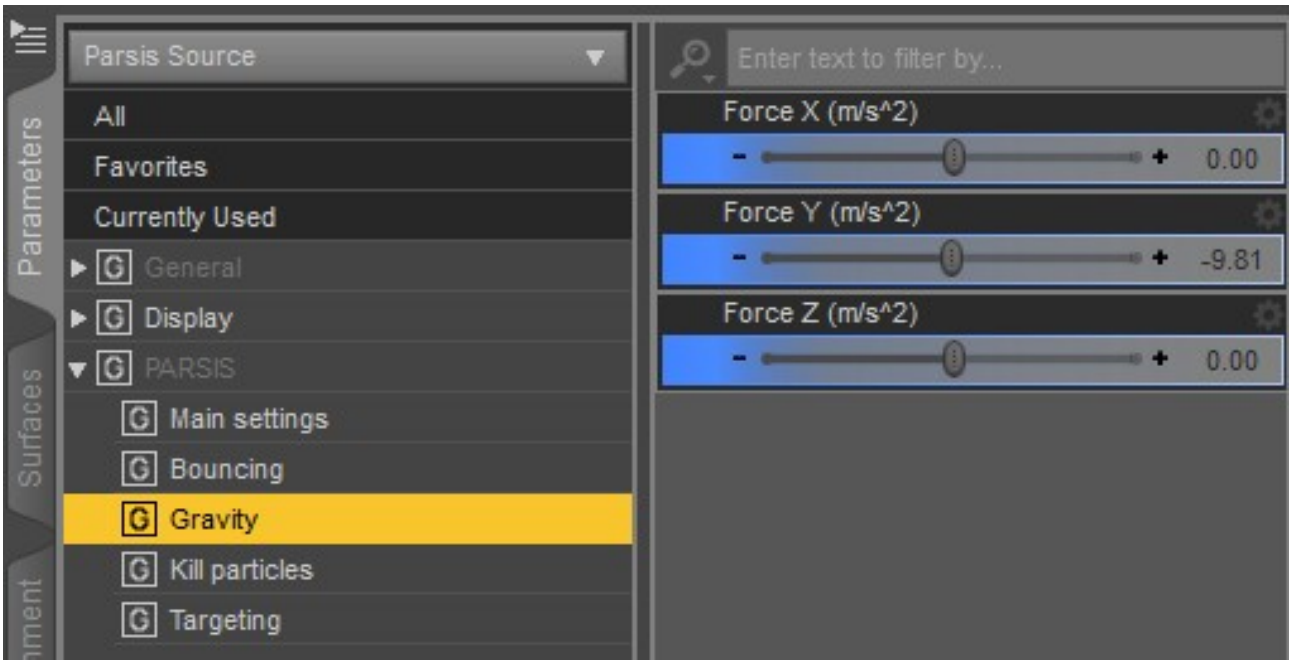


Bouncing constant: This property controls the degree of bouncing. Zero constant causes no bouncing at all. A value of 1.0 causes a perfect bouncing, the speed will not be reduced in any of the bounces. A value over 1.0 will produce an increased speed at each bounce. It's animatable.

Bouncing friction: This controls the friction of the bouncing. The higher the friction, the lower the speed after a collision. It's animatable.

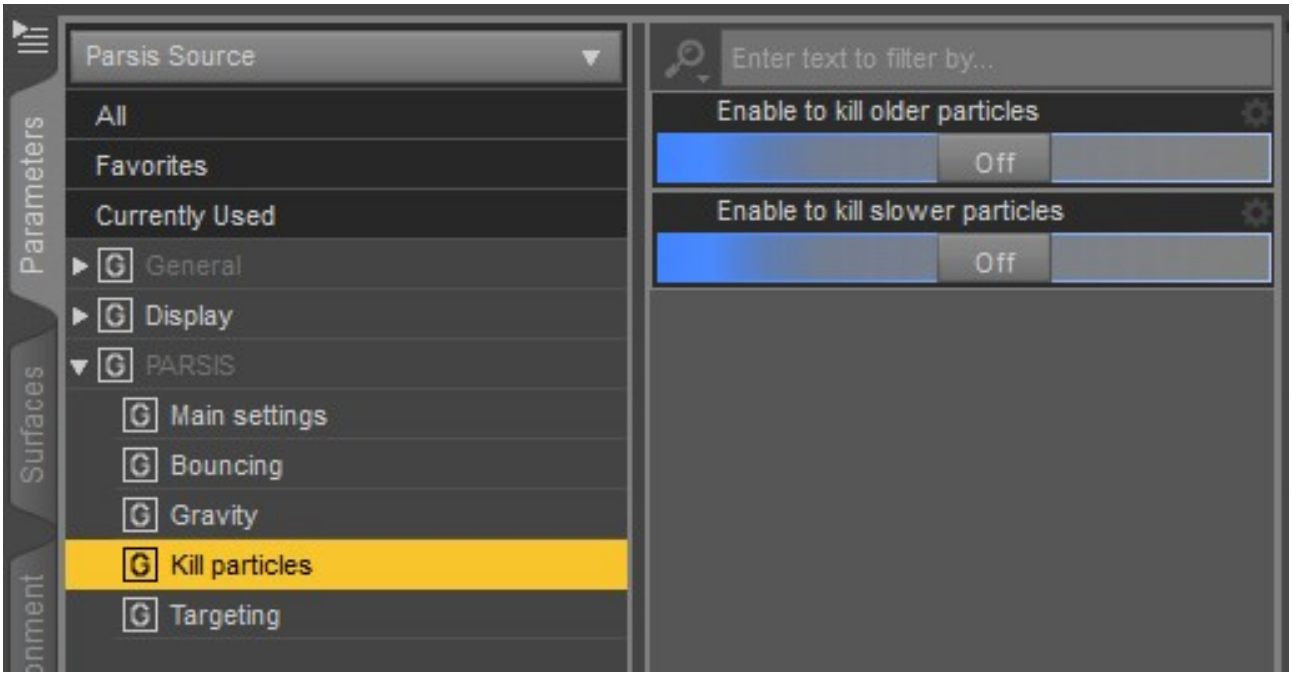
Cut-off speed bouncing: This is the low limit speed for the friction to act. Lower speeds will not be reduced by friction. It's animatable.

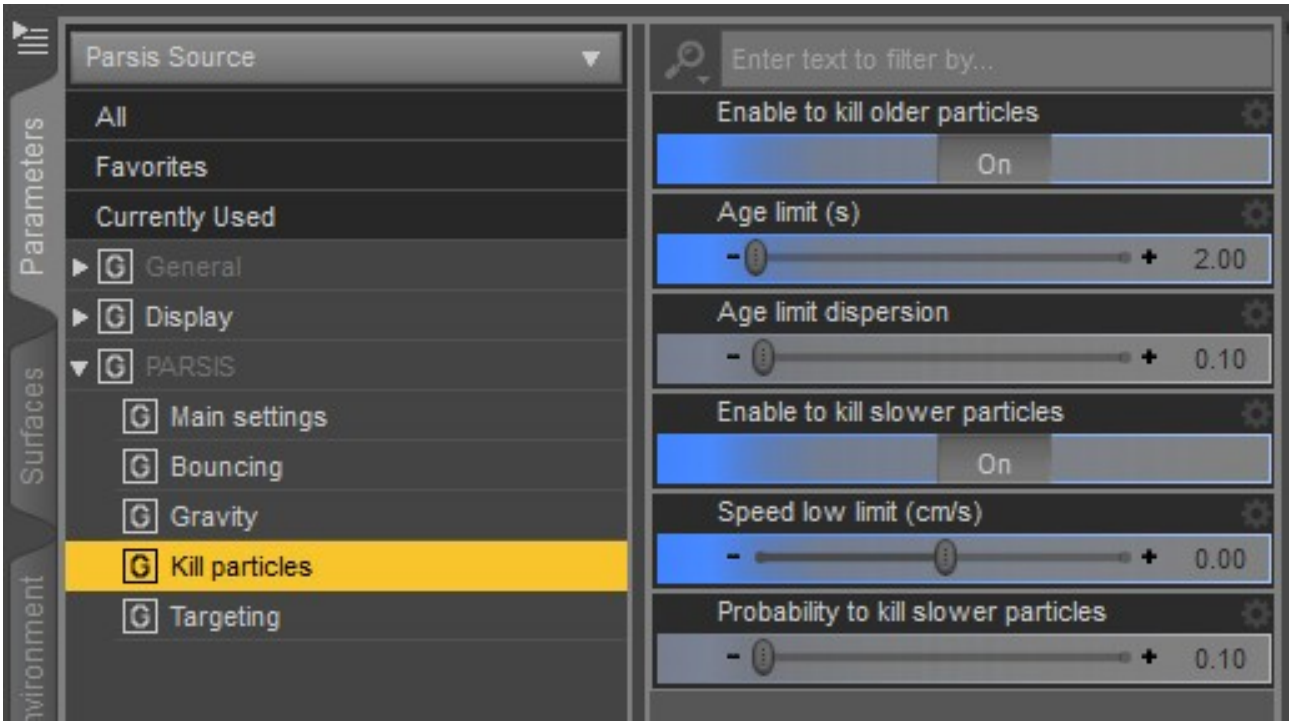
Gravity group:



Force X, *Force Y,* and *Force Z* are the properties that control the gravity force. The values are in m/s^2.

Kill particles group:





Enable to kill older particles: This property enables the elimination of a particle of this specific source after some time (*Age limit*) from its emission.

Age limit: This the time after emission of a particle before is eliminated from the scene. It's animatable.

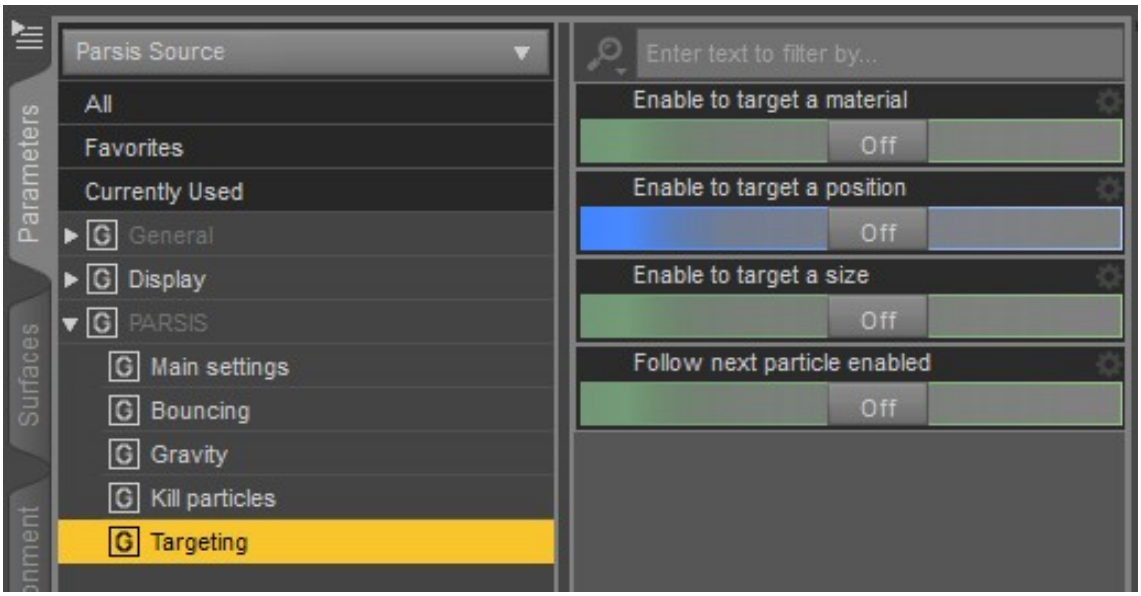
Age limit dispersion: This the variability of the *age limit* of the particles. It's the standard deviation around the *age limit*. It's animatable.

Enable to kill slower particles: This property enables the elimination of particles of this specific source when their speed is below a certain limit (*Speed low limit*).

Speed low limit: This is the minimum speed of a particle. Below this, the particle is eliminated. It's animatable.

Probability to kill slower particles: This the variability of the *Speed low limit* of the particles. It's the standard deviation around the *Speed low limit*. It's animatable.

***Targeting group**:*



Enable to target a material: This enables the particles to "target a material". That is, a particle will start with a material (or surface) at emission and will be changing it along time until a certain material (the target) is reached. The materials are identified as numbers from 0 to 255.

Initial material: This is the number representing the initial material of particles.

Target material: This is the number representing the last material of particles.

Time to reach the target material (s): This is the time, in seconds, to take a particle from the initial material to the target material.

Enable to target a position: This enables the particles to try to reach a position in the scene, the target position. The rate at which the particles reach the position depends on the existing forces in the scene (including gravity). The more intense are the forces, the higher the rate.

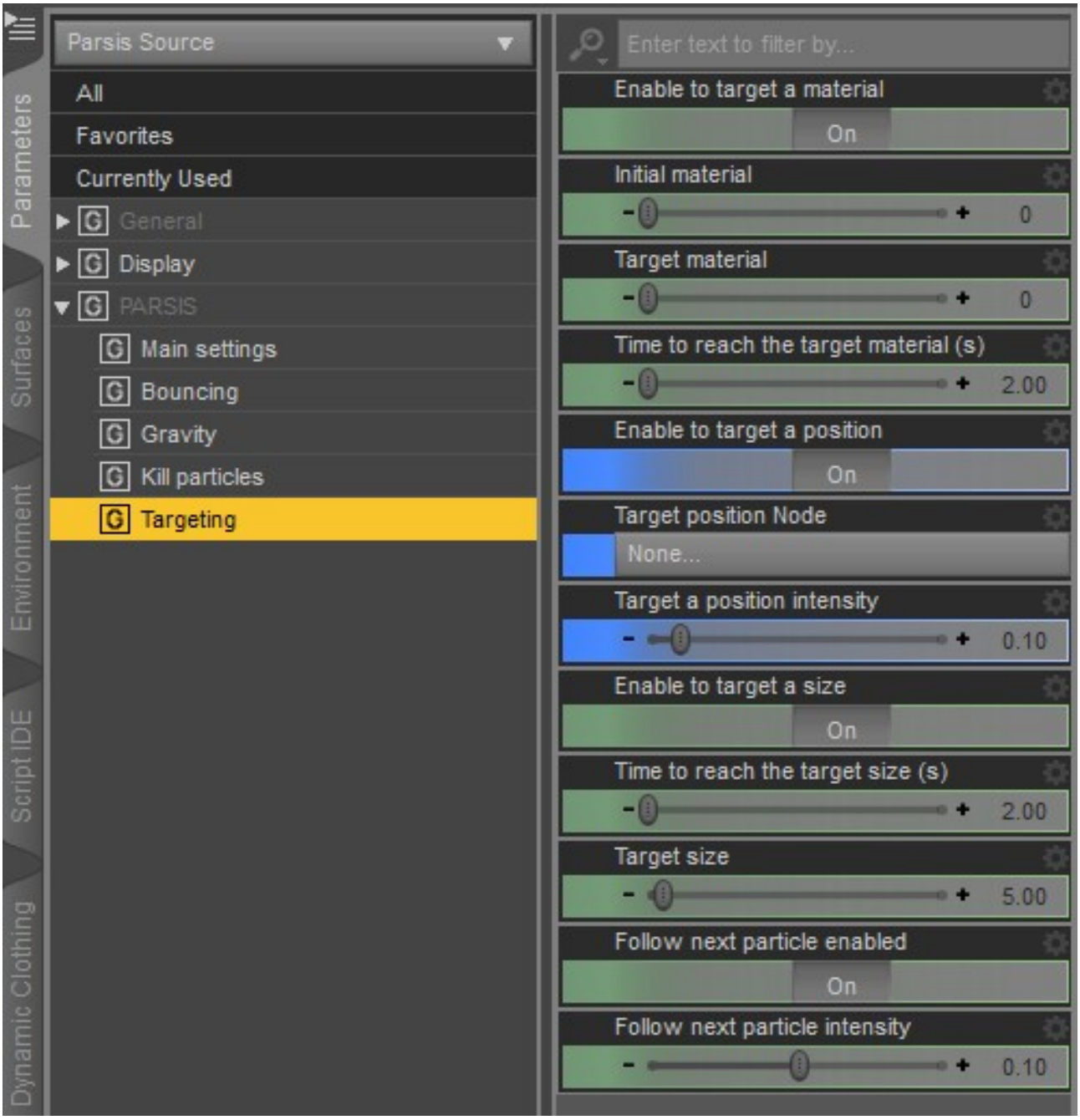
Target position node: This is the node whose position is the target for the particles. It could be any node, but it is recommended to use a null node to get a more clear position.

