



---

# PreonLab 4.2.4

## Manual

Get support via:

 <http://help.preonlab.eu/>

 [support@fifty2.eu](mailto:support@fifty2.eu)

 +49 (0) 761 45 89 23 – 81

Mo-Fr: 9am-5pm (CET)

August, 2020

© FIFTY2 Technology GmbH

# Contents

<b>1</b>	<b>About</b>	<b>1</b>
<b>2</b>	<b>Installation</b>	<b>2</b>
2.1	Installation / Update	2
2.1.1	Windows	2
2.1.2	Linux	2
2.1.3	preonpy	3
2.1.4	Systems without graphics	3
2.2	Licensing	3
2.2.1	Managing license files	5
2.2.2	Installation of an RLM license server	6
2.2.3	Restricting license access	6
2.2.4	Troubleshooting	7
2.3	Environment variables regarding multi-threading	7
2.3.1	Setting the number of threads	8
2.3.2	Performance on multi-socket systems	8
2.3.3	Troubleshooting	8
<b>3</b>	<b>PreonLab GUI general usage</b>	<b>9</b>
3.1	User interface overview	9
3.2	Graphics window	10
3.2.1	Navigating in the graphics window	10
3.3	Toolbar	10
3.3.1	Translate	11
3.3.2	Scale	11
3.3.3	Rotate	11
3.3.4	Particle picking	11
3.3.5	Placement tool	11
3.3.6	Measure tool	12
3.4	Taskbar	12
3.5	Timeline	12
3.6	Scene Inspector	14
3.7	Property Editor	16
3.8	Message window	17
3.9	Console	17
3.10	OSD (On-Screen-Display)	19
3.11	Keyboard shortcuts	19
3.12	User Preferences	20
3.13	Presets	21
3.14	Experimental features	22
3.15	Start parameters	22

3.16	Known issues and workarounds	22
3.16.1	OpenGL	22
3.16.2	Display resolution	22
3.16.3	Drag & drop	23
3.16.4	Starting PreonLab via Windows Remote Desktop	23
<b>4</b>	<b>Common properties</b>	<b>25</b>
4.1	General	25
4.2	Appearance	25
4.3	Transformation	26
4.4	Statistics	27
<b>5</b>	<b>Connections</b>	<b>29</b>
5.1	Using the connection editor	30
5.1.1	Only showing selected objects	30
5.1.2	Rearranging objects	30
5.1.3	Filtering connection types	30
5.1.4	Grouping objects	31
5.2	Transform connections	31
5.2.1	Relative transformations	31
<b>6</b>	<b>Keyframing</b>	<b>32</b>
6.1	Keyframe editor	32
6.1.1	Copy and paste keyframes	34
6.1.2	CSV import / export	34
6.2	Best practices	34
6.2.1	Make the camera follow an object	35
6.2.2	Shortcuts	35
6.3	Known limitations	35
<b>7</b>	<b>Statistics and plots</b>	<b>36</b>
7.1	Plots	36
7.1.1	Plot color	37
7.1.2	Plot range	37
7.1.3	Sampling	37
7.1.4	Export	37
<b>8</b>	<b>Scene and basic objects</b>	<b>38</b>
8.1	Scene	38
8.2	Scene UI Settings	40
8.3	Transform groups	41
8.4	Point	42
<b>9</b>	<b>Fluids</b>	<b>43</b>
9.1	Preon solver	43
9.1.1	General settings	43
9.1.2	Pressure-solver settings	44
9.1.3	Viscosity	47
9.1.4	Surface tension	51
9.1.5	Solid-fluid interaction	52
9.1.6	Timestep computation	53

9.1.7	Deletion criteria	54
9.1.8	Thermodynamics	55
9.1.9	Evaporation	57
9.1.10	Local refinement	59
9.1.11	Rendering of particles	61
9.1.12	Serialization	61
9.1.13	CSV export	62
9.2	Snow solver	62
9.2.1	Best practices	63
9.2.2	Properties	63
9.2.3	Snow Parametrization	65
9.3	Preon mesher	66
9.3.1	First steps	66
9.3.2	Parameters explained	67
9.3.3	Common issues	68
<b>10</b>	<b>Sources</b>	<b>70</b>
10.1	Area source	70
10.1.1	Specifying the source area	71
10.2	Volume source	73
10.2.1	Preview of generated particles	75
10.2.2	Filling containers with a volume source	75
10.2.3	Alignment	77
10.2.4	Specifying initial velocities	77
10.3	Rain source	78
<b>11</b>	<b>Boundary Domains and Conditions</b>	<b>79</b>
11.1	Box and cylindrical domain	79
11.1.1	Using meshes to define the domain volume	80
11.1.2	Using multiple domains	80
11.1.3	Using boundary domains to improve performance	81
11.2	Maximum velocity condition	81
11.3	Air Objects	82
11.3.1	Evaporation with air objects	83
11.3.2	Thermodynamics with air objects	83
11.4	Experimental: Open boundary plane	84
11.5	Outflow domain (Prototype)	85
11.5.1	Connection to sources	86
<b>12</b>	<b>Force Fields</b>	<b>87</b>
12.1	Gravity	87
12.2	Drag Force	87
12.2.1	Constant	88
12.2.2	Terminal Velocity	88
12.2.3	Automatic Terminal Velocity	88
12.2.4	Liu Model	89
12.3	Air Flow	89
12.3.1	Static air flow data import via the csv format	89
12.3.2	Static air flow data import via the EnSight Gold format	90
12.3.3	Transient air flow data import	91
12.3.4	Viewing the air flow field	92



12.3.5 Air flow parameters	92
12.3.6 Air Flow box	94
12.3.7 Best practices	94
12.4 Air Pressure	95
12.5 Car suspension model	95
12.5.1 Half-car suspension model	97
12.5.2 Best practices	100
<b>13 Solid Objects</b>	102
13.1 Thermodynamics	104
13.1.1 Known limitations	104
13.2 Film wetting	104
13.2.1 Parameters explained	105
13.2.2 Visualizing the wetting film	105
13.2.3 Evaporation of film	106
13.3 Visualization of solid objects	106
13.3.1 Random coloring for solids	107
13.4 Primitive shapes	107
13.5 Mesh	108
13.5.1 Mesh resource	108
13.6 Alembic Mesh	109
13.7 Porous Rigid	110
13.7.1 Best practice	110
13.8 Changing the pivot / center-of-mass	111
13.8.1 Rotating around a custom axis	111
13.8.2 Solid velocities for meshes with pivot	111
<b>14 Rigid body simulation</b>	113
14.1 Center-of-mass	115
14.2 Bullet	115
14.2.1 Collision margin	116
14.3 Particle-based rigid body solver	117
14.3.1 Collision margin	117
<b>15 Rendering</b>	118
15.1 Cameras	118
15.2 Clipping object	119
15.3 Lights	120
15.3.1 Directional light	120
15.3.2 Point light	121
15.4 Preon renderer	121
15.4.1 The rendering dialog	122
15.4.2 Rendering during simulation	122
15.4.3 Parameters	123
15.5 Materials	125
15.5.1 Shared material paramaters	125
15.5.2 Surface material	126
15.5.3 Textured surface material	127
15.5.4 Volumetric material	128
15.6 Presets	130
15.6.1 Examples	130

<b>16 Sensors</b>	134
16.1 Mesh-based sensors	134
16.2 Sensor color legend	134
16.3 Distance sensor	135
16.4 Volume sensor	135
16.4.1 Using meshes as volume sensors	136
16.5 Wetting sensor	136
16.5.1 Known issues	138
16.6 Force sensor	138
16.7 Particle tracker	139
16.8 Heat flux sensor	139
16.9 Pathlines	141
16.9.1 Parameters explained	142
16.9.2 CSV export	142
16.10 Sensor plane	143
16.10.1 Context actions	145
16.11 Sensor mesh	145
16.12 Projection fields	146
16.12.1 Velocity projection field	146
16.12.2 Height field	146
16.13 Height Sensor	146
16.14 Air flow visualizer	148
16.15 Python particle access	150
<b>17 Import &amp; Export</b>	151
17.1 Scene loading and saving	151
17.1.1 Archiving and reducing disk space consumption	153
17.1.2 Known issues	153
17.2 Import meshes	153
17.3 Import animation data	154
17.3.1 Data format	155
17.4 Import VDAFS data	155
17.5 Import Alembic file	156
17.5.1 Alembic files with HDF5 data format	156
17.6 Import Airflow	157
17.7 Import statistic data	157
17.8 Import Point Cloud Resource	157
17.8.1 Importing temperature samples from CSV	157
17.8.2 Importing velocity samples from CSV	159
17.9 Export Alembic file	160
17.10 Export particle data	160
17.11 Export to EnSight	161
17.12 Export video	161
17.12.1 Best practices	162
<b>18 PreonCLI</b>	164
18.1 Simulating a scene	164
18.2 Environment variables	164
18.3 Running a Python script	164
18.4 Optional start parameters	165

<b>19 Python API</b>	166
19.1 Supported Python version	166
19.2 Installation as Python package	166
19.2.1 Installing Python from package manager	166
19.2.2 Installing Python from source	167
19.2.3 Installing Python on Windows	168
19.2.4 Installing PreonPy	168
19.2.5 Installing multiple versions	169
19.3 Usage	169
19.3.1 Error Handling	170
<b>20 Distributed computing using MPI</b>	171
20.1 Installation	171
20.1.1 MPI	171
20.1.2 PreonNode	172
20.2 Usage	172
20.3 Optional start parameters	173
20.4 Environment variables	174
20.5 Performance guide	175
20.5.1 Do not create one process per core	175
20.5.2 Particle workload requirements	175
20.5.3 Reduce post-processing effort	175
20.5.4 Network requirements	176
20.5.5 Multi-socket systems	176
20.5.6 Heterogeneous clusters	176
20.5.7 Maximum number of nodes	176
20.5.8 Scaling	176
20.6 Best practices for typical environments	177
20.6.1 Running PreonNode with Open MPI	177
20.6.2 Running PreonNode on IBM Platform LSF	178
20.6.3 Running PreonNode with MPICH	178
20.7 Known limitations	178

# 1 About

PreonLab is a physically-based simulation framework which is developed to simulate free-surface flows as fast as possible.