Target position intensity: This, in combination with forces, controls the rate at which the target is reached.

Enable to target a size: This property enables the particles to reach a size.

Time to reach the target size (s): This is the time, in seconds, that needs the particle to reach a certain size (the target size)

Target size: This is the final target size of the particle. The initial size is the one in the *Main settings group*.

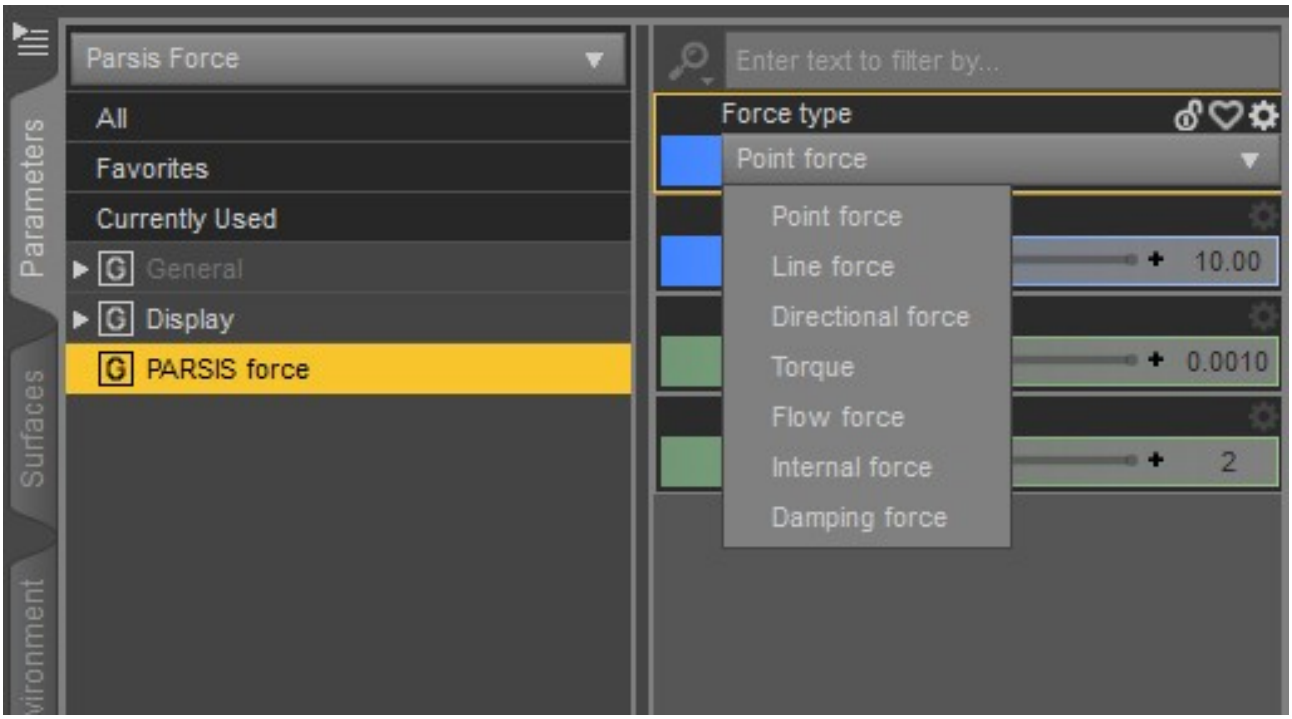


Follow next particle enabled: This causes each particle (except the first emitted one) to try to follow the next particle in order of emission.

Follow next particle intensity: This is the intensity of the following of the next particle.

Forces:

Force type: this property is used to select the type of force.



Point force:

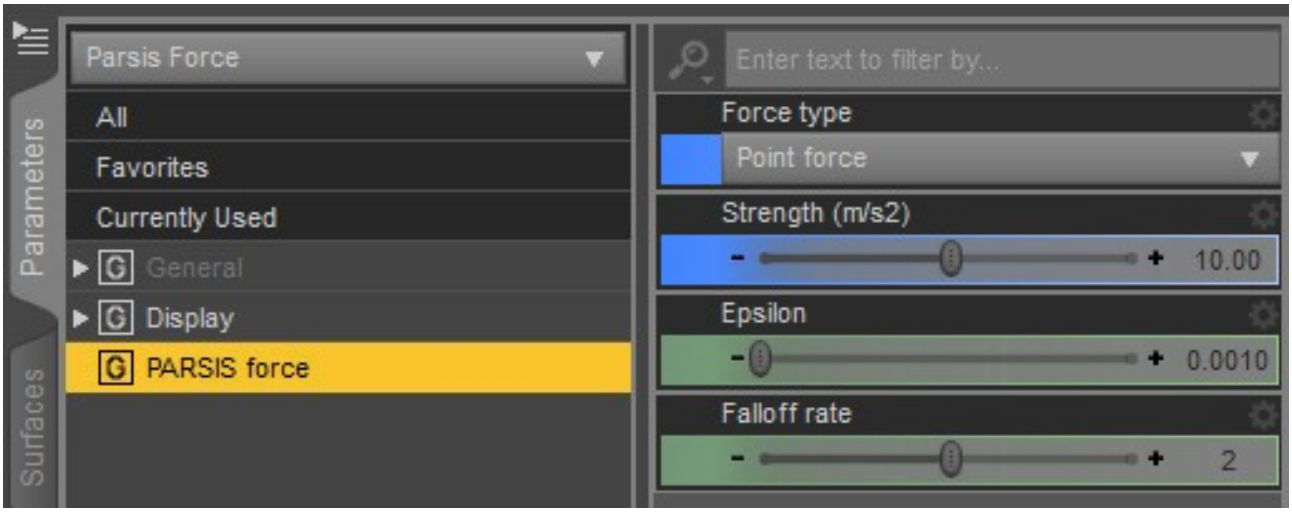
This is a force that extends its action from a point in space.

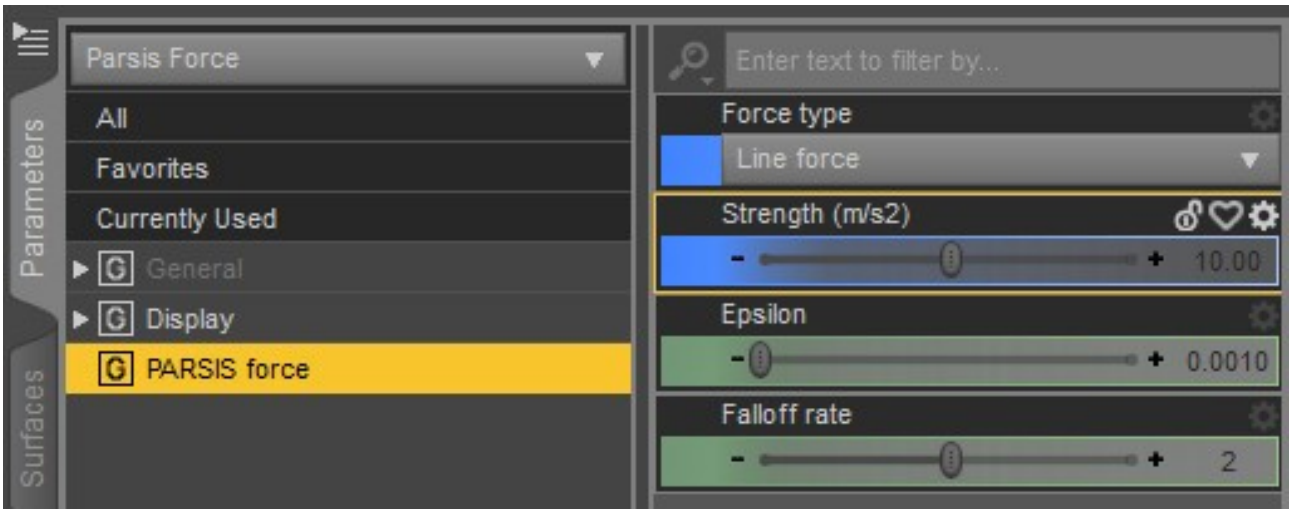
Strength: Magnitude of the force in m/s^2

Epsilon: A parameter that dampens the decay so the force never goes to infinite around its center.

Falloff rate: It is the rate of decay of the force, as an exponent. The higher, the faster the force decrease with the distance from the center.

Strength, *Epsilon*, and *Falloff rate* are animatables.

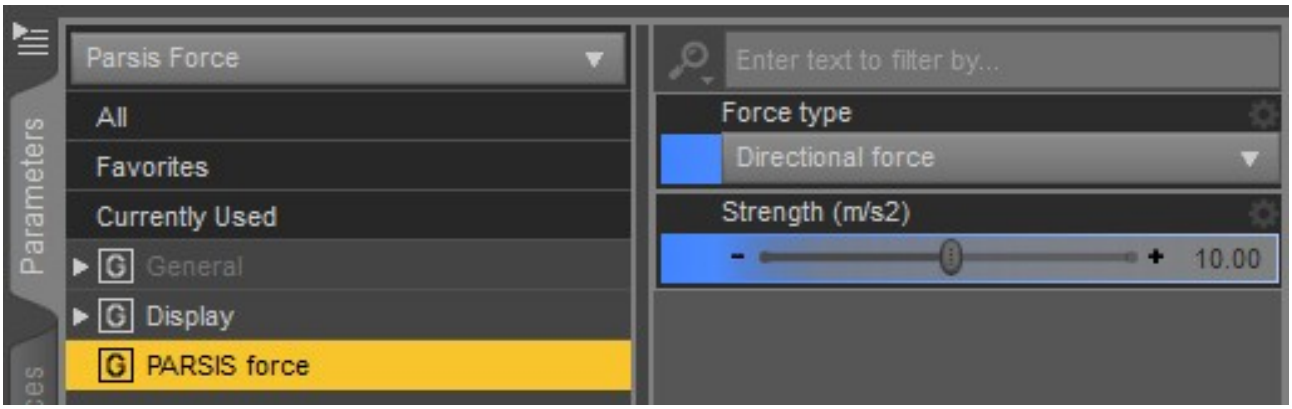




Line force:

This is a force that extends its action from a line in space.

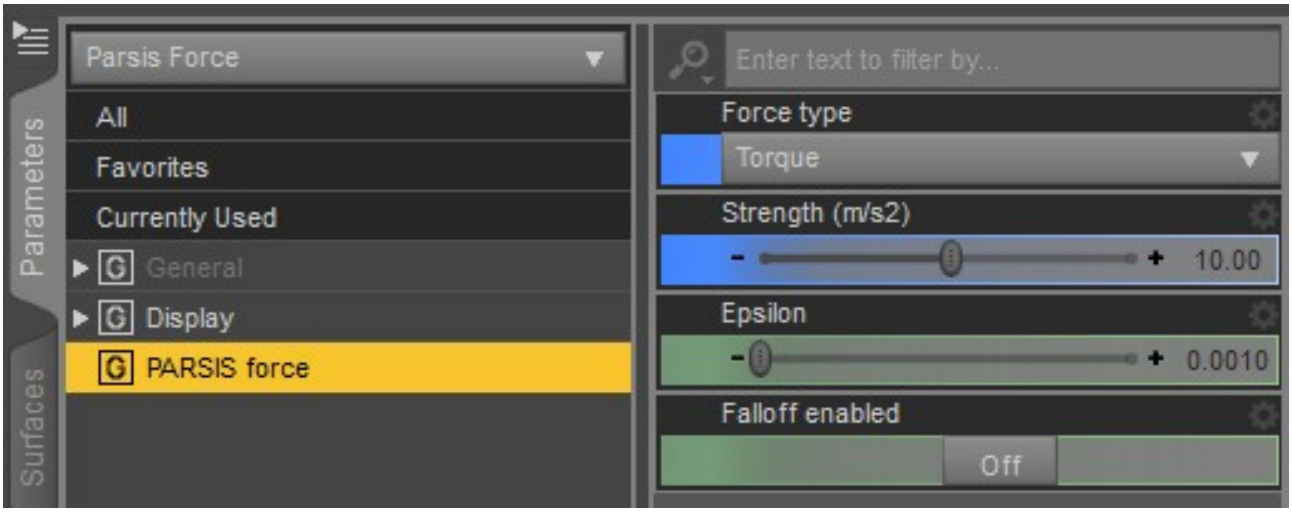
Strength, *Epsilon*, and *Falloff rate* have the same function as in *Point force*.



Directional force:

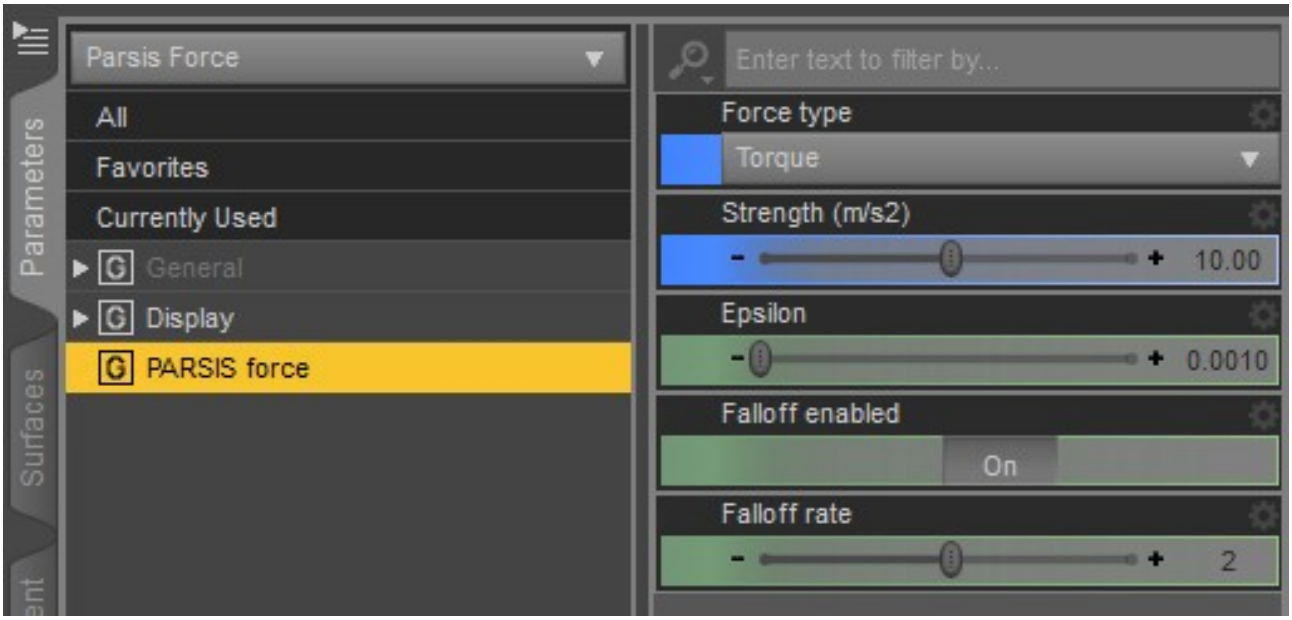
This is a force that extends its action in all space without any change.

Strength: the constant magnitude of the force.

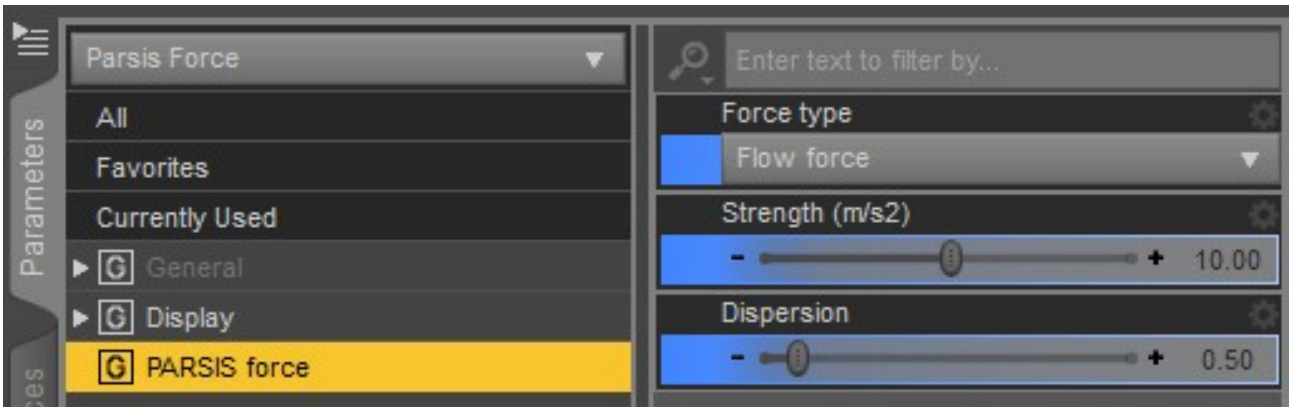


Torque force:

This force causes the particles to circle the central line of force. If *Falloff enabled* is set to Off (the default), the force will not decrease with distance.



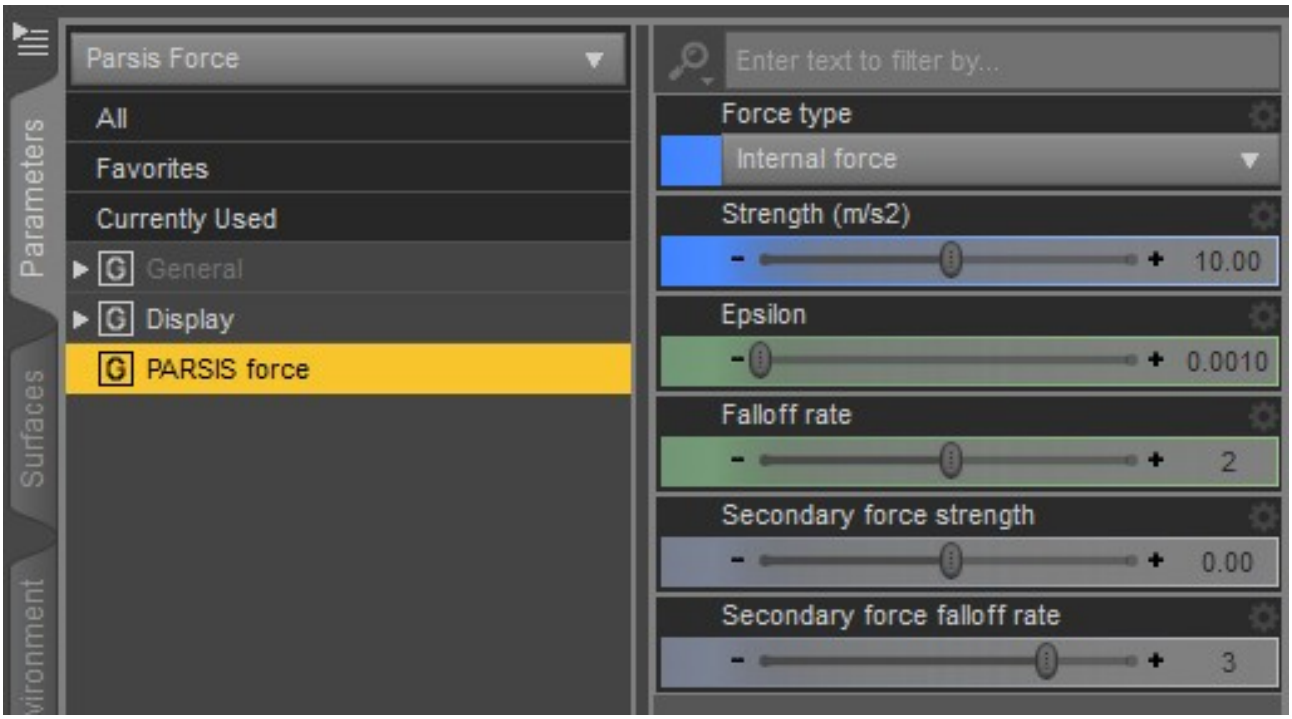
Strength, *Epsilon*, and *Falloff rate* have the same function as in *Point force*.



Flow force:

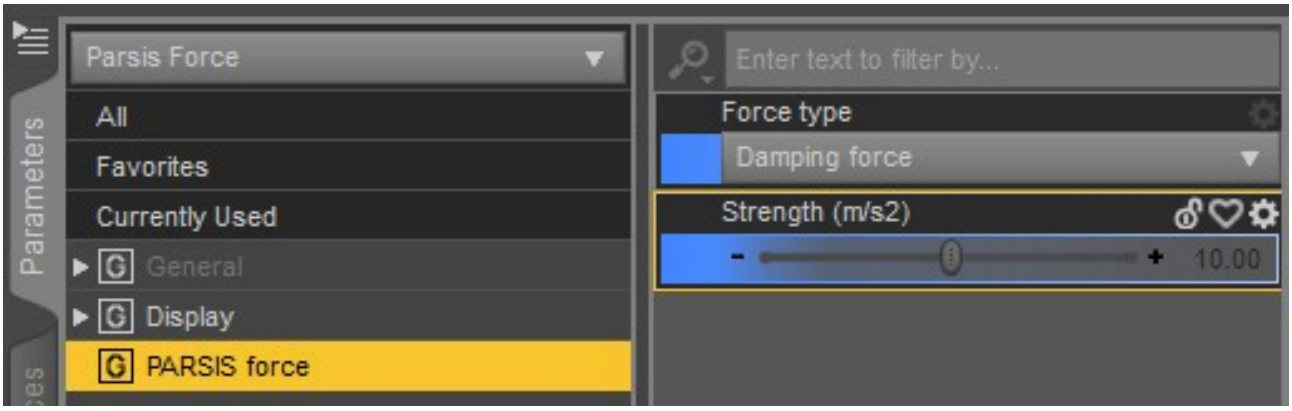
This force is similar to directional force, but its intensity and direction randomly vary over time and space.

Strength: This is the intensity of the force. The *Dispersion* is the magnitude of variation.



Internal force:

This is the force exerted between the particles. In fact, this PARSIS force has two components: one can be attracting and the other a repulsing one, for example, with different decay rates. They share the same *epsilon*. The secondary force is optional.



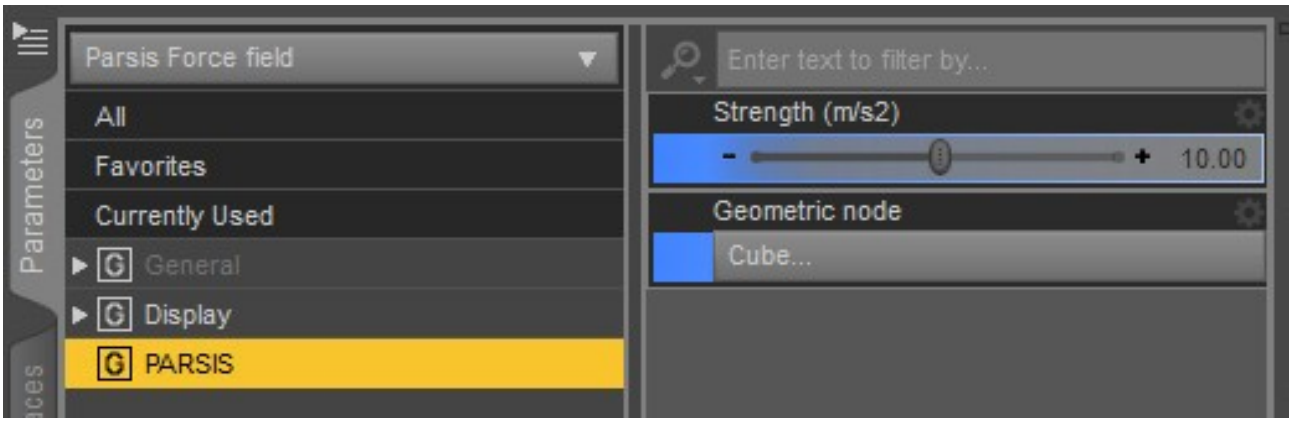
Damping force:

This force is opposed to the velocity of particles. The higher the velocity, the higher the force in the opposite direction.

The *Strength* is the magnitude of the opposition.

***Force fields*:**

The PARSIS force fields are forces confined to some region in space controlled by a geometric node.

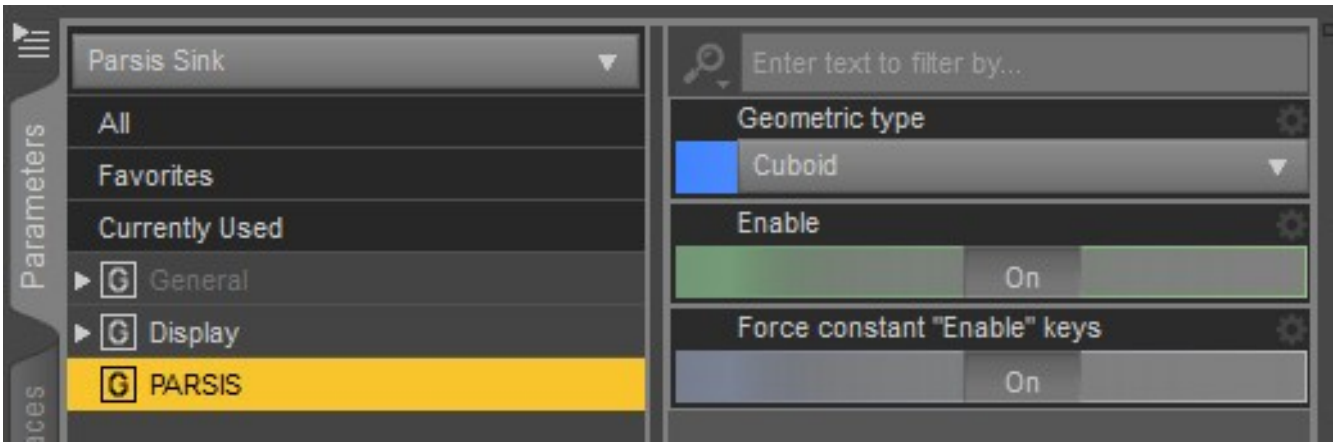


Strength: Magnitude of the force in m/s^2

Geometric node: This is the node where the force acts. It can be any geometric node in the scene. By choosing a geometric node, the user can click on the button of the property or simply parent the Force field to the desired node. The node should have a closed mesh, but some open meshes can work as well. Any force field without a node will be ignored in the simulation.

***Sinks*:**

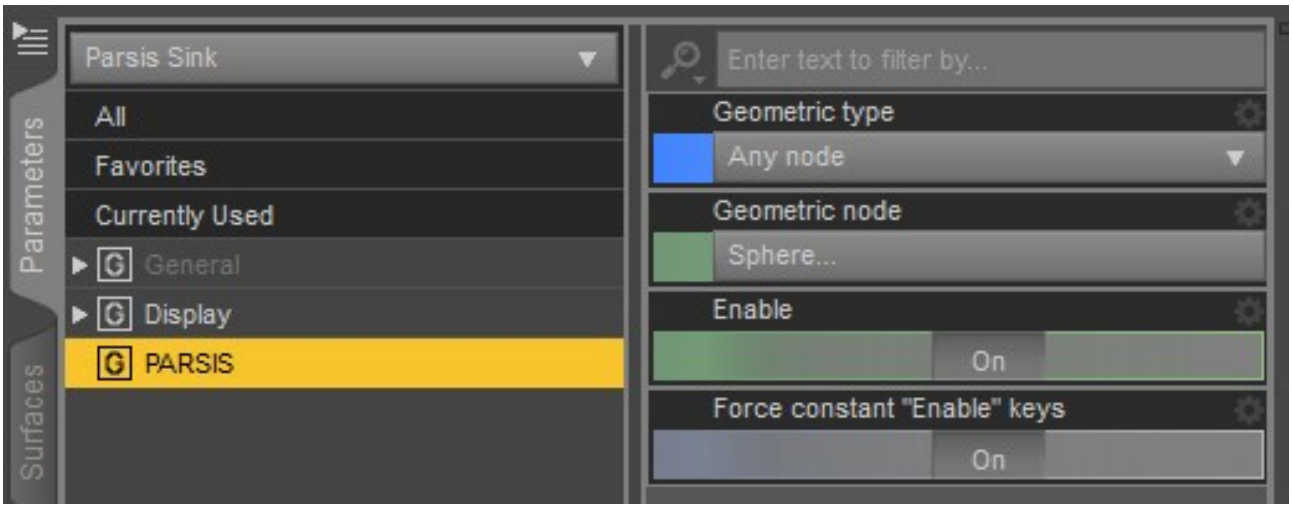
The sinks are regions in space where the particles disappear.



Geometric type: The sink region of action is delimited by a geometric object. It can be a resizable cuboid or a geometric node chosen by the user. The node should have a closed mesh, but some open meshes can work as well.

Enable: If this property is set to *Off*, the particles can pass through the sink without elimination. It's animatable.

Force constant "Enable" keys: If this property is set to *On*, the keyframes of *Activate* are treated as a constant interpolation: if the user adds a first keyframe 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 sink is active until the user sets it to *Off*.