We believe that this is influenced by three main factors: efficiency, usability, and reliability.

**Efficiency:** PreonLab is powered by our point-based fluid simulation kernel PREON®. The key idea behind PREON® is to avoid artificial and specialized models to capture specific properties of real-world fluids. Instead, PREON® solves the fundamental physical equations that govern the flow of fluids in a very fast and resource efficient way, allowing simulation in unprecedented resolutions. This opens a completely new range of simulation possibilities – revealing new insights in the early stages of engineering development and design.

**Reliability:** Reliability is crucial for professional software. For us, this does not only mean that we have to meet our own high standards in terms of security and software quality, but most of all delivering solid simulations that meet the expectations of the engineer. Reproducible simulation results and validation are in the focus of our daily work and throughout our internal quality requirements.

**Usability:** We believe that simulation tools can be user friendly and should not be overloaded with hundreds of options and parameters to set. Our goal is to condense the options as much as possible, so that your workflow is optimized, while the chances of misconfiguration is minimized. PreonLab drastically reduces your preprocessing effort and provides strong postprocessing capabilities.

## 2 Installation

This chapter explains how to install and update PreonLab. It also explains how to link PreonLab to your license file and what needs to be additionally installed for network licenses. At the end of the chapter, system-dependent issues are listed together with workarounds for these known problems.

### 2.1 Installation / Update

#### 2.1.1 Windows

PreonLab runs on Windows 7 and higher versions. **Please note, that you will need administrative privileges in order to install, update or uninstall PreonLab on Windows. Therefore, we recommend to right-click onto the respective file and select "Run as administrator" to launch the process.**

**Installation:** PreonLab is provided to you as an installer program, e.g. *PreonLabInstaller.exe*. The graphical installer will lead you through the installation process.

**Update:** PreonLab automatically checks for updates during the startup process, if a network connection is established. If a new version is found, you will be informed about the changes of the new version. You can also manually check for updates using *PreonLabMaintenance.exe*.

**Uninstallation:** You can uninstall PreonLab using *PreonLabMaintenance.exe*. You can also use the "Programs and Features" tool from your system settings and choose PreonLab to uninstall.

#### 2.1.2 Linux

PreonLab runs on many different Linux distributions and is officially supported on RHEL/CentOS 6.6 and higher versions.

**Installation:** PreonLab is provided to you as an executable, e.g. *PreonLabInstaller*. It may be necessary to setup Execute file permission to the installer file in a terminal window:

```
chmod +x PreonLabInstaller
```

The graphical installer will lead you through the installation process.

**Update:** PreonLab automatically checks for updates during the startup process if a network connection is established. If a new version is found, you will be informed about the changes of the new version. You can also manually check for updates using *PreonLabMaintenance*.

**Uninstallation:** You can uninstall PreonLab using *PreonLabMaintenance*. You can also just delete the complete folder containing PreonLab, e.g. */opt/PreonLab*.

### 2.1.3 preonpy

The Python API PreonPy is integrated into PreonLab and PreonCLI, but it is also available as a regular Python package. See Section 19.2.4 for further instructions on how to install the Python package into an existing Python 3 installation. Also see Chapter 18 for information about PreonCLI and Section 19.3 about general usage of the PreonPy API.

PreonLab and PreonCLI both come with a Python 3.5 distribution, which contains the whole Python standard library. On Windows, the dependencies of all modules are included. On Linux, you may have to install some shared libraries using the system package manager or provide them otherwise. Currently, it is not possible to install additional packages into the bundled Python distribution. In this case you have to use a regular Python installation and install PreonPy as a regular Python package.

### 2.1.4 Systems without graphics

PreonLab requires a graphical desktop environment and a graphics card, that supports at least OpenGL 3.3. The command line version PreonCLI as well as the Python package do not require graphics. It can be used to run simulations or execute Python scripts that utilize the PreonPy API.

There is a separate download available that only contains PreonCLI and can simply be installed and used on a system without graphical desktop environment by unpacking the provided zip-file. PreonPy can be installed as a Python package using common Python tools (see Section 19.2.4).

## 2.2 Licensing

When you start PreonLab for the first time, you will be asked for a valid product license, confirm Figure 1. PreonLab uses the Reprise License Manager (RLM) and supports the following license types:

**Node-locked license:** This license is node-locked to a single computer and will not work on any other computer. It can be provided to PreonLab as a local license file or

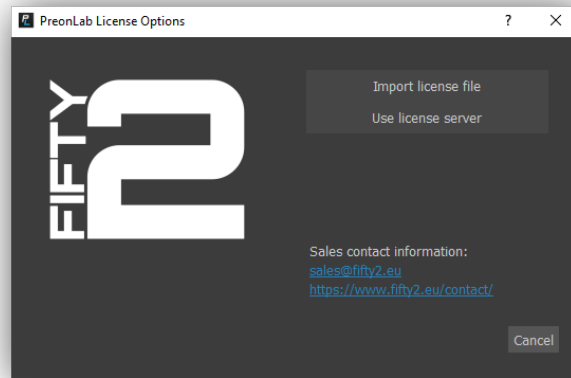


Figure 1: Licensing dialog.

via the RLM license server. If this is your type of license, you should have received a single license file (e.g. license.lic).

**Floating license:** A license which is served by an RLM license server. If this is your current licensing model, your IT administrator has received both, a license file and the specific FIFTY2 server settings. If your license server is within your LAN, it will be automatically found by PreonLab. The usage of floating licenses can be limited to a group of named users. If this is an option you want to employ, the licensor has to be informed before the licenses are issued.

The above listed license types can contain the full PreonLab package or a subset of components. Currently, the following options are:

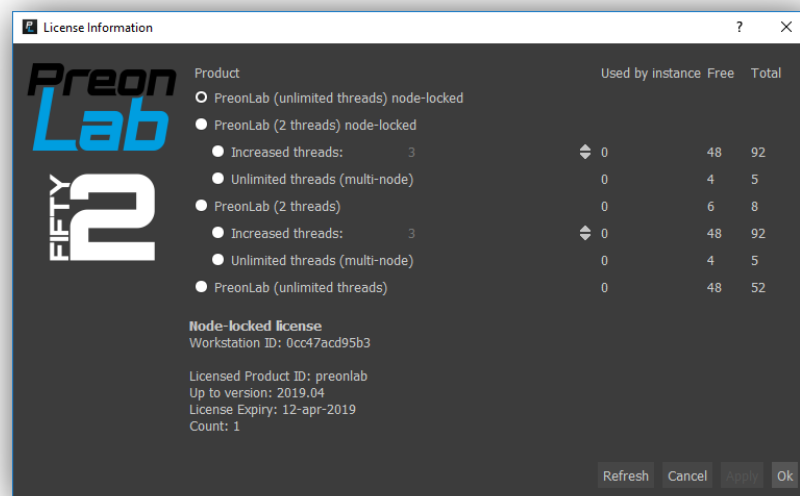
**Full license:** This license contains all components of PreonLab.

**Prepost license:** The prepost license is used for scene setup and postprocessing tasks. You can also simulate, but with a reduced number of simulation threads. The PreonCLI and the Python interface PreonPy is also included.

In general, licenses only allow a single running instance on one machine and can either have unlimited simulation threads or, like the prepost licenses, include a limited number of threads. In the latter case, additional *boost* licenses increase the number of simulation threads. Note that the *full* license is only unlimited on a local workstation. For distributed simulations with PreonNode it provides 32 threads that can be shared on multiple nodes. The number of simulation threads can be increased with one or multiple *thread* licenses or with a single *MPI unlimited* license.