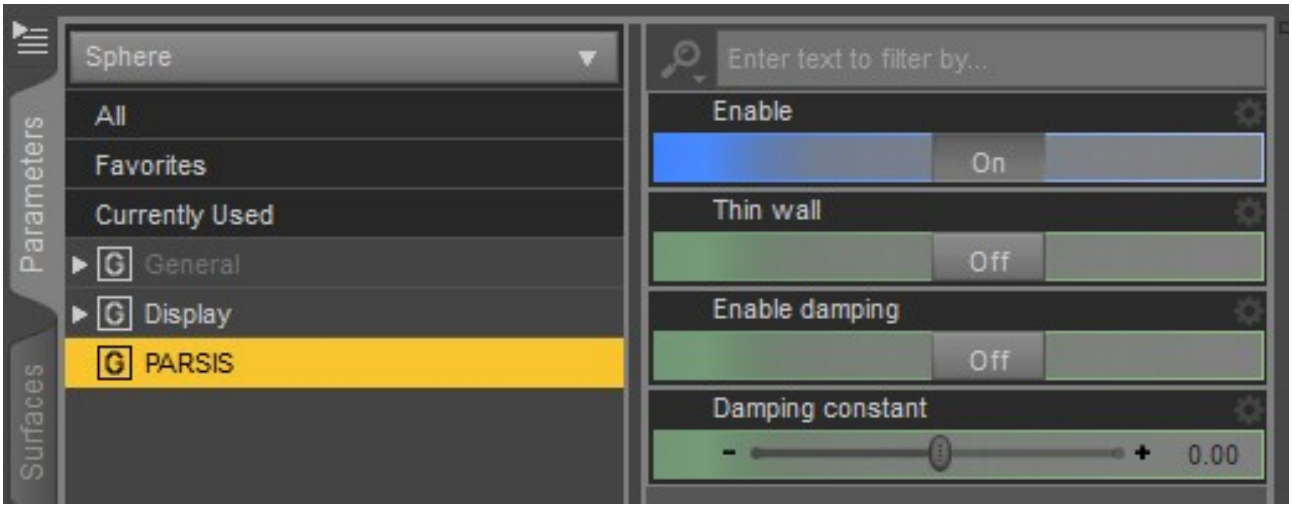


When *Any node* is chosen as a *Geometric type*, it appears the property *Geometric node*.

Geometric node: This can be any geometric node in the scene. By choosing a geometric node, the user can click on the button of the property or simply parent the Force field to the desired node. The node should have a closed mesh, but some open meshes can work as well.

Obstacles:

The obstacles can be any geometric node in the scene. They must be assigned (Using the *Assign Parsis Obstacles* option in the *Create a new Parsis Object* dialog)



Any valid node will have a PARSIS properties group after assignation.

Enable: This property can be used to disable an object without going to the *Create - Parsis object* menu.

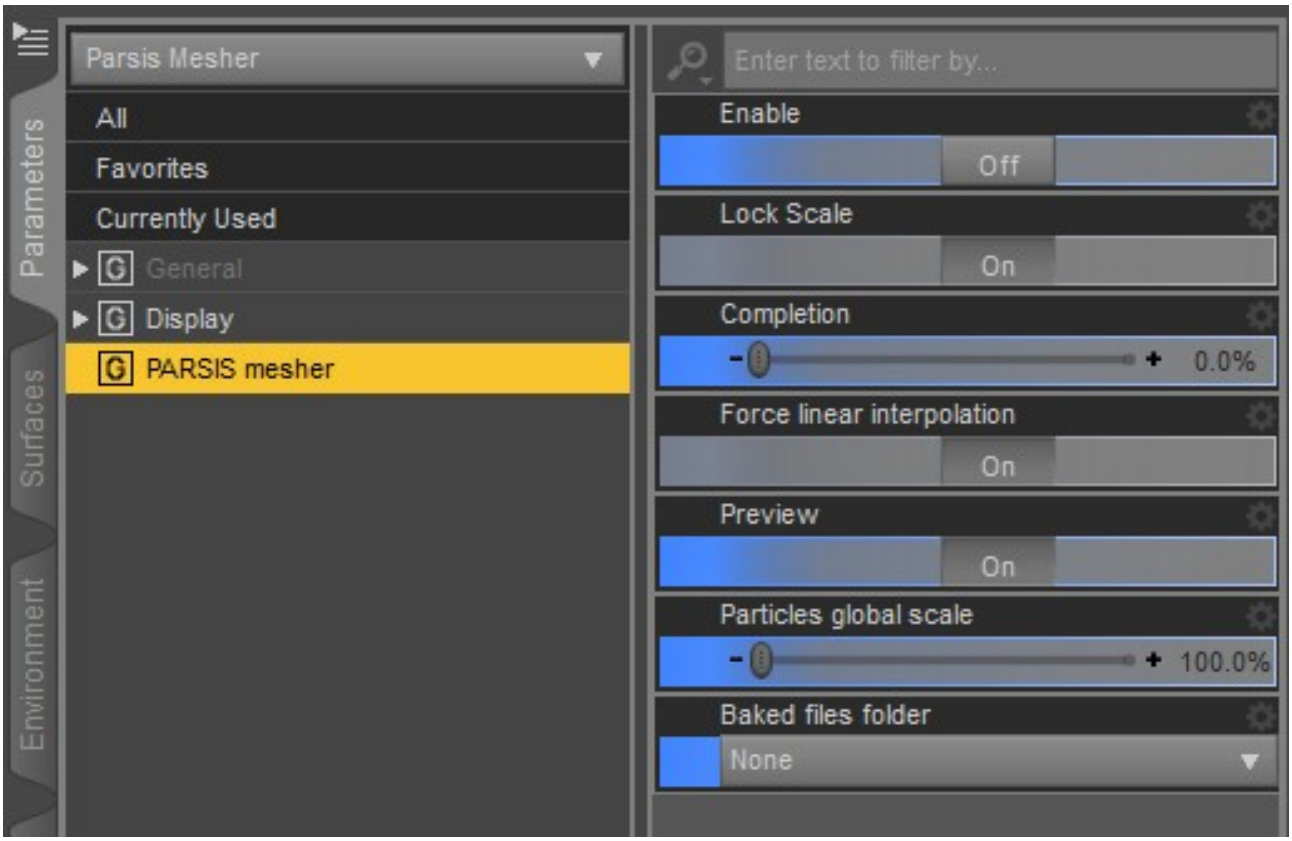
Thin wall: When the geometry of the object is not a closed mesh, this property could be set *On* to use the object as an obstacle. However, this is less exact than using a closed mesh.

Enable damping: If this property is *On,* the obstacle behaves as a “thick medium” that allows the pass of the particles but reduces their speeds. This effect usually achieves better results than the *Damping* force.

Damping constant: This property controls the damping of the particles by the obstacle.

***Mesher*:**

The mesher is the object that displays the particles after simulation.



Enable: this button synchronizes the mesher with the *Data folder* and allows the user to see the results of the simulation.

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 access to the scaling parameters

Completion: controls the animation of the particles simulation. Set it to 0% at the beginning of your animation, and at 100% at the end. It's animatable.

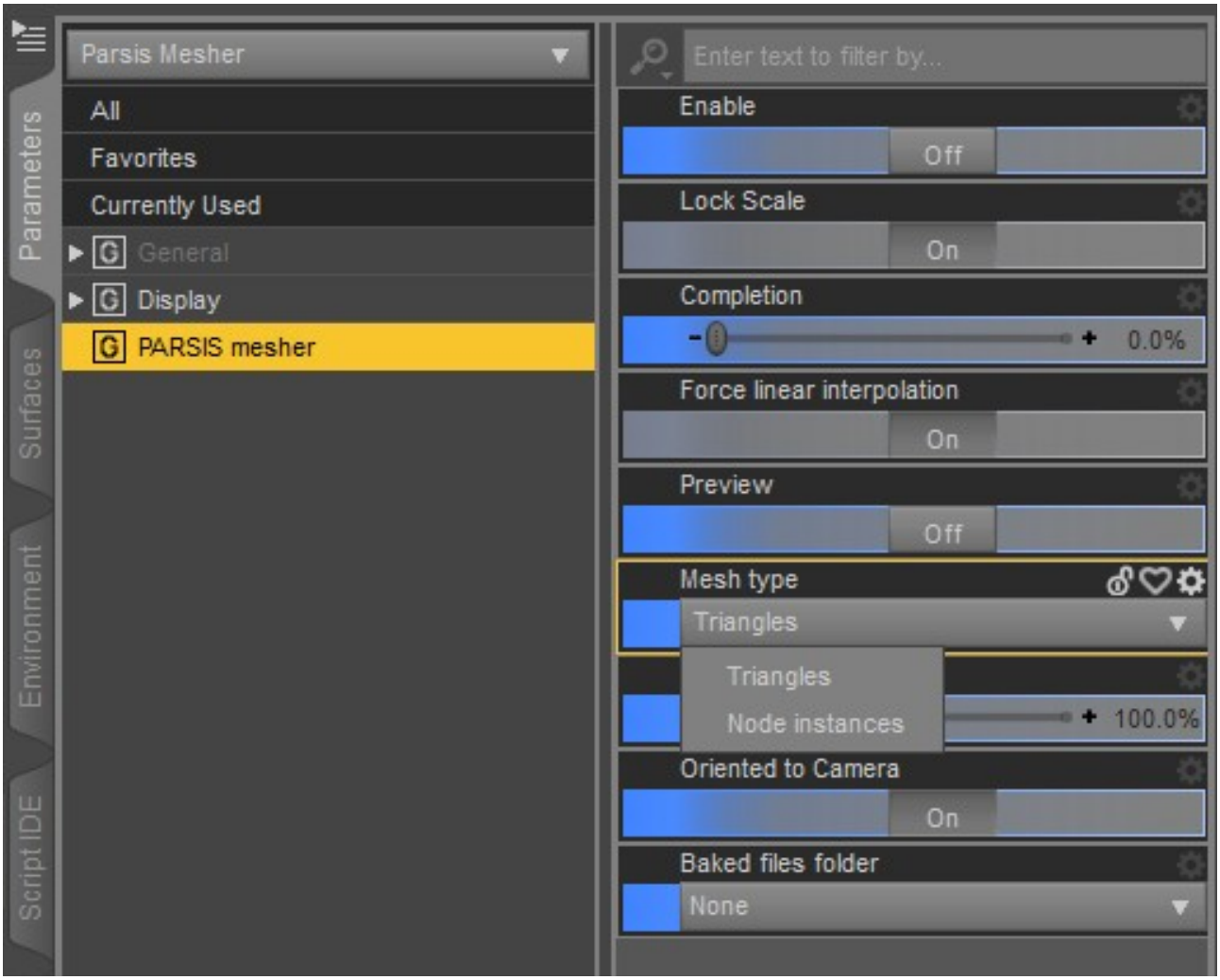
Force linear interpolation: The default interpolation of the keyframes in Daz Studio is nonlinear, but to get a correct synchronization of the simulation in the Mesher with the scene, a linear interpolation is needed. Setting On this button forces a linear interpolation of the keyframes of the Mesher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

Preview: If this is *On*, the mesher will show a fraction of the particles as points in the viewport.

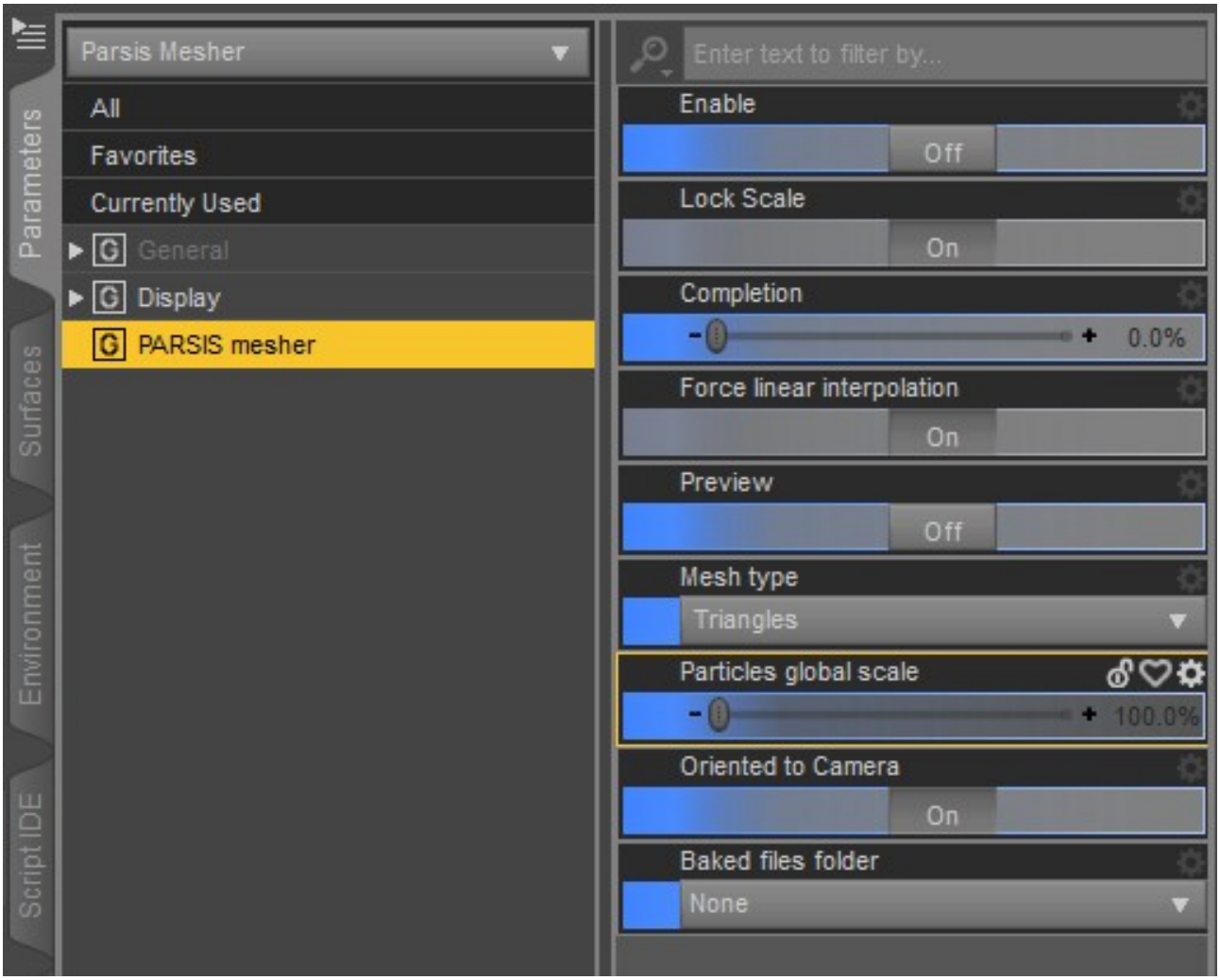
Particles global scale: The particles could be scaled using this property. Their size will change, while the position will be the same. Its animatable *Preview* mode will ignore this setting.

Mesh Type: this property only appears if Preview is set to Off. It has two options: *Triangles* (in fact, billboards) and *Node instances*.

Data folder: This is used to get the location of the data folder, where the baked files are stored.