PreonLab first tries to checkout the set of licenses, that were used in the last PreonLab session. If this is not possible, it will try to find another suitable license and warn about the implications. Preferably it will checkout a node-locked license. If no node-locked license is available, a suitable floating one is used. You can view license information and switch between different licenses in PreonLab via *Help*→*License Information* (see Figure 2). You can also see which floating licenses are available in total,

free to checkout and used by the current PreonLab instance.



**Figure 2:** License information dialog showing current checked out license and all available licenses. Dialog can be opened via *Help*→*License Information*.

See chapters 18, 19 and 20 for more information on how to handle licensing in PreonCLI, PreonPy and PreonNode.

## 2.2.1 Managing license files

If you have not yet imported any license, PreonLab will let you choose between different options for licensing. You can either import a local license file or enter the address of the RLM license server, if this is not automatically broadcasted in your network. Reprise provides an extensive manual for license administrator and users that can be found at <http://www.reprisesoftware.com/admin/software-licensing.php>.

PreonLab stores license files in the file `C:/Users/<USER>/AppData/Local/PreonLab/PreonLabLicense.lic` (Windows) or `~/.config/PreonLab/PreonLabLicense.lic` (Linux). In certain situations it is necessary to manually touch this file. For example, on machines without a graphical user interface, you can not use the PreonLab dialog for setting up the license configuration. You can then place the license file at this location yourself. For pointing PreonLab to a specific RLM license server, the file looks like follows:

```
HOST <URL or IP address of license server> ANY 5053
```

5053 is the default port of RLM. You may change it, if you run it on a different port. You can also copy a working file from a machine with the PreonLab GUI.

Alternatively, you can use an environment variable to specify the location of a license file or to directly specify the port and host of the license server. For this, either the



*fifty2\_LICENSE* or *RLM\_LICENSE* environment variables work. Accordingly, on Linux, you could set the the license server as follows:

```
export fifty2_LICENSE=5053@<URL or IP address of license server>
```

Or you could set the path to a license file as:

```
export fifty2_LICENSE=<path to license file>
```

You can find more details under the first question "What is the order of license files processed by my application?" at the [RLM License FAQ](#).

## 2.2.2 Installation of an RLM license server

1. Please install an RLM server on the computer with the MAC-address for which the license was created. You can download the RLM server from here:

<http://www.reprisesoftware.com/admin/software-licensing.php>.

2. Download two more files from our transfer server. The link has been sent to you via email along with the license file. On Linux, download the files *fifty2* and *fifty2.set* from the **RHEL 6:7** folder. Note, that the file *fifty2* has to be executable. On Windows, the files *fifty2.exe* and *fifty2.set* are found in the **Windows** folder.
3. Put your license file (ends with .lic and was sent to you by email) and the two files described above into the installation folder of the RLM server you have installed. Afterward, the server has to be started and should then be ready to use.
4. Now, when starting PreonLab on a computer inside your LAN, PreonLab should automatically find the license server. If this does not work, try to enter the address of the license server when prompted.

## 2.2.3 Restricting license access

Sometimes it's desirable to restrict certain licenses to a specific user or machine group, e.g. to prevent users from consuming *full* licenses on their workstations while the cluster could make better use of them. This is possible using some custom configuration on the license server in a so called ISV options file.

There are three basic types of restrictions that work with PreonLab. **INCLUDE/INCLUDEALL** makes it possible to restrict license access to a set of users or machines. The inverse can be achieved with **EXCLUDE/EXCLUDEALL**. With **MAX** you can set an upper limit of licenses that can be used by the specified group.

The set of users/machines can be defined in terms of user names, host names, IP addresses or projects. While projects also need an environment variable on the client side, the others are defined only in the options file.

These restrictions can either be applied to all licenses, to all licenses of a certain type (like *Full license*) or all licenses belonging to one specific line in a license file. In the last case an ID has to be added to the respective line in the license file.

## Example

Assuming we have three *full* and three *prepost* licenses. The *full* licenses should be used by some cluster machines consisting of nodes *compute1*, *compute2* and *compute3*. Additionally, user *john* should also have access to one specific *full* license and the machine with IP address *192.168.26.26* should use at most 1 *prepost* license.

This can be achieved using the following configuration in the options file:

```
HOST_GROUP cluster compute1 compute2 compute3
INCLUDE preonlab user john id=52
INCLUDE preonlab host_group cluster
MAX 1 preonlab_prepost internet 192.168.26.26
```

The ID 52 used in the second line has to be assigned in the license file using the `_id` parameter.

```
LICENSE fifty2 preonlab 2019.12 20-dec-2019 1 ... _id=52
LICENSE fifty2 preonlab 2020.06 20-jun-2020 2 ...
LICENSE fifty2 preonlab_prepost 2019.12 20-dec-2019 3 ...
```

Note that the **RESERVE** option is currently not supported by PreonLab.

## 2.2.4 Troubleshooting

Checkout [https://www.reprisesoftware.com/RLM\\_Troubleshooting\\_Tips.pdf](https://www.reprisesoftware.com/RLM_Troubleshooting_Tips.pdf) for a guide on how to investigate typical licensing related issues. For instance, when using PreonCLI or PreonNode it can be useful to set the `RLM_DIAGNOSTICS` environment variable to check licensing settings. Also consider [https://www.reprisesoftware.com/RLM\\_License\\_Administration.pdf](https://www.reprisesoftware.com/RLM_License_Administration.pdf) for more detailed information.

## 2.3 Environment variables regarding multi-threading

PreonLab, PreonCLI, PreonNode and also the PreonPy Python package consider a set of environment variables that affect the way multithreading works inside. They are evaluated and used by OpenMP, more specifically libgomp on Linux and Microsofts implementation on Windows. The documentation of the environment variables can be found [here for Linux](#) and [here for Windows](#).

In general, those settings don't need to be touched, except for the cases listed below and if you know what you're doing.

### 2.3.1 Setting the number of threads

You can specify the number of used threads using the `OMP_NUM_THREADS` environment variable. For instance, this would set the number of threads to 8:

```
export OMP_NUM_THREADS=8 (Linux)
set OMP_NUM_THREADS=8 (Windows)
```

Please note that this can be overridden by threading settings in the scene file if the **individual #threads** property is enabled in the scene.

### 2.3.2 Performance on multi-socket systems

If you are on Linux and run one of our applications on a multi-socket system, we recommend to specify the following OpenMP environment variables before launching the simulation:

```
export OMP_PLACES=sockets
export OMP_PROC_BIND=close
```

Note, that this can interfere with other settings, e.g. if there is already a thread affinity mask set for the calling process.

### 2.3.3 Troubleshooting

In our experience, problems often occur because environment variables were set without the knowledge of the user. It therefore may be interesting to look into the values of those variables. On Linux this can be achieved by setting the variable `OMP_DISPLAY_ENV` to `VERBOSE` and running e.g. `PreonCLI -v`.

It also may be interesting to look at the thread affinity mask using `taskset -pc <process id>` with the `process id` from the running process. With this command you can identify if the process is allowed to run on all cores. It may be worth it to have a closer look at the environment variables mentioned above if this is not the case.

### 3 PreonLab GUI general usage

This chapter gives an overview over the basic components and subwindows of PreonLab. It explains how to navigate in the graphics window and introduces the various object tools.

#### 3.1 User interface overview

The user interface is exemplified in Figure 3.

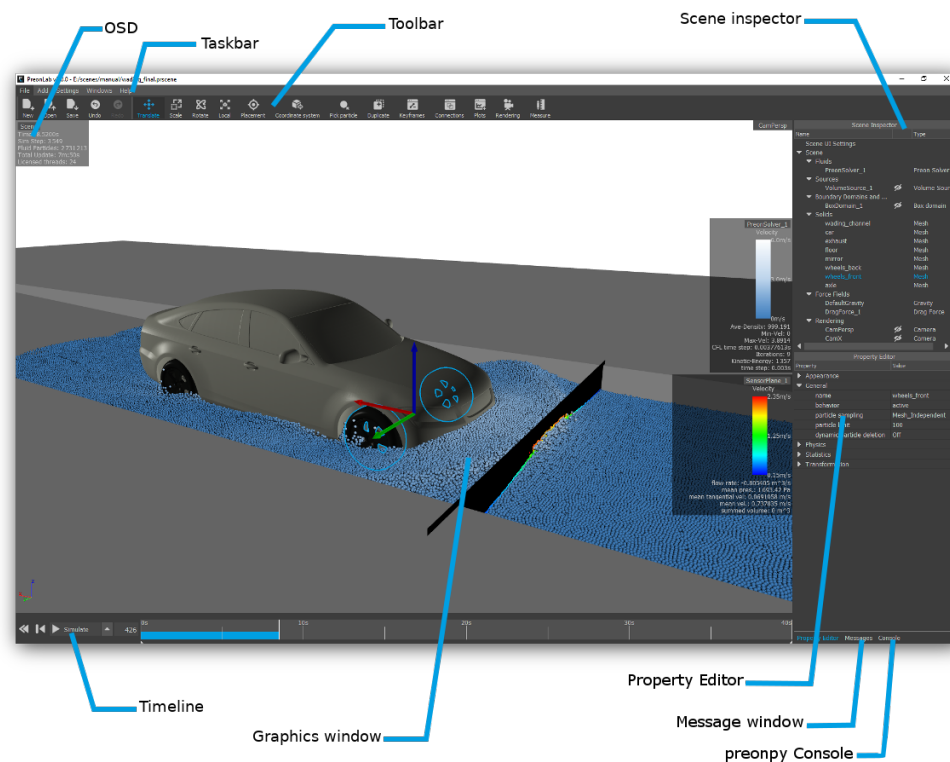


Figure 3: PreonLab user-interface.