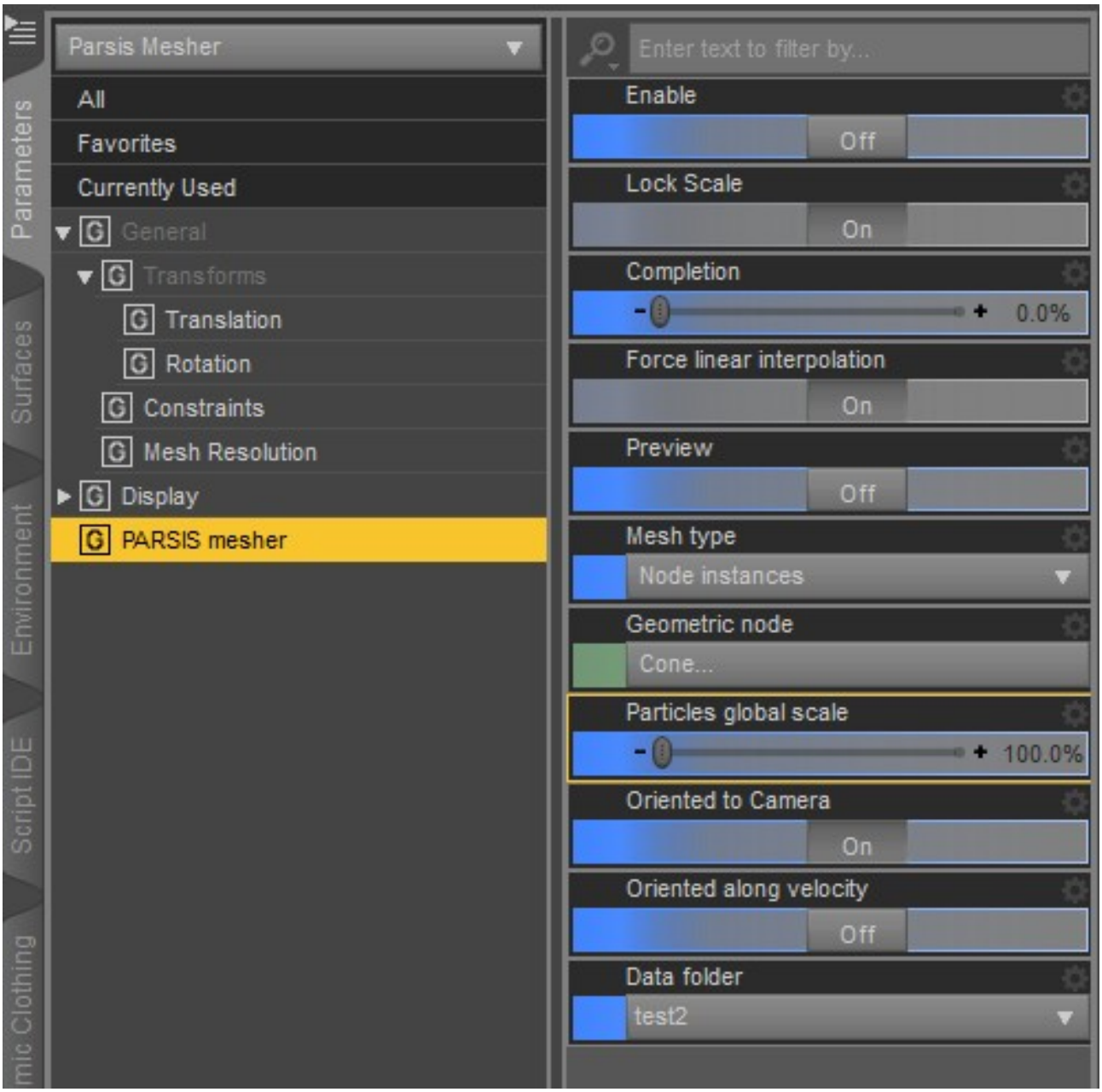


If *Triangles* is chosen, the properties will be as the following image:



Oriented to Camera: this orients the particles to face the opposite direction in which the camera is oriented. It's only valid with *Triangles* as *Mesh type*.

If *Node instances* is chosen, the properties will be as the following image:



Geometric node: this is the geometric node that will be used as a model for the particles.

Oriented along velocity: if this option is set *On*, the particles will be oriented along the velocity vectors. This is useful to simulate projectiles.

How to use the materials with the mesher

Triangles type of mesh:

When the mesher is pointing to a data folder, the plugin will load the first file to know how many materials are supposed to be used in the simulation. Then, the mesher creates as many materials (or shaders) as needed. It numbers each one of them as *ParsisMatXXX*, that is *ParsisMat000, ParsisMat001*, and so on. They can be seen as the shaders in the Daz Studio's *Surfaces – Editor.* The user should edit the materials or apply a shader to them.

Each particle will be changing its material along the time in the scene. It starts with the number of the material stated in the source as initial material (if *Enable to target a material* of at least one source is *On*). Then, at some time after, it changes to the following material in numeric order. And so on, until it reaches the last material.

If the *Enable to target a material* property of all the sources in the scene is *Off*, then the only available material will be the *ParsisMat000.*

There are 6 shaders for the *Triangles type of mesh* included in this package. Three are for the Iray render and three for the 3Delight one. In addition, there are 10 shader utilities for controlling the opacity of the particles, 5 for each one of the two renders.

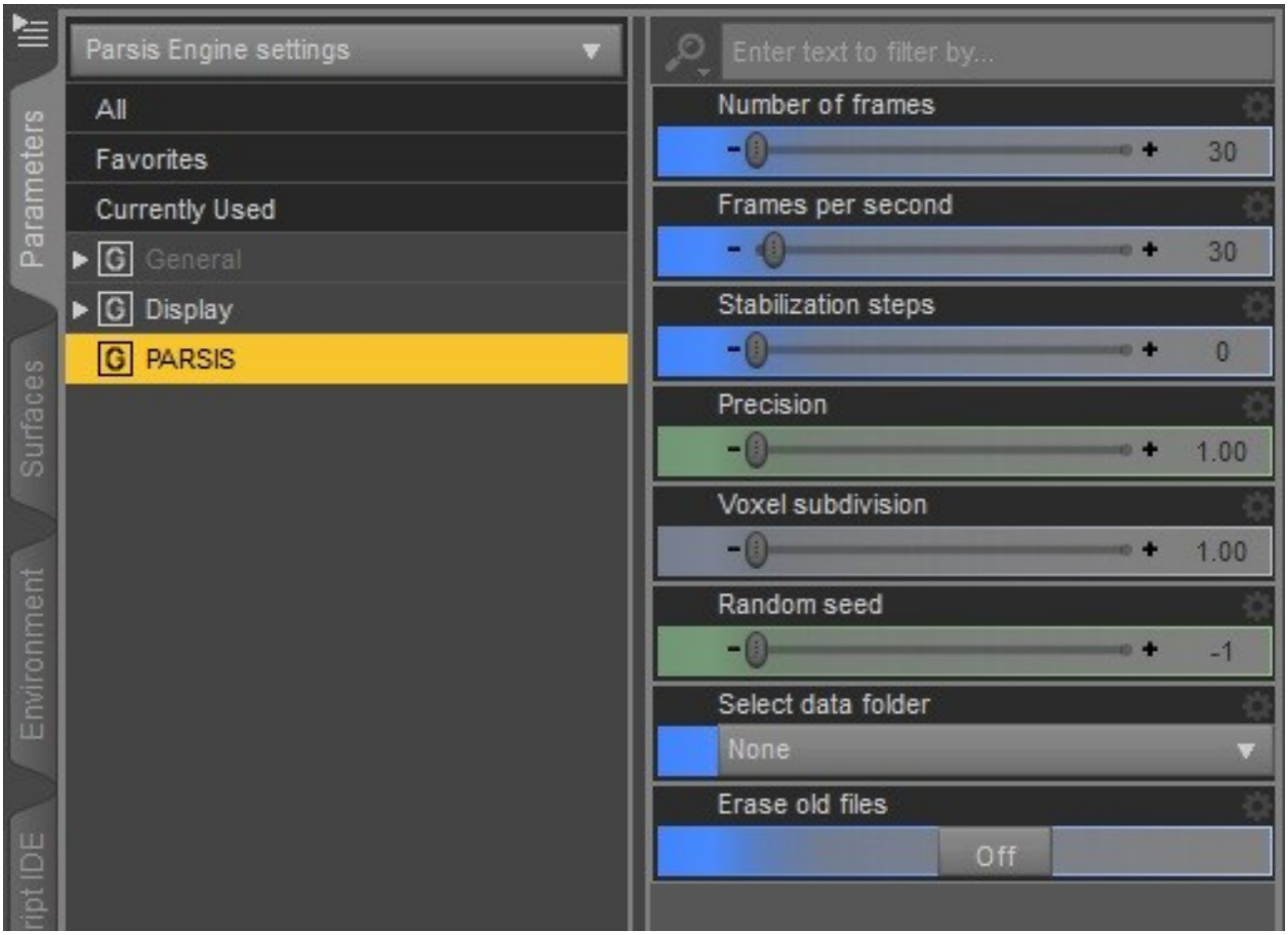
Instance nodes type of mesh:

The process is similar to the *Triangles type* one. But instead of shaders in the mesher itself, copies of the *geometric node* are made and parented to the mesher. Each one has the name of the *geometric node* followed by the suffix *_ParsisNodeXXX*. The user should modify the shaders of these node copies.

The copies shouldn't be deleted nor the original *geometric node*, otherwise, the mesher won't show the particles. If any copy is erased, to restart the mesher, the user must change the *mesher type* to *triangles* and then change again to *Instance nodes.*

ENGINE SETTINGS:

This object can contain the main settings of the engine and it is saved with the scene. There could be as many *Engine settings* objects in a scene as the user wants, but only one will be used during a run of the engine.



Number of frames: This is the number of frames that PARSIS will simulate.

Frames per second: This is the rate of frames simulation.

Stabilization steps: PARSIS will run the number of steps indicated by this property without advancing the timeline.

Precision: It is the precision of the simulation; the higher the precision, the better (and slower) the results.

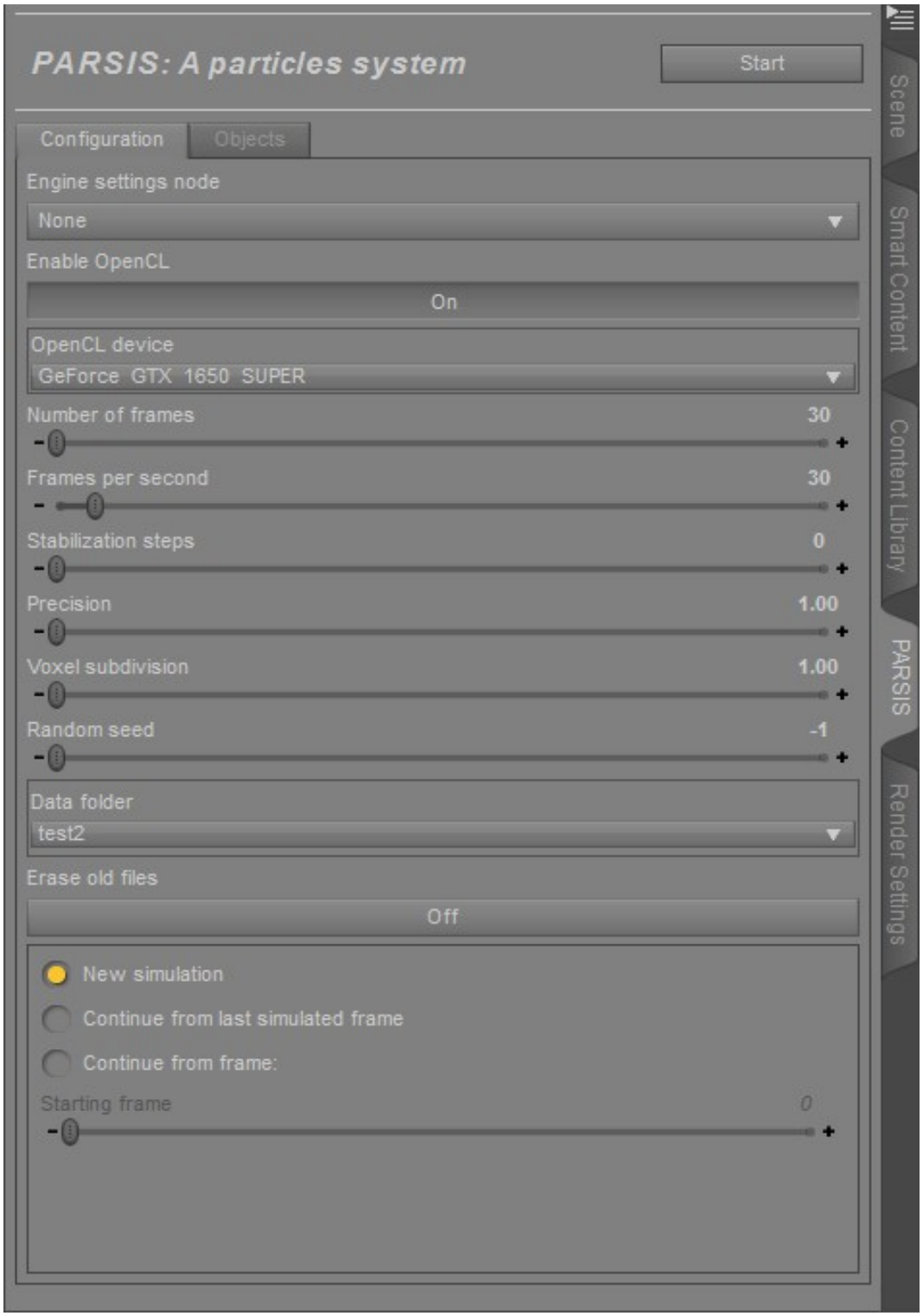
Voxel subdivision: This is the subdivision that the engine internally applies to the geometric objects. The higher, the more accurate is the geometry for calculations, but also the slower the simulation. However, for thin objects, a too high value could reduce the accuracy.

Random seed: This is the random number of the seed. The engine uses a pseudorandom numbers generator in the creation and elimination of particles, flux forces, and other actions. The generator creates a unique sequence of random numbers for each *seed*. The user can set the seed. But if this number is set to -1 (the default), the engine will get a seed from a timer, so the random numbers sequence will be different each run.

Select data folder: This is used to get the location of the data folder, where the baked files and logs are stored

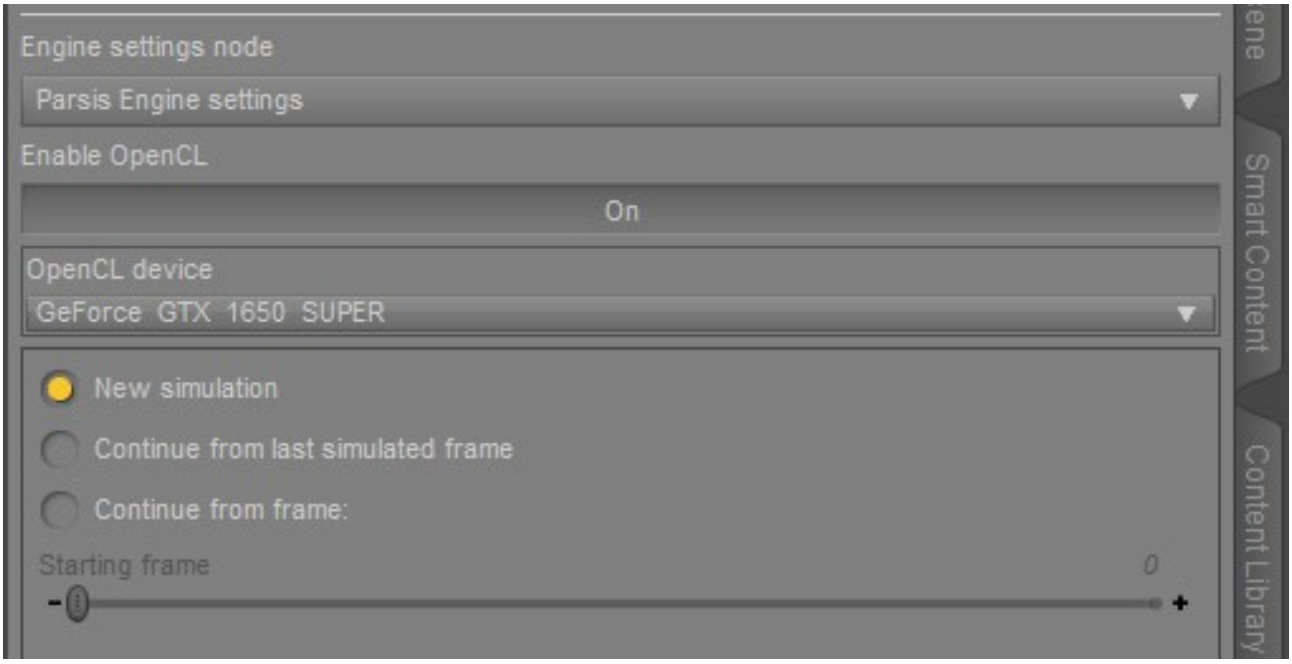
Erase old files: By *en*abling this property, the old files in the current folder will be erased the next time the simulator is running. However, if the user checks the *Continue from last simulated frame* or *Continue from frame* options in the PARSIS pane the *Erase old files* property will be ignored.

Instead of using the *Engine settings* object for the settings, it could be used the *PARSIS pane*, but for except the OpenCL and Data folder settings, it does not save anything between Daz Studio sessions.



In addition to the settings the *Engine settings* contains, the *PARSIS pane* has the following parameters:

Engine settings node: This could be any *Engine settings* object in the scene. If any of them is selected, the pane will shrink as in the following image:



Enable OpenCL: this allows to switch off or on the OpenCL. The simulation without OpenCL is somewhat slower.

OpenCL device: This is the OpenCL device that will be used in the simulation.

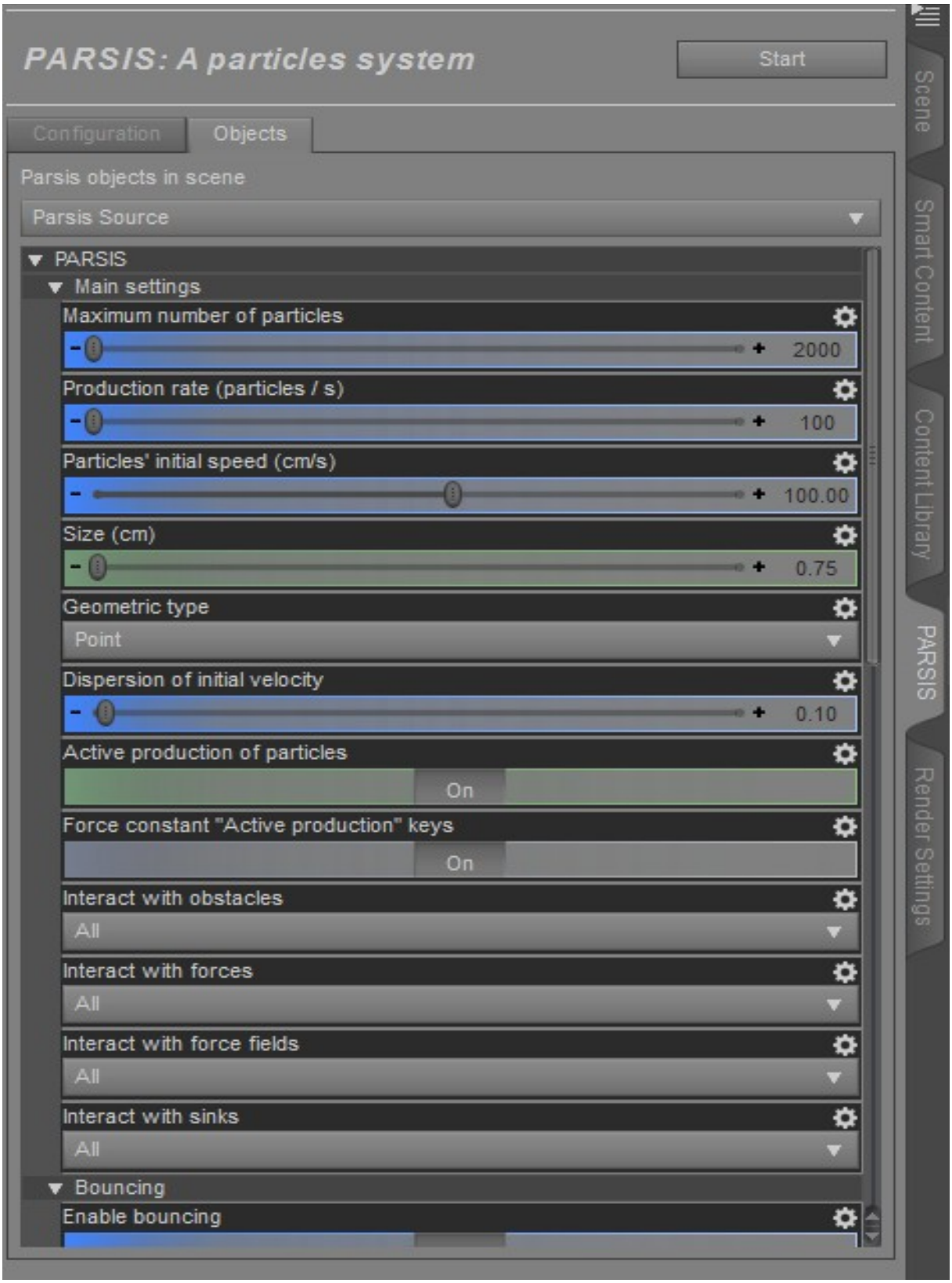
NOTE: The PARSIS' macOS version does not support OpenCL, so it do not have the *Enable OpenCL* and the *OpenCL device* options

New simulation: If this radio button is checked, the simulation will start from the zero frame.

Continue from last simulated frame: if this button is checked, the simulation will start since the last frame simulated.

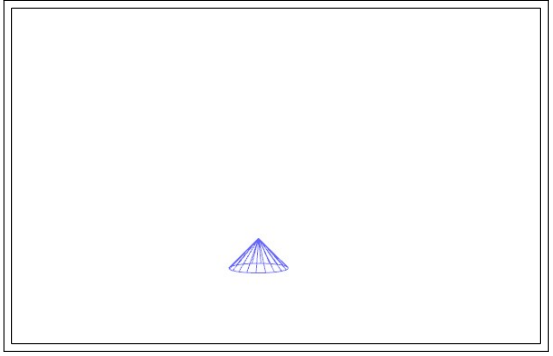
Continue from frame: if this is checked, the simulation will start from any simulated frame. The plugin will show the number of frames available in the corresponding *data folder* (showed in the *pane* or in the *Engine settings* object:

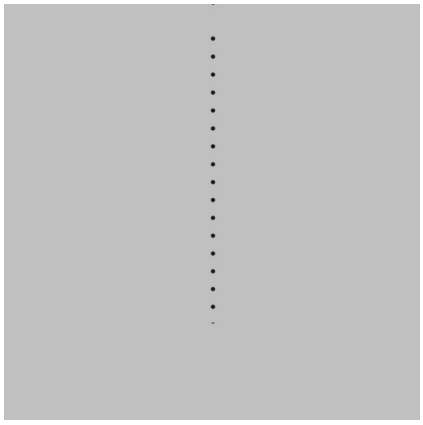
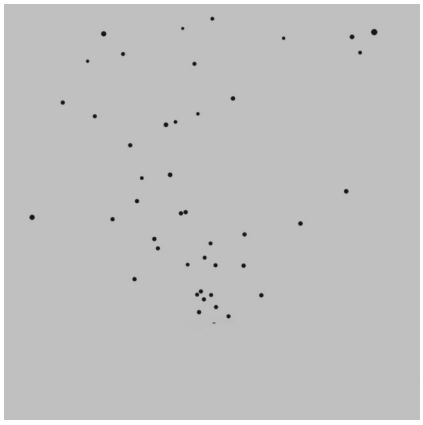
Instead of using the Parameters tab to set up the configuration of the PARSIS objects, the user can use the *Objects* tab of the pane, by choosing the object in the *Parsis objects in scene* combo box.



EFFECT OF THE SETTINGS

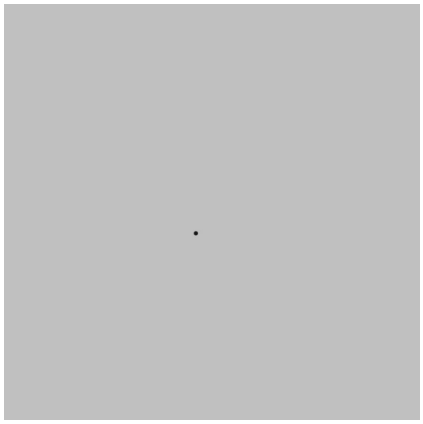
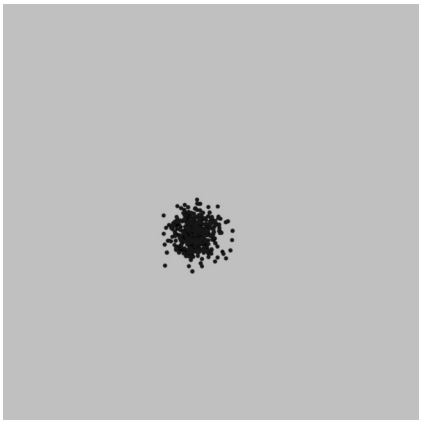
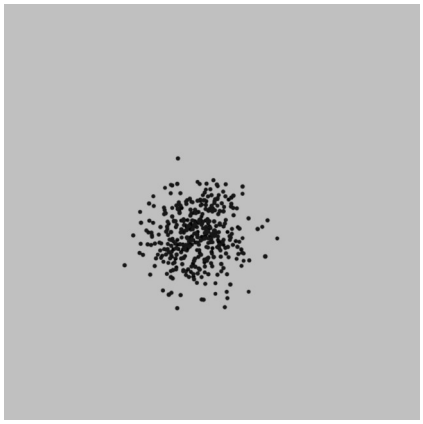
Dispersion of initial velocity:

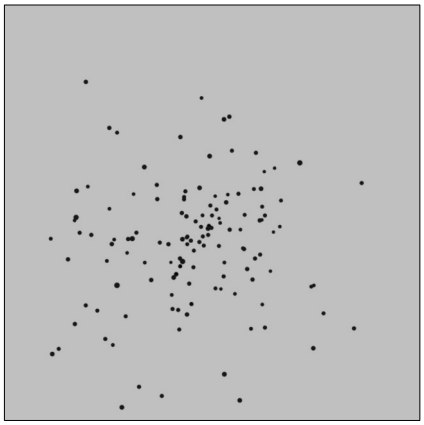
Configuration of the scene


Dispersion = 0.0	Dispersion = 0.1 (default)	Dispersion = 0.5
		

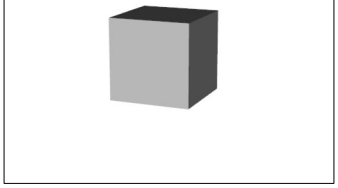
Blob geometric type:

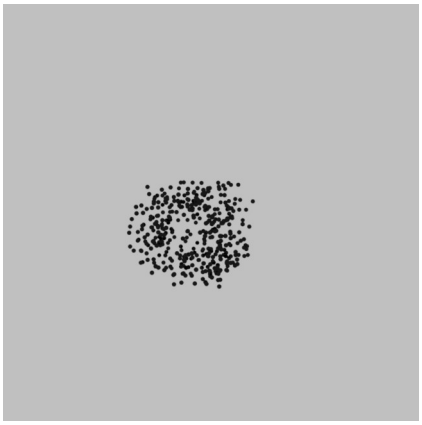
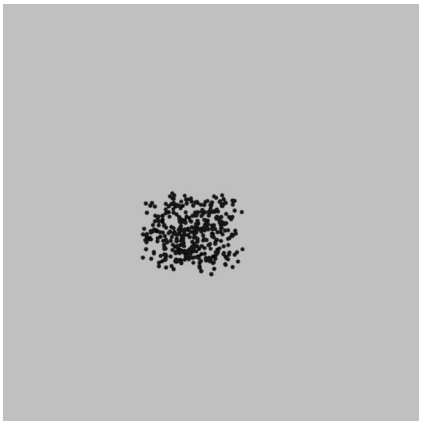
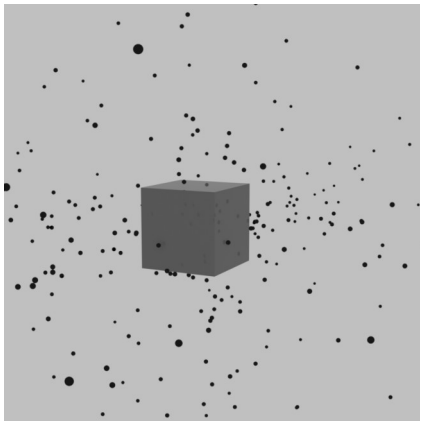
Configuration of the scene


Dispersion = 0.0	Dispersion = 0.5	Dispersion = 1.0
		

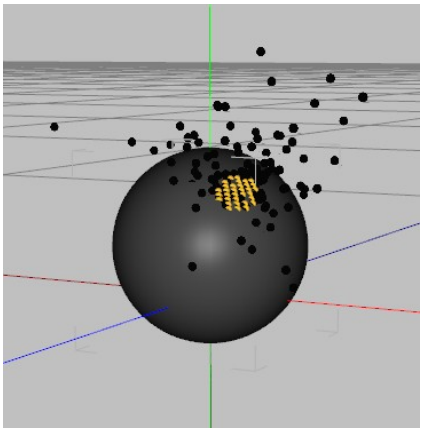
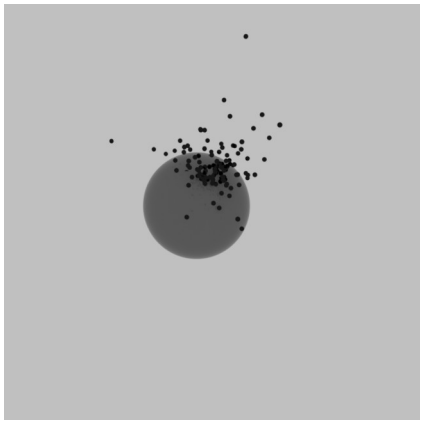