## 3.2 Graphics window

The graphics window displays the scene viewed from your current camera in single view and for the four cameras in quad view. You can switch from single view to quad view by hitting the space key of your keyboard.

### 3.2.1 Navigating in the graphics window

You can directly interact with the scene using your mouse. For some of these actions, you need to press the **control key** on the keyboard simultaneously. By default, the **control key** is SHIFT. This can be changed in *Settings*→*User Preferences*→*control key* (see Section 3.12).

Control	What it does
mouse wheel or control key + right mouse button + mouse move	Zoom in and out.
control key + left mouse button + mouse move	Rotate the camera.
control key + mouse wheel pressed + mouse move	Translate the camera (panning).
left mouse button click	Select object under mouse pointer.
left mouse button + move mouse	Select multiple objects which are within the drawn rectangle.
mouse wheel click	If the mouse pointer is over an object, the camera pivot is set such that the camera can be rotated around the corresponding point on the object.
space	Switches between single and quad view.

Table 1: Controls for navigating in the graphics window.

## 3.3 Toolbar



**Figure 4:** PreonLab toolbar. The toolbar is context sensitive and shows different options depending on the selected objects.

The toolbar in PreonLab is separated into a File toolbar and a Scene toolbar. The File toolbar is located on the left and has actions like creating a new scene and undo and redo. The Scene toolbar is context sensitive and shows actions and tools related to

the current scene and the currently selected object. Some of them are tools to move, rotate and scale objects in space, and to measure a distance between any two points on objects in the scene. There are shortcuts for some toolbar actions which can be found in Section 3.11.

### 3.3.1 Translate

Shows a dragger tool to translate the currently selected object(s). You can choose between moving along the unit axis or along the local rotated axis by toggling the *Local* button in the toolbar. If you hover with the mouse over one of the dragger axes it is highlighted in yellow. This also holds for the other dragger tools.

### 3.3.2 Scale

Shows a dragger tool to scale the currently selected object(s) symmetrically either along the unit axes (red, green, blue) or uniformly (white axis). When the **control key** is pressed, the draggers allow scaling in one direction only. By default, the **control key** is SHIFT. This can be changed in *Settings*→*User Preferences*→*control key* (see Section 3.12).

### 3.3.3 Rotate

Shows a dragger tool to rotate the currently selected object(s). You can toggle between rotating around the unit axis or around the local rotation axis by toggling the *Local* button in the toolbar.

### 3.3.4 Particle picking

By default, the particle picking mode is disabled and clicking on any particle selects the whole object (fluid or solid) to which this particle belongs to. In order to select a single fluid or solid particle with the mouse, activate the particle picking mode. This will display the physical values as well as the ID of the picked particle in the OSD. The ID of a particle is relevant for the **particle tracker** which is required to also save and plot the statistics of a single fluid particle, see Section 16.7.

### 3.3.5 Placement tool

The placement tool places the currently selected spatial object onto another spatial object on a left mouse button click. This tool can only be activated and is only shown if exactly one object is selected. If the mouse cursor is not hovering over any spatial object on click, or if the mouse was moved between pressing and releasing the left mouse button, no action is performed.

### 3.3.6 Measure tool

The measure tool allows you to measure the distance between two points in the scene. When activated, every two left mouse button clicks on spatial objects or fluid particles will show the distance between the two defined points on the screen overlay under *MeasureTool*. When both these measure points are set, an additional button appears allowing to save the current measurement. Saving a measurement results in both measure points appearing in the scene inspector, connected to a new distance sensor, while the *Transform* output of the spatial object(s) where the points were placed on is also automatically connected to the points. This simplifies the procedure described in 16.3.

## 3.4 Taskbar

The taskbar has multiple categories. Under *File*, there are entries related to the scene. In *Add* you can add objects to the current scene. *Settings* has entries for user preferences and for changing properties of the collider and the rigid body solver. *Windows* allows you to change which parts of the user interface are displayed. Via the *Help* menu you can open an *About* dialog which shows the current version of PreonLab and a link to the log file for the current session. *Help* also has an entry that shows license information.

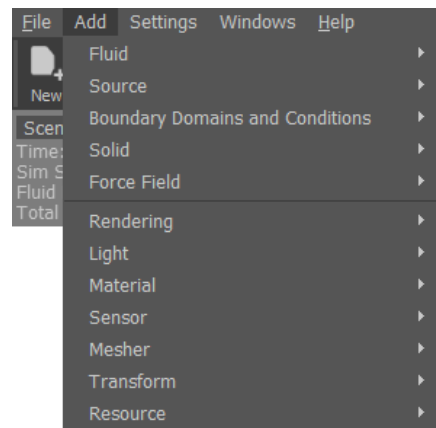


Figure 5: Taskbar.

## 3.5 Timeline

The graphics window displays a view or multiple views of your scene for one point in time. You can navigate in time and start playback, post-processing or simulating via the *Timeline* (see Figure 6).

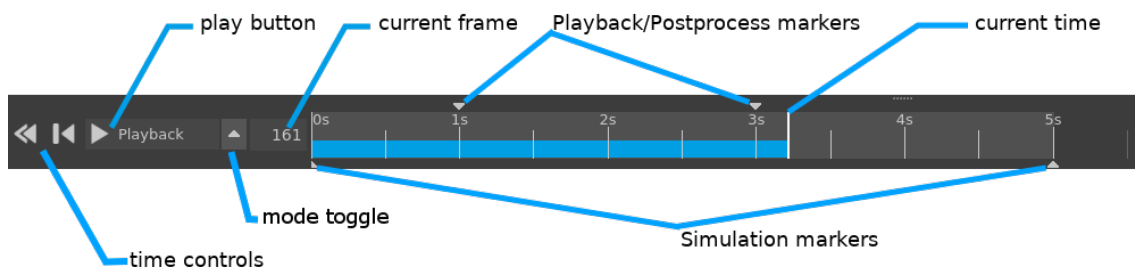
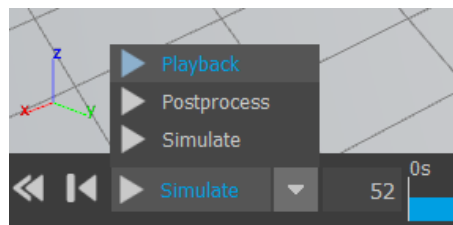


Figure 6: Timeline with controls on the left and an overview of several time-related functions.

The timeline shows you the simulation time range in a lighter gray. It is defined by the two arrow markers at the bottom which can be moved via drag and drop or by changing **simulation start** or **simulation end** in the scene properties. The current time is highlighted with a white vertical marker. The blue horizontal bar indicates which part of the simulation time range has already been simulated. The arrow markers at the top define the time range for the Playback and Postprocess mode.

You can jump in time either by clicking on some point in the illustrated time interval or by editing the *current frame* field. Here, you can either enter a frame number or a time formatted as follows: *1mo:2d:3h:4s:5ms*. This will set the white vertical marker accordingly. Hovering with the mouse over the timeline displays the respective time and corresponding frame for the current mouse position. Hovering over an arrow marker displays the exact time assigned to that marker.

On the left side of the timeline there are additional controls to jump in time and a play button. The play button allows you to either (i) start a playback of your scene, (ii) start post-processing the scene or (iii) start simulating your scene. The mode can be changed by clicking on the arrow next to the play button as shown in Figure 7.



**Figure 7:** Clicking on the downward facing arrow next to the play button opens a list in which you can choose the mode: Playback, Postprocess or Simulate.

**Playback:** When starting playback, PreonLab will successively load and display the single frames of your scene. PreonLab will not overwrite any data on disk apart from recording images if this is enabled.