Any node geometry type:

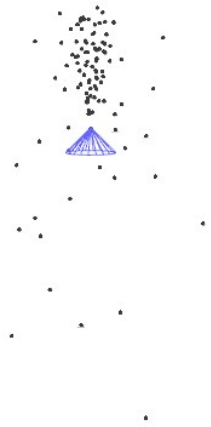
Configuration of the scene


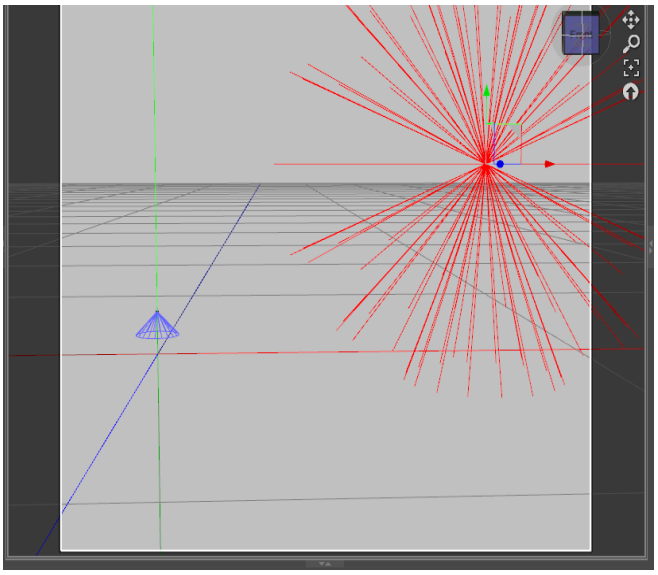
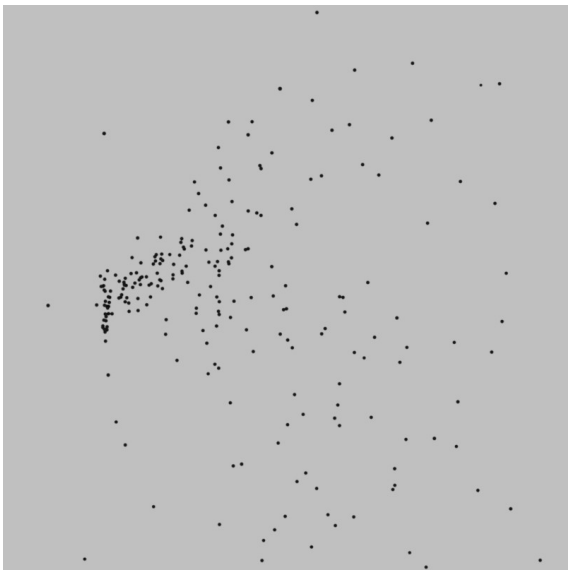
Particles Origin: Surface Speed =0.0 (node invisible)	*Particles Origin*: Volume Speed =0.0 (node invisible)	*Particles Origin*: Surface Speed =100.0 (node visible)
		

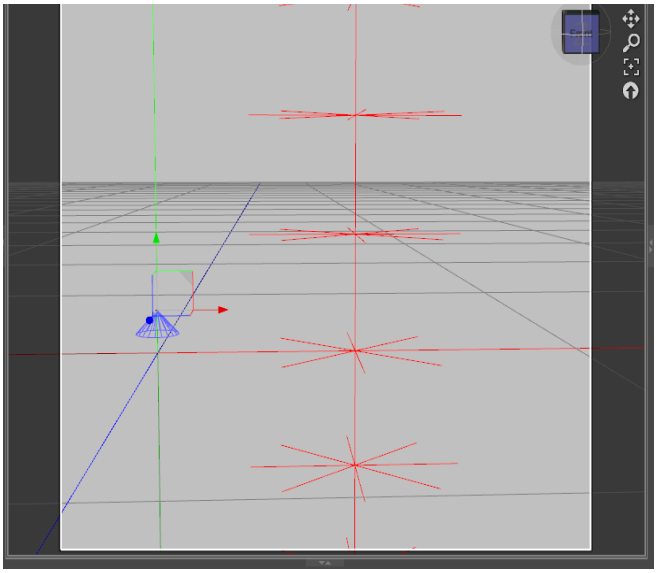
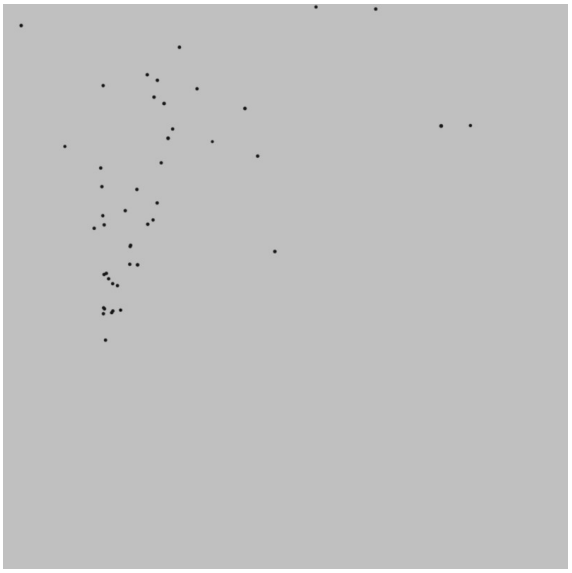
Selected Vertices case:

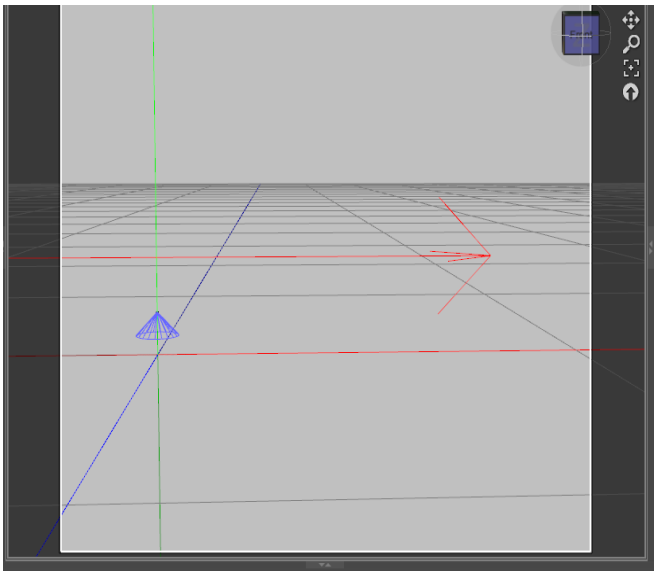
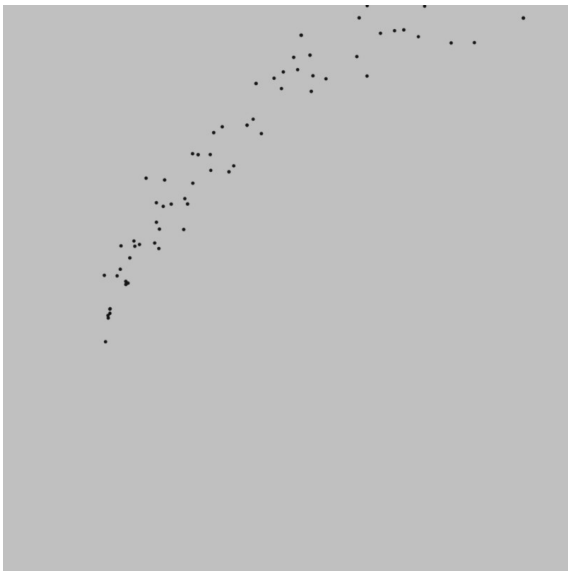
Particles Origin = Selected Vertices	
	

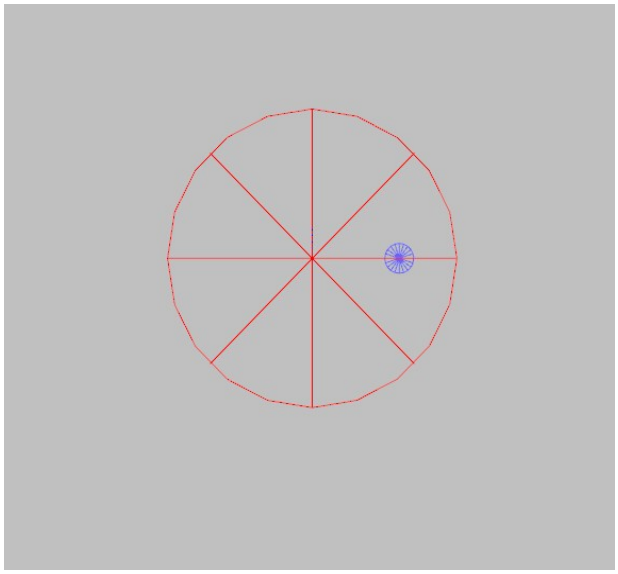
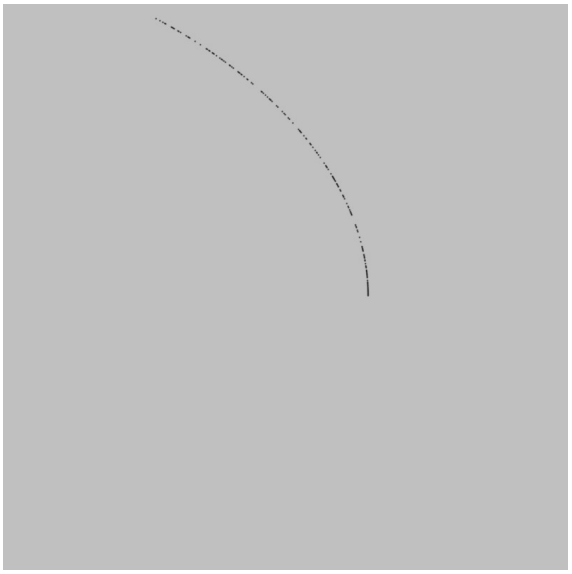
Forces:

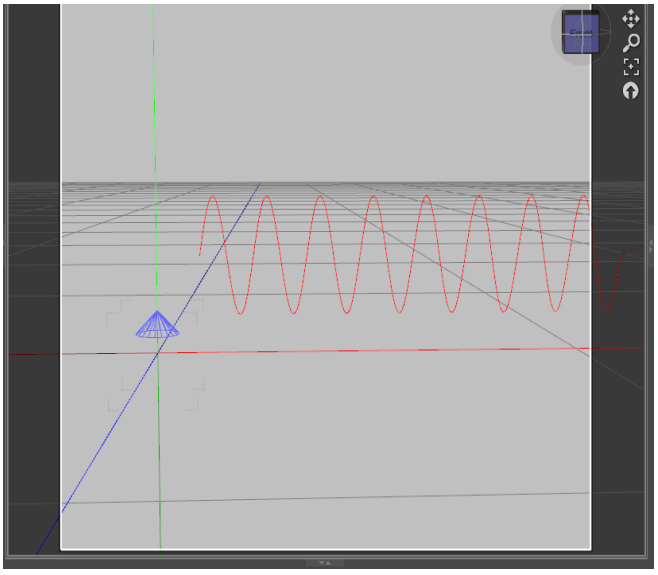
Only gravity


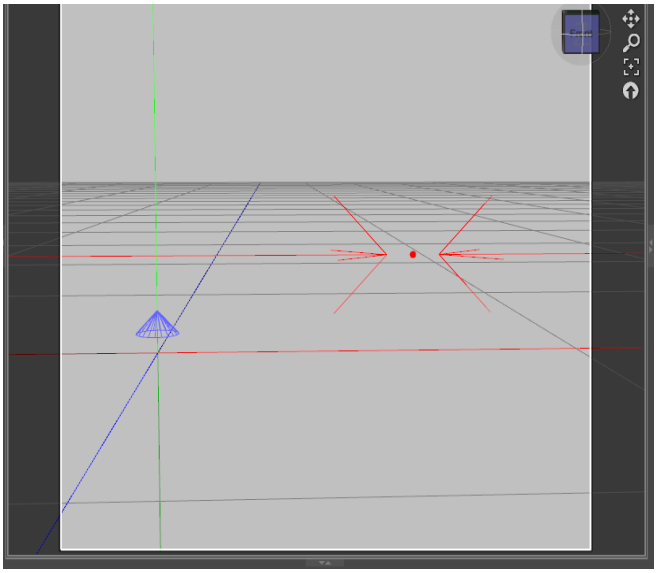
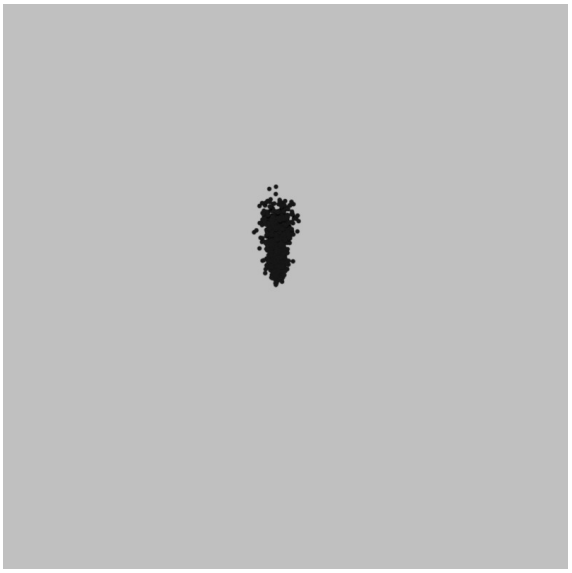
Point force:	
Intensity = 1000 m/s^2	
	

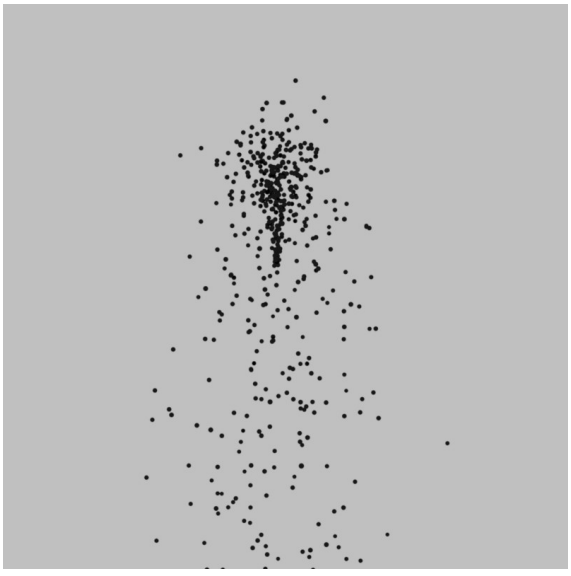
Line force:
Intensity = 10000 m/s^2
 

Directional force:
Intensity = 10 m/s^2
 

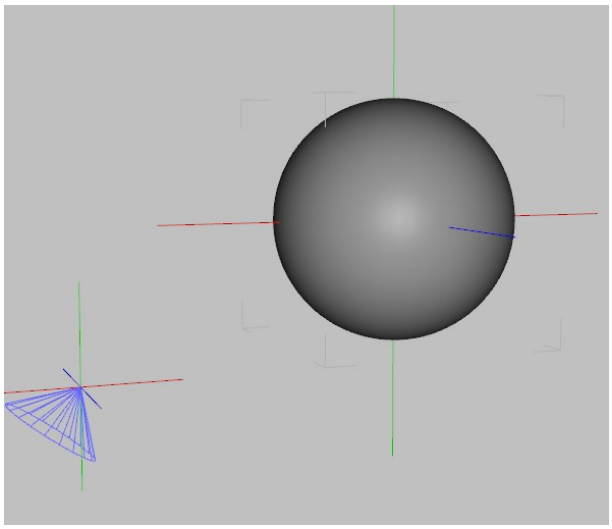
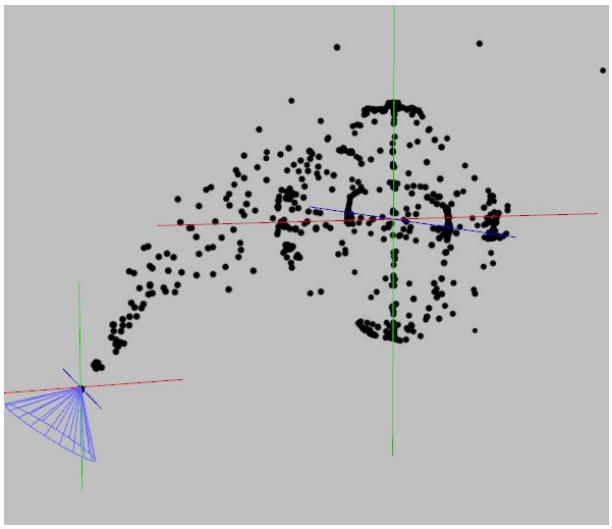
Torque force:
Intensity = 1.0 m/s^2 , Falloff rate = 1, Precision = 10.0 Source speed = 0.0 cm/s
 