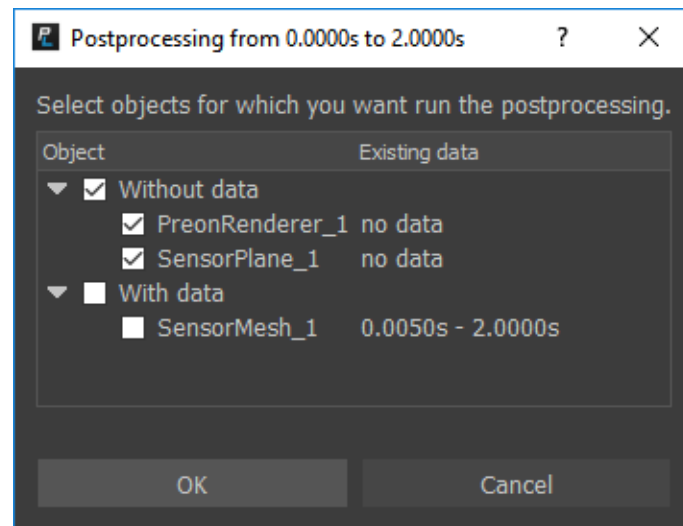
**Postprocess:** This mode allows you to gather data for postprocessors like sensors on already existing simulation data. When you start a postprocessing run, a dialog is shown that allows you to select which postprocessors should update their data in this postprocess run. An example for this dialog is shown in Figure 8.

**Simulate:** This will start or resume the simulation from the **simulation start** or current time. Fluid dynamics are only computed in this mode. Active sensors will gather data during simulation as well.

When clicking on the play button, PreonLab will execute the current mode until you either click the play button again (which shows a *pause* symbol during a run) or until either the **playback/postprocess end** or the **simulation end** (depending on the current mode) is reached.

By pressing your defined **control key** while hovering with the mouse over the play button, you can switch between the normal play mode and single step mode. In





**Figure 8:** The postprocess dialog allows to select which postprocessors should be updated in a postprocessing run and which not.

single step mode, each time you click the play button, PreonLab will only perform a single step of playback/post-process/simulation.

**Context menu:** Right-clicking on the timeline displays a context menu with four additional options. First, you can adjust the displayed time interval to the simulation end time by selecting *Zoom to simulation end*. The other three options delete existing simulation data. You can either delete the simulation data for the whole simulation, before the currently loaded frame or after the currently loaded frame. These actions delete simulation data but do not remove rendered images. The scene itself is not changed, meaning also that resource files like mesh files of solid objects are not deleted.

### 3.6 Scene Inspector

The scene inspector lists all objects that are in the scene by their name and categorizes them into different groups. The color of an object name shows the status of the object: active (white), inactive (orange), cached (yellow). You can select any object by left-clicking on it. When right-clicking an object, a context menu is shown with object-specific actions, e.g., export of sensor data. You can also select multiple objects by simultaneously pressing the CTRL key and left-clicking objects subsequently. Controls for the scene inspector are listed in Table 2. Please also refer to Table 3 for additional shortcuts that can be used all over PreonLab.

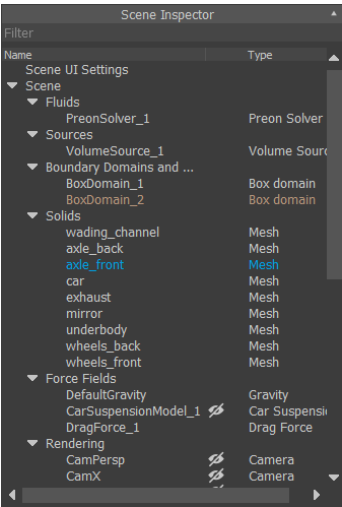


Figure 9: Scene Inspector.

Control	What it does
right mouse button click	Selects the object under the mouse pointer and shows the context-menu with object-specific actions if available.
CTRL + left mouse button clicks	Select multiple objects.
SHIFT + left mouse button click	Select all objects between already selected object and clicked object.
CTRL + a	Select all objects in scene inspector.

Table 2: Controls for scene inspector.

On top of the object list you find a search field (see Figure 9 where you can enter (partial) object names to quickly filter the list to objects which have the search string as (part of) their names. The search is case-insensitive. In large scenes with thousands of objects, this helps to reduce the list to a manageable size. Click the X icon to the right of the search field to clear the search string.

If you want to select all children of a specific object category, i.e., all **Solids**, right-click on the respective category and choose *Select all children* from the context menu. Note that if you have filtered the list of objects before, the action changes to *Select all filtered children* and does exactly that.

Furthermore, the context menu provides actions to *Show/hide* objects in the graphics window, too. With these you can show/hide selected objects, selected objects of a specific category, all objects of a specific category, or all objects of the entire scene.

### 3.7 Property Editor

The property editor displays the editable properties for the selected object. If multiple objects are selected, the union of properties is shown. In this case, for a common property with differing values, the default value of the property is shown. The properties are grouped semantically. Groups can be expanded or collapsed. Multi-dimensional properties like positions are also represented as expandable groups. You need to expand the group and edit each dimension individually.

A tooltip is displayed if you rest the mouse pointer over a property. Furthermore, the property editor is context sensitive, which means that some (sub-)properties might be hidden and only displayed if the dependent parent property is set.

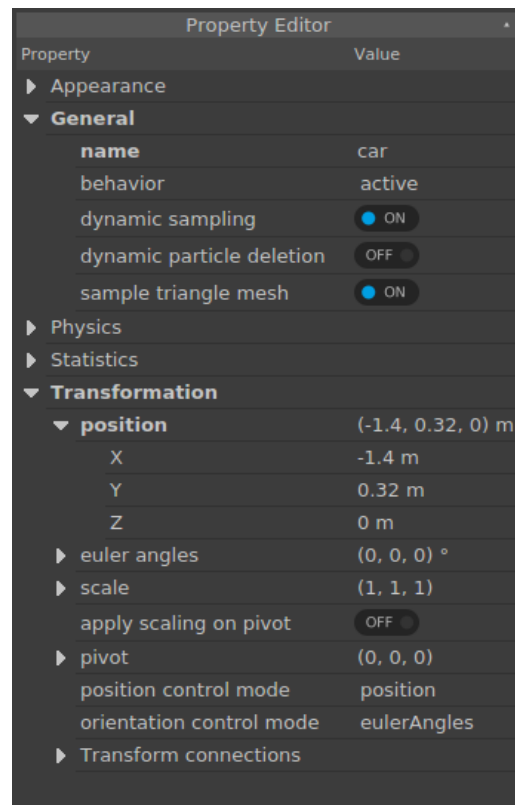


Figure 10: Property Editor.

**Default values of properties:** Properties which deviate from their default value are printed in bold letters. Similarly, a property group which contains at least a property that deviates from its default value is also printed in bold letters. Properties can be reset to their default value by right-clicking on the property and choosing *Set to default value*. All the properties of a group can be reset to their default values by right-clicking on the group and selecting *Set all sub-properties to their default values*.

**Keyframed properties:** Properties that are keyframed (see Chapter 6) are recognizable by a colored background. In that case, a green background indicates the presence of a keyframe at the current point in time, yellow indicates that the value displayed has been interpolated from the keyframes, and a red background warns that the value has been manually changed by the user and will not be used for the simulation if no additional keyframe is inserted at that specific point in time. There are also properties that can be keyframed based on particle values, e.g, temperature-dependent viscosity. These properties are displayed as shown in Figure 11. A red color indicates that no keyframes are set yet. Clicking the icon opens the keyframe editor for the respective property.

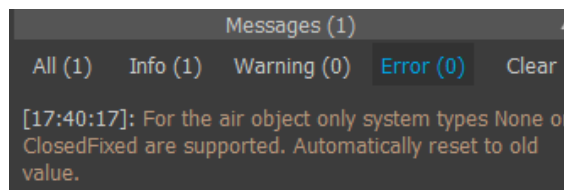


**Figure 11:** Properties that are not keyframed based on time are indicated in the property editor as seen above.

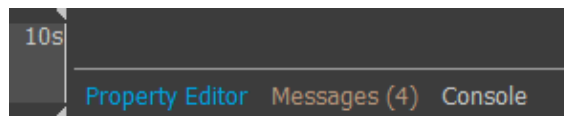
### 3.8 Message window

PreonLab prints three different types of messages: *Information*, *Warning* and *Error*. These messages are printed and grouped accordingly in the message window. By default, the message window is tabbed next to the *Property Editor* and not visible. Make the message window visible by clicking on *Messages*. A number is given for each message type which shows how many new messages of this type are new (unread messages), for example as shown in Figure 12. If the message window is not visible, a number is added to the tab title *Messages* which text is colored red if there are unread error messages, orange if there are unread warnings as visualized in Figure 13.

Note that all messages are printed to the *log.htm* file of your current PreonLab session. The location of this file is shown in the *Help*→*About* dialog.