Flow force:
Intensity = 10000 m/s^2, Dispersion = 2.0, Precision = 3.0
 

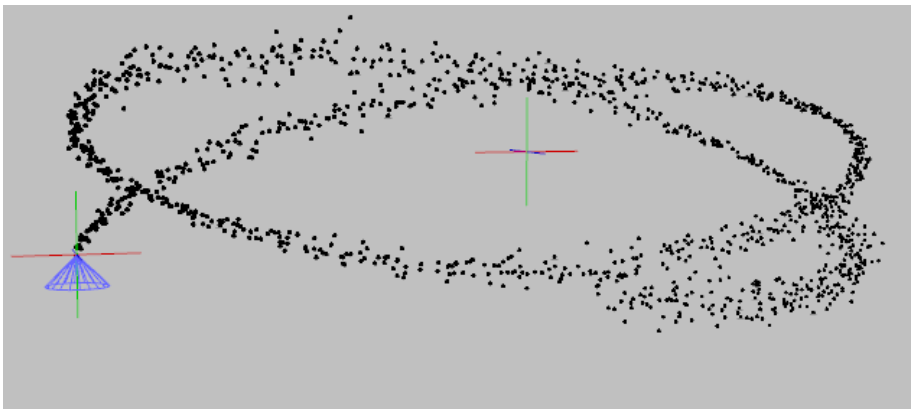
Internal force:	
Intensity = 10 m/s^2, Falloff rate = 1	
	

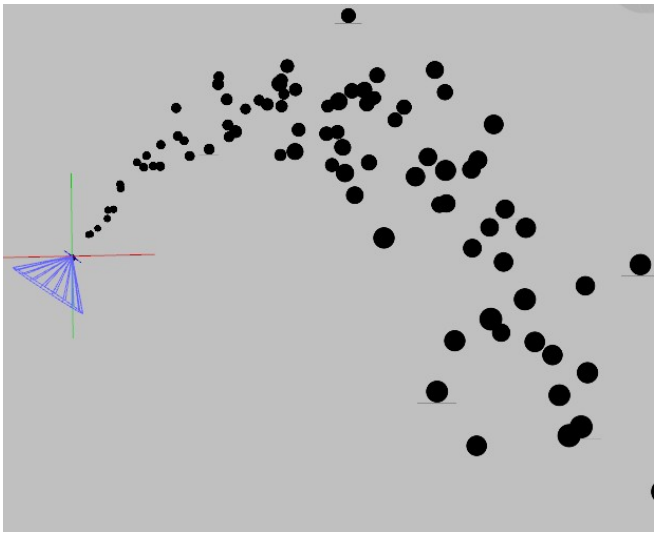
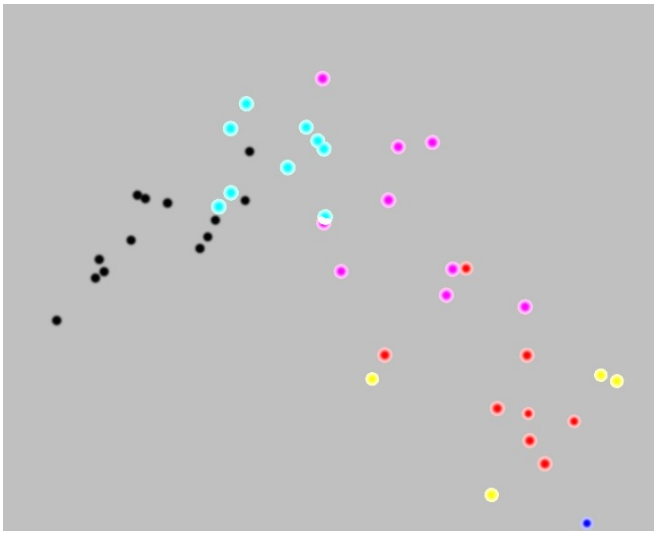
Damping force:	
Intensity = 0.0 m/s^2	Intensity = 10.0 m/s^2
	

Force field:

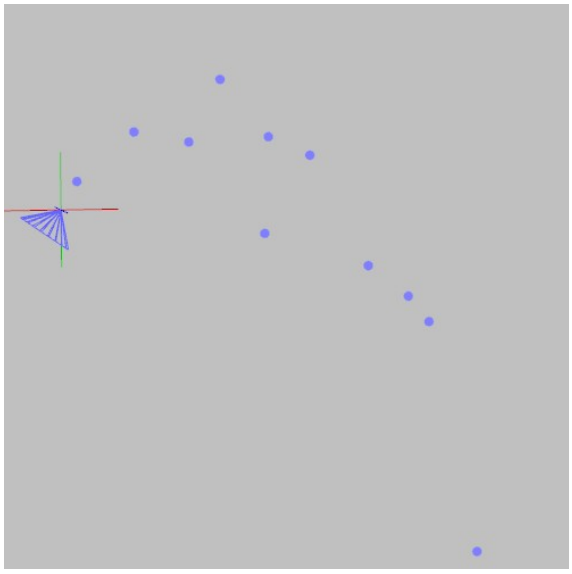
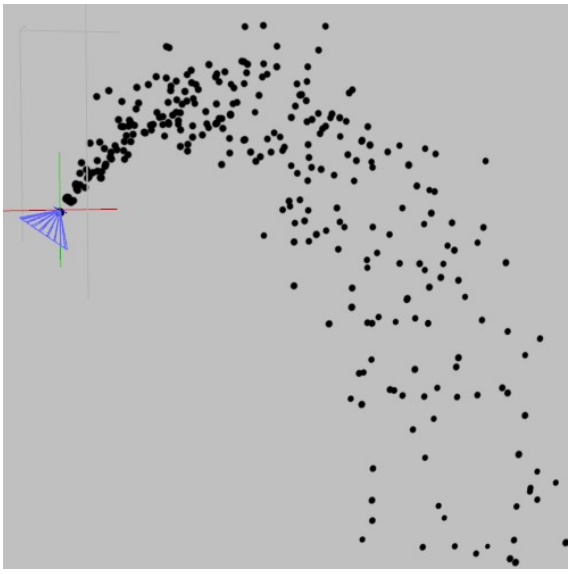
Intensity = 1000 m/s^2	
	

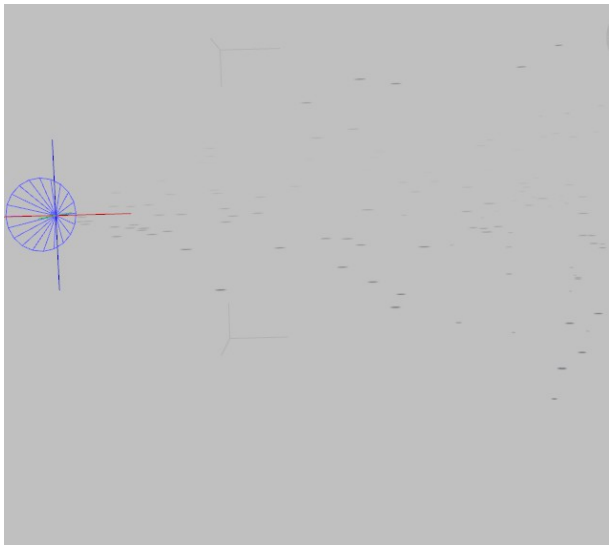
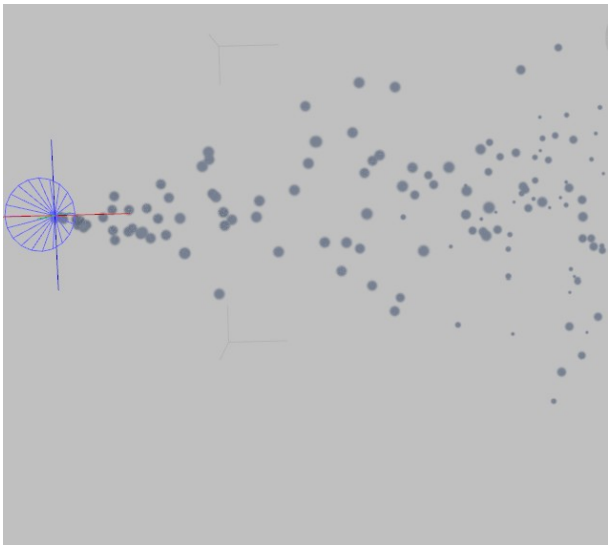
Targeting:

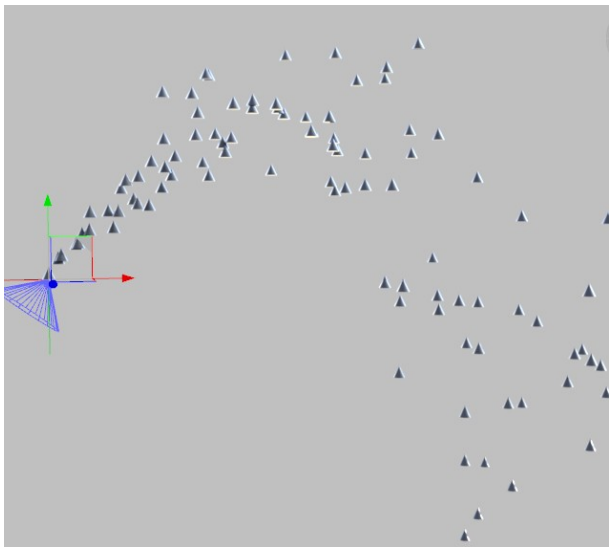
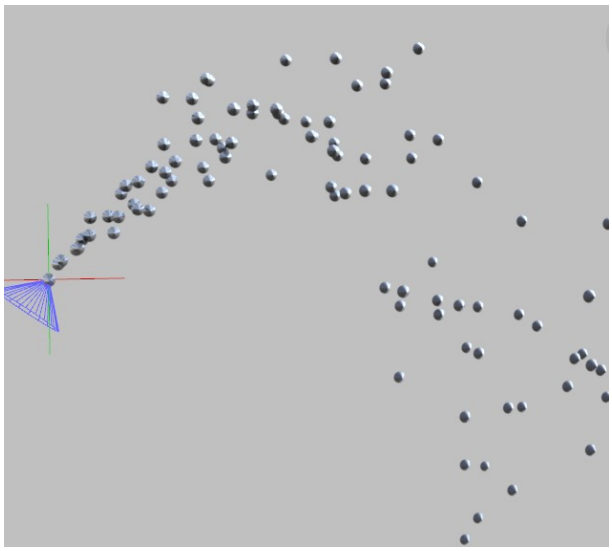
Target a position: The target is a Null node, the gravity force is the default.	
Intensity= 0.1 (default)	
Intensity= 2.0 Precision = 5.0	

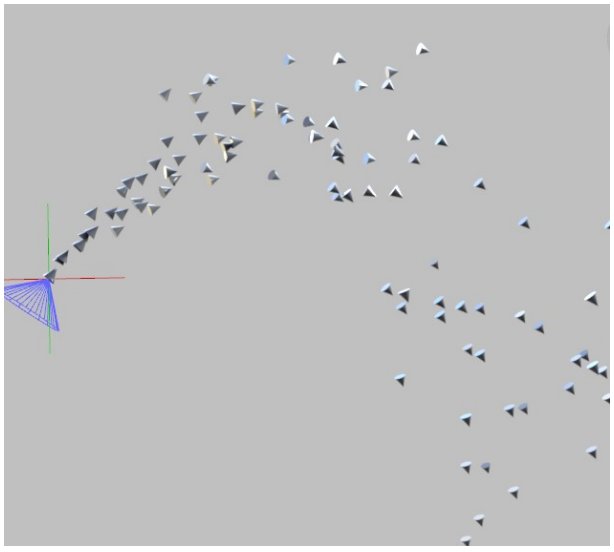
Target a size:	***Target material:***
	

***Mesher*:**

Preview	Mesh type: Triangles (Shader applied)
	

No oriented to camera	Oriented to camera
	

Mesh type: Geometric Node (using a cone primitive)	
No oriented	Oriented to camera
	

Oriented along velocity


References:

McAllister, D. K. (2000). The design of an API for particle systems. UNC Computer Science Tech Report.

Licences:

ArrayFire

PARSIS uses a modified version of ArrayFire 3.8.0

OpenVDB 6.1.0

https://github.com/AcademySoftwareFoundation/openvdb

IlmBase

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.

OpenBlas

Copyright 2009, 2010 The University of Texas at Austin.
All rights reserved.

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

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

2. 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.

THIS SOFTWARE IS PROVIDED BY THE UNIVERSITY OF TEXAS AT AUSTIN ``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 UNIVERSITY OF TEXAS AT AUSTIN 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.

The views and conclusions contained in the software and documentation are those of the authors and should not be interpreted as representingofficial policies, either expressed or implied, of The University of Texas at Austin.

Version 1.0.2

Bugs fixed:

- If the Fisio SB AddOn was installed, Daz Studio crashed when an enabled *Parsis Mesher* was using *Node instances* as a *Mesh Type*.

Version 1.0.1

Bugs fixed:

- Daz Studio crashed when enabled in a *Source* targeting a material that has run some times but now has different *Targeting material* values, before changing the *Mesh type* in the enabled *Mesher* to Node instances and run again.

- When using more than a *Source* in the scene and using a random seed other than -1, all the sources displayed the same particles' trajectories.

- Sometimes, when saving an enabled *Mesher* with Node instances as the *Mesher type*, some of the instances stayed in the scene, even if the *Mesher* was disabled or hided.

- When choosing an open-mesh object (as a plane) as an *Any node Source*, and running the simulation, most of the times, Daz Studio crashed. Note: open-mesh objects are not allowed as *Sources*.