**Figure 12:** Message window with one unread information message and older error messages.

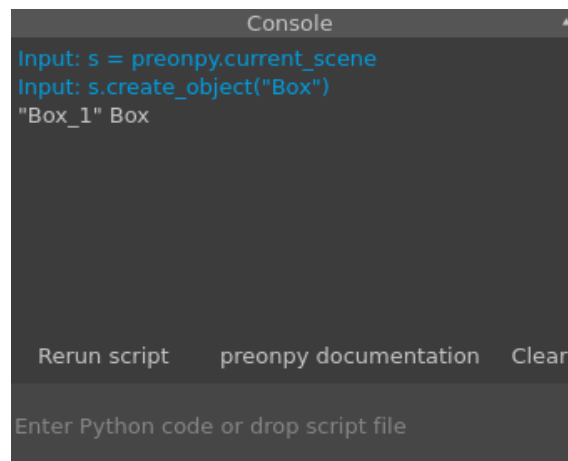


**Figure 13:** Message count indicates new messages.

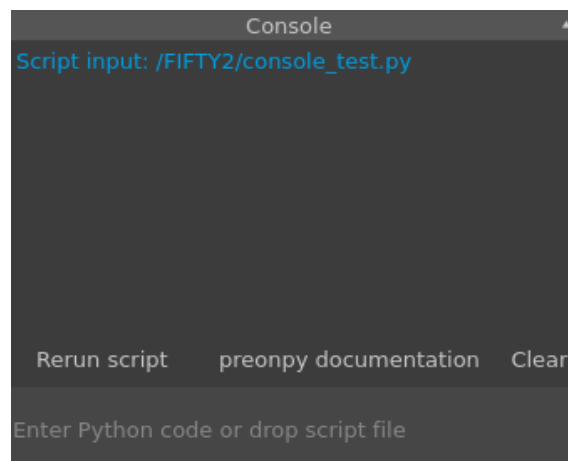
### 3.9 Console

In PreonLab's built-in console you can make full use of the PreonPy API described in Chapter 19. You can either enter single commands one after another like in Figure 14 or enter an entire script file via drag and drop, see Figure 15. Pressing the *Rerun script*

button opens a list of previously run scripts to conveniently rerun one of them. A click on *preonpy documentation* opens the full documentation of PreonPy in your browser. *Clear* simply clears the console window.



**Figure 14:** Two PreonPy commands entered one after another to create a box in the current scene.

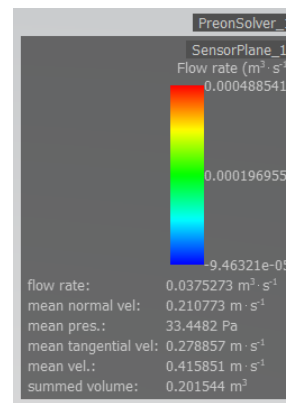


**Figure 15:** Entering a script file via drag and drop runs the script and shows the file path.

### 3.10 OSD (On-Screen-Display)

Simulation related statistics and computation times can be displayed in the graphic window as an overlay (OSD). The OSD can be completely disabled by setting **Scene UI Settings**→**Appearance**→**show osd** to off or OSD elements can be enabled/disabled and adjusted on a per-object level. The corresponding properties can be accessed for each object in the property editor **Statistics**→**OSD Settings**.

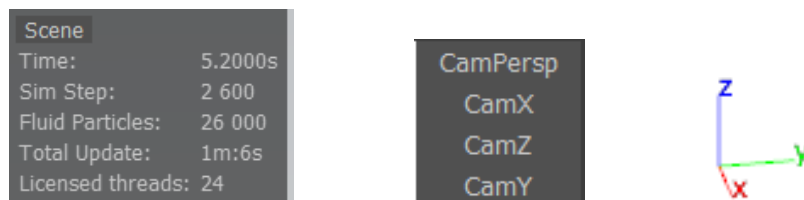
The OSD groups content for each object showing the object name in the title of the OSD element. Figure 16 shows the collapsed OSD element of a Preon solver and the expanded OSD for **Sensor plane**. OSD elements are collapsed and expanded by left-clicking on the object name in the OSD element.



**Figure 16:** OSD for solver and collapsed sensor.

OSD elements of objects can be arranged at various predefined positions in the graphics window. The top-left corner is reserved for the **Scene** which by default lists the current time (matches time in timeline), the simulation step, the number of fluid particles in the scene at the current time, and the total computation time from start to current time. The OSD element for the **Scene** is illustrated in Figure 17.

In the top-right corner you find the camera menu. You can select the active camera by clicking on this menu and select any camera from the menu.



**Figure 17:** OSD elements for **Scene**, camera menu and global coordinate axis.

### 3.11 Keyboard shortcuts

The following keyboard shortcuts apply throughout PreonLab.

Key	What it does
w	Enters / leaves translation mode.
e	Enters / leaves scale mode.
r	Enters / leaves rotation mode.

<b>p</b>	Enters / leaves placement mode (if available).
<b>m</b>	Enters / leaves measure mode.
<b>DEL</b>	Deletes the selected object(s).
<b>CTRL + d</b>	Duplicates the selected object.
<b>CTRL + c</b>	Copy the selected object(s) into your clipboard. These objects can be pasted into the same or another PreonLab instance.
<b>CTRL + v</b>	Pastes previously copied objects.
<b>CTRL + h</b>	Changes render mode of selected objects from <b>smoothShaded</b> to <b>wireframe</b> , from <b>wireframe</b> to <b>invisible</b> and from <b>invisible</b> to <b>smoothShaded</b> .
<b>CTRL + b</b>	Changes behavior of selected objects from <b>active</b> to <b>cache</b> , from <b>cache</b> to <b>inactive</b> and from <b>inactive</b> to <b>active</b> .
<b>CTRL + k</b>	Creates a transformation keyframe (position, orientation, scale) at the current frame for the selected object.
<b>CTRL + s</b>	Save current scene.
<b>CTRL + SHIFT + s</b>	Save current scene under a different location or with different options.
<b>CTRL + n</b>	Create a new scene.
<b>CTRL + o</b>	Open a scene.
<b>CTRL + q</b>	Close PreonLab.
<b>CTRL + z</b>	Undo last operation, except camera manipulation.
<b>CTRL + y or CTRL + SHIFT + z</b>	Redo operation. Note that this is system dependent.
<b>CTRL + g</b>	Takes a screenshot of the currently active graphics window. The screenshot will be saved in the scene directory in the following subfolder: <i>Visualization/OpenGL/[NameOfCamera]</i>

**Table 3:** List of keyboard shortcuts.

### 3.12 User Preferences

The *User Preferences* can be found under the menu *Settings* in the task bar (see Figure 3). They allow you to specify several preferences that are used when creating a new scene, as well as a *maximum number of threads* used by PreonLab. These preferences are saved to your disk and restored when starting PreonLab. While the default values specified here are assigned to newly created scenes, they still can be modified per scene later.

Preference	What it does
<b>OSD background color</b>	The background color of the OSD overlay.

<b>rendered particles target</b>	Sets the target number of rendered particles (in millions) per fluid solver object. If necessary, particles will be downsampled adaptively to a coarser resolution to match the target. If particle downsampling is employed, the whole fluid is rendered using flat shading to hide the transitions between particles of different sizes.
<b>default scene directory</b>	The directory where new scenes are created.
<b>default up axis</b>	The up axis (either z-axis or y-axis) used for new scenes.
<b>show grid by default</b>	Defines whether the orientation grid should be drawn by default in newly created scenes.
<b>default background color</b>	New scenes will be created with this color as the first background color.
<b>default background color 2</b>	New scenes will be created with this color as the second background color. If this color differs from the first background color, a color gradient will be rendered.
<b>default video export directory</b>	For new scenes, videos will be exported to this directory by default.
<b>maximum number of threads</b>	The number of CPU threads used by PreonLab will never exceed this maximum number. This correlates with <i>individual #threads</i> and <i>#threads for this scene</i> in the scene properties, i.e., a lower number of threads may be specified per scene.
<b>experimental features</b>	If on, experimental features of PreonLab are exposed. See Section 3.14 for more details.
<b>saves scenes before simulation or post-processing</b>	If on, scenes will be saved before starting to simulate or post-process. Untitled scenes will not be saved. Note that if enabled, PreonLab will overwrite your existing scene without any confirmation request upon starting a simulation or post-process.
<b>control key</b>	The key that needs to be pressed and held to access additional actions, e.g. camera control (see Table 1). The default is SHIFT.

**Table 4:** The user preferences to define.

### 3.13 Presets

For some object types, presets are available that contain common property configurations for the object. Examples include camera configurations, fluid particle coloring schemes or material settings for rendering. To apply a preset, right-click on an object in the scene-inspector, select *Set preset* and then choose the preset you want to apply. Right-clicking in the graphics window will display the same menu for the currently selected object. If there is no *Set preset* option, there are no presets available for the object.



## 3.14 Experimental features

PreonLab includes experimental features that are only available if they are activated in the user preferences (c.f. Section 3.12). The experimental features are still in prototype phase and under development. Their functionality and effects may still change drastically in the future. Accordingly, these features are not visible by default.

## 3.15 Start parameters

PreonLab can be started with optional start parameters. The available parameters are listed in Table 5.

Start parameter	What it does
<code>--scene &lt;filename&gt;</code>	Directly loads the given <i>&lt;filename&gt;</i> as scene. If <i>&lt;filename&gt;</i> is not a valid scene file, an empty untitled scene will be opened.
<code>--noGeometryShaders</code>	Prevents PreonLab from using geometry shaders. This may help on some Linux-based systems if various artifacts appear in the graphics view. See Section 3.16.1 for more information.
<code>--skipglxtest</code>	On Linux, PreonLab executes an OpenGL version test on startup using <i>glxinfo</i> to make sure the graphical system is matching the requirements. When providing this start parameter, the test is skipped.

Table 5: Possible start parameters for PreonLab.

## 3.16 Known issues and workarounds

### 3.16.1 OpenGL

On some Linux-based systems and systems with weak graphics hardware, e.g., on-board graphics processor, PreonLab may show various artifacts when displaying lines. Usually, this can be fixed by starting PreonLab with the option `--noGeometryShaders`. The only restriction of using this flag is that all lines will have a width of one pixel.

### 3.16.2 Display resolution

Changing the resolution of your display while PreonLab is running might lead to some inconsistencies in the user-interface. This is fixed when PreonLab is restarted.

### 3.16.3 Drag & drop

On Windows, the option to drag & drop files (e.g., geometry files or Python script files) onto PreonLab does not work if the application is running elevated (e.g., via *Right click*→*Run as Administrator*). In this case, the file explorer does not have the same integrity level assigned and the Windows message for the drag & drop is blocked. The easiest workaround is to not run PreonLab elevated.

### 3.16.4 Starting PreonLab via Windows Remote Desktop

#### Problem description

When using Windows Remote Desktop (RD) for login to a remote machine the following can be observed:

- PreonLab instances already running on the remote PC can be worked with
- starting PreonLab via RD does not work (i.e., does not open)

Reason is that if opened via RD, the PreonLab instance is housed within RD and employs its graphics drivers. However, PreonLab requires a higher version of OpenGL for the display of the graphics window than is provided by RD.

#### Current workaround

**Warning:** This workaround only works with users that have administrative rights.

First, we need to get the ID of the active user:

- Login via RD
- Open the Windows command line or Windows PowerShell
- Enter *query user* and look at the result, see Figure 18.
- Remember the ID of the active user (the one with the *,>'* in front. In the example shown in Figure 18, the ID is 2.

```
Microsoft Windows [Version 10.0.18363.720]
(c) 2019 Microsoft Corporation. Alle Rechte vorbehalten.

C:\Users\remote>query user
BENUTZERNAME      SITZUNGSNAME      ID  STATUS  LEERLAUF  ANMELDEZEIT
user              1               Getr.      26  19.03.2020 10:20
>remote          rdp-tcp#11       2  Aktiv   .         19.03.2020 11:56
```

Figure 18: Possible result for querying the user in a RD session.

Next, we create a file on the desktop on your remote computer and name it *PreonLab\_remote.bat*. Make sure that *.bat* really is the file extension. Adapt the following code such that you end up on the correct partition and execute the PreonLab instance you want to have started. Furthermore, change the number in the first line of the script to the ID you have noted down previously.

```
tscon 2 /dest:console
C:
cd \
cd "C:\Program Files (x86)\PreonLab\PreonLab.exe"
```

Finally, run the script:

- Execute the batch file as an administrator.
- Right-click on the file and choose the respective action. Reminder: The user you have logged in must be identical to the one running this script.
- You will be logged out of the RD session, so login again.
- As soon as you are back, you should find a running PreonLab instance.

**Warning:** If you do not login again after running the script and logout after you have finished working, there is a logged in user on your remote machine which might pose a security threat!

As a final note, make sure that the PreonLab version is either the latest one, or does not check for updates when started. Otherwise, a PreonLab upgrade window pops up which breaks the whole process. Reason is that after pressing the OK button, the PreonLab instance will again be started within RD.

## 4 Common properties

In this chapter, some common properties are explained that are shared by many object types.

### 4.1 General

Property	What it does
behavior	Controls the behavior of the object during simulation and playback. <b>active</b> means that the object is part of the simulation and will save simulation data to disk when simulating. <b>inactive</b> is the opposite, the object will be completely ignored during simulation and playback. <b>cache</b> means that the behavior of the object is determined by data read from disk. Cached objects will therefore never be influenced by other objects, but they can influence other simulated (active) objects.

Table 6: Properties in group General.

### 4.2 Appearance

The following properties are shared by many objects that are visualized in the graphics window or by the Preon renderer such as solid objects and fluids.

Property	What it does
render mode	Changes the way the object is displayed. <b>smoothShaded</b> is the default setting and will enable smooth per-pixel lighting. <b>wireframe</b> is only available for meshes and displays the wireframe rendering of the mesh triangles. <b>invisible</b> will hide the object completely.
color	The color of the object. Note that depending on the object and its properties, this color may be overridden by another visualization, for example when visualizing the fluid velocity using a color gradient.

<b>show bounding box</b>	Defines whether the bounding box of the object should be drawn. If enabled, the extents of the object on the x-, y- and z-axis are shown as additional read-only properties.
<b>opacity</b>	Value between 0 and 1 determining the opacity of the object. Note that if you want to hide the object completely, it is recommend to set <b>render mode</b> accordingly for better performance.
<b>exclude from reflections</b>	If enabled, this object will not be mirrored in reflective surfaces when using Preon renderer. Mainly intended for sensors.

Table 7: Common properties in group Appearance.

## 4.3 Transformation

Property	What it does
<b>position control mode</b>	Determines how the position for the object is specified. The default is <b>position</b> , which lets you specify the position directly. It is also possible to control the position using its derivatives by choosing <b>velocity</b> or <b>acceleration</b> .
<b>position</b>	The local position of the object in x, y and z coordinates. Only available if <b>position control mode</b> is set to <b>position</b> . Local means that the position is always relative to the transform parent. If there is no transform parent, it is equal to the global position (see below)
<b>global position</b>	The global position of the object in x, y and z coordinates. This is a read-only property and it is mainly used to make the global position of objects available to the python system for scripting purposes.
<b>velocity</b>	The local velocity of the object. Only available if <b>position control mode</b> is set to <b>velocity</b> .
<b>start position</b>	The local position of the object at time 0. Only available if <b>position control mode</b> is set to <b>velocity</b> or <b>acceleration</b> .
<b>acceleration</b>	The local acceleration of the object. Only available if <b>position control mode</b> is set to <b>acceleration</b> .
<b>start velocity</b>	The local velocity of the object at time 0. Only available if <b>position control mode</b> is set to <b>acceleration</b> .
<b>orientation control mode</b>	Determines how the orientation for the object is specified. You can either enter the rotation as Euler angles (choose <b>eulerAngles</b> ) or as a rotation around an axis (choose <b>revolution</b> ). You can also specify rotations per second around an axis by choosing <b>revolutions_PerSecond</b> .
<b>revolution axis</b>	The local axis around which the object should rotate. Only available if <b>orientation control mode</b> is set to <b>revolution</b> or <b>revolutions_PerSecond</b> .

<b>revolution</b>	Sets the rotation around the chosen axis. Zero means no rotation, one means one full revolution (360 degree), 0.5 means rotation of 180 degrees and so on. Only available if <b>orientation control mode</b> is set to <b>revolution</b> .
<b>revolutions per second</b>	Sets the revolutions per second around the chosen axis. Only available if <b>orientation control mode</b> is set to <b>revolutions_PerSecond</b> .
<b>revolution start</b>	Sets the state of the revolution around the chosen axis at time 0. Only available if <b>orientation control mode</b> is set to <b>revolutions_PerSecond</b> .
<b>euler angles</b>	The local orientation of the object expressed in rotation around the x-axis (phi), rotation around the y-axis (theta) and rotation around the z-axis (psi). All rotations are in degrees. The values are interpreted as extrinsic Tait-Bryan angles. Only available if <b>orientation control mode</b> is set to <b>eulerAngles</b> .
<b>scale</b>	The scale of the object as a three-dimensional vector.

Table 8: Properties in group Transformation.

Property	What it does
<b>adjust own transform</b>	Specifies whether the local object transformation may be modified when setting up a transform parent. If enabled, the transformation of the object will be adjusted so that it always keeps its global transformation.
<b>adjust child transform</b>	Specifies whether the local transformations of children may be modified when creating or deleting a transform connection. If enabled, the transformation of child objects will be adjusted so that they always keep their global transformation.
<b>inheritance mode</b>	Specifies whether the object inherits position, orientation or both (all) from its parent. This setting is only relevant if the object has a transform parent.

Table 9: Properties in group **Transformation** → Transform connections.

Regarding the transform connection properties described in Table 9, please also read Section 5.2.1 on page 31 for additional information.

## 4.4 Statistics

Objects in PreonLab can gather various statistics during simulation or post-processing. Plots for these statistics can be viewed using the Plot dialog. Additionally, objects may display statistics for the current time in the on-screen-display (OSD). The following properties control whether and how statistics are gathered, stored and displayed.

Property	What it does
track statistics	Enables or disables gathering of statistics during simulation and playback. You should only turn this off if you are sure that you don't need statistics for this object and you experience performance problems caused by statistics gathering.
osd position	Specifies where the statistics for this objects are displayed in the osd.
show sim statistics	Specifies whether statistics (except for performance timings) are displayed in the osd.
show timings	Specifies whether performance timings are displayed in the osd.
show osd in video	Specifies whether the statistics for this objects are also included in OpenGL frames written to disk.
show color legend	Specifies when to display the color legend. By default, the property is set to <b>Dynamically</b> , for which the color legend is only shown when data to display is available. Alternatively, <b>Never</b> will hide the color legend while <b>Always</b> will show the color legend even when there is no data for the object. This is for example useful to consistently display the color legend in an exported video even if data is only measured sporadically.
record statistics	Specifies whether statistics are written to disk whenever a new frame is reached during simulation or playback.
track memory	If enabled, memory consumption for this object is tracked as a statistic.
store statistics per substep	If off, statistics are only gathered when a new frame is reached. This helps to reduce the size of statistics data for objects for which the statistics are not needed in sub-frame precision. This is <b>off</b> by default for rigids and sources and <b>on</b> for all other objects.
live csv export	Specifies whether statistics are written to disk in the CSV format whenever a new frame is reached during simulation or playback.
csv file (read-only)	The location of the csv file in which statistics are stored if live csv export is enabled.

**Table 10:** Common properties in group Statistics and subgroups OSD Settings and Statistics Recording.

## 5 Connections

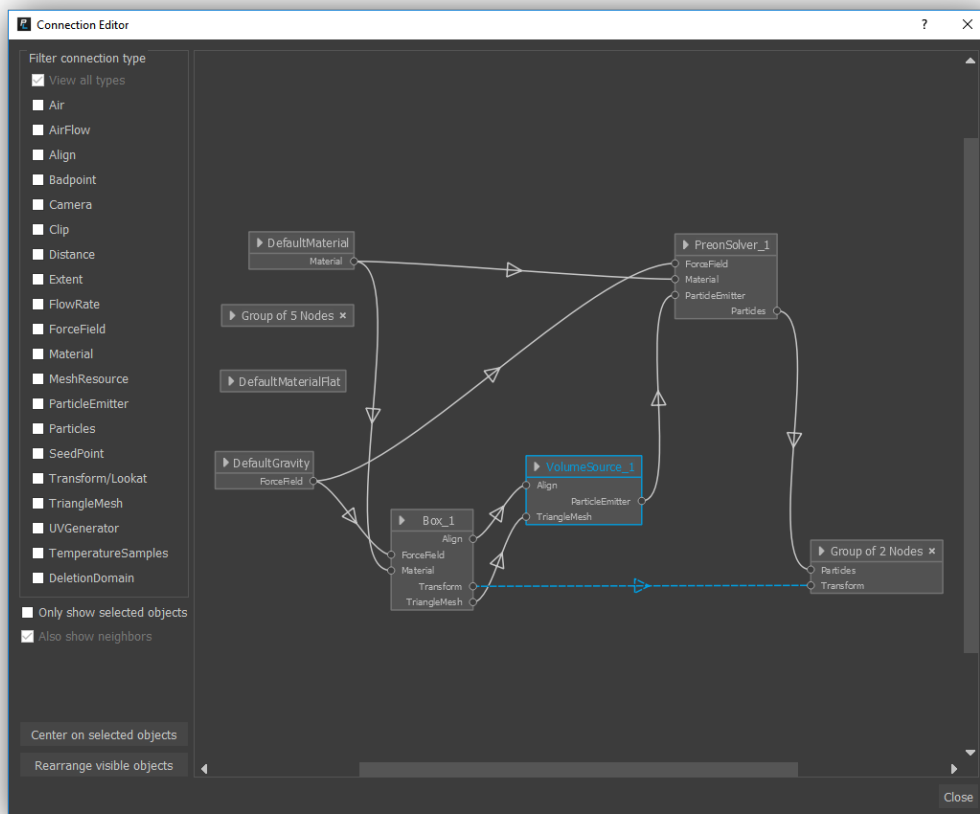


Figure 19: Connection editor.

A connection establishes a relation between two objects. Connections are always directed, they start from one object and end in another object. Figure 19 shows connections for a simple scene containing a solver, a volume source, a box and two sensors. Connections allow to define the interplay between different objects in a scene. PreonLab creates some basic connections automatically when inserting objects. The default gravity is for example connected to the solver via the *ForceField* slot which ensures that the fluid simulated by the solver is subject to gravity. Similar, the volume source is automatically connected to the solver via the *ParticleEmitter* slot, so that the fluid particles generated by the source are simulated by the solver. However, there are cases in which it is necessary to create connections manually, for instance to de-



fine on which geometry the force sensor should measure forces. Another use case for manual connections are *Transform* connections that are discussed in Section 5.2.

## 5.1 Using the connection editor

By default, the connection editor shows all objects in the scene and all their connections, while the selected ones are highlighted. For each object, the connection editor displays all available input and output slots. Input slots are located on the left, while output slots are located on the right. To view all available slots for an object, expand the object by clicking on the white arrow to the left of the object name. New connections can be created by clicking on an output slot and releasing the mouse on an input slot of another object. Existing connections can be deleted by clicking on the connection and pressing the delete key.

For large scenes, the connection graph might get quite large making it hard to create new or delete existing connections. Therefore, there are a number of tools to simplify the currently displayed graph as described in the following subsections.

### 5.1.1 Only showing selected objects

Using the checkbox *Only show selected objects* on the left of the connection editor, you can limit the graph to only show objects that you currently have selected in the scene using the Scene Inspector or the graphics window. With the additional checkbox *Also show neighbors* below, objects that are currently connected to at least one of your selected objects will also be shown. You can also center the current view on your selected objects by clicking on the *Center on selected objects* button.

### 5.1.2 Rearranging objects

To rearrange the boxes symbolizing the scene objects in the connection editor, they can just be dragged using the mouse. The respective positions of each object in the connection editor is saved in the scene. Additionally to manually rearranging the objects in the graph, the connection editor can automatically rearrange the currently visible objects by clicking on the *Rearrange visible objects* button. Note that this only works as long as only a moderate number of objects are visible.

### 5.1.3 Filtering connection types

You can restrict the connection editor to only show slots of a specified type and also show objects that have at least one slot of this type. This can be done by selecting the respective connection types on the left side of the connection editor.

### 5.1.4 Grouping objects

To gain better overview and perform multiple connection tasks on a set of objects, you can use the grouping feature. To use it, select similar objects by left-clicking and drawing a rectangle around them. All contained objects will then be merged into a group. Common connections are displayed as a single edge. You can add and remove connections from groups as you would do with regular objects. This corresponds to adding/removing connections to/from all contained objects separately. Note that only those slots are available, that all contained objects have in common. If connections are not shared by all objects, a dashed edge is displayed. Those can be deleted or a new edge can be drawn over it, to unify the group. Moving a group box translates the position of the contained object boxes.

## 5.2 Transform connections

The *Transform* connection allows you to combine the transformation of one object with the transformation of another object. In general, this can be used to specify the transformation of one object relative to the transformation of another. A simple use case is to move multiple objects synchronously by keyframing a single parent object. If there is an *Transform* connection from an object A to an object B, we say that object B derives its transformation from object A and that object A is the parent of object B. Note that only position and orientation are derived, while scale is not.

### 5.2.1 Relative transformations

Be aware that once an object derives its position and orientation from another object, the meaning of the position and orientation displayed in the property editor changes. These properties are now relative to their parent and do not state the global position and orientation of the object. This also has consequences when creating a *Transform* connection. Let's consider an object A located at position (10, 0, 0) and an object B located at (0, 0, 0). Now a *Transform* connection is created from object A to object B. You may notice that while object B remains at the same place, its position displayed in the property editor changes to (-10, 0, 0). PreonLab does this automatically to keep the object at the same global position. In general, PreonLab always preserves the global position of an object when creating or removing a transform connection and changes the local position if necessary. It does the same for the orientation, however there is an exception: If an object has an **orientation control mode** of *revolution* or *revolutions\_PerSecond*, the local orientation will not be adjusted when creating or deleting a transform connection. This exception was introduced because adjusting the local orientation automatically can lead to a changed rotation axis, which is usually highly undesirable.

If this (default) behavior is not what you intend to achieve when creating a *Transform* connection, there is the possibility to disable it. Go to **Transformation**→**Transform connections** to find the properties described in Table 9 on page 27 that allow you to customize the behavior on a per-object basis.

## 6 Keyframing

Using keyframing, you can specify which and how properties change over time. For instance, keyframing the position of an object can be used to move the object over time. Keyframing works by defining a sequence of keys, whereat each key specifies the value of the keyframed property at a certain point in time. The sequence of keys defines a function that maps from time to values of the property.

For some fluid properties, keyframes can also be used to map temperature to other properties, see Section 9.1.8 for more details.

### 6.1 Keyframe editor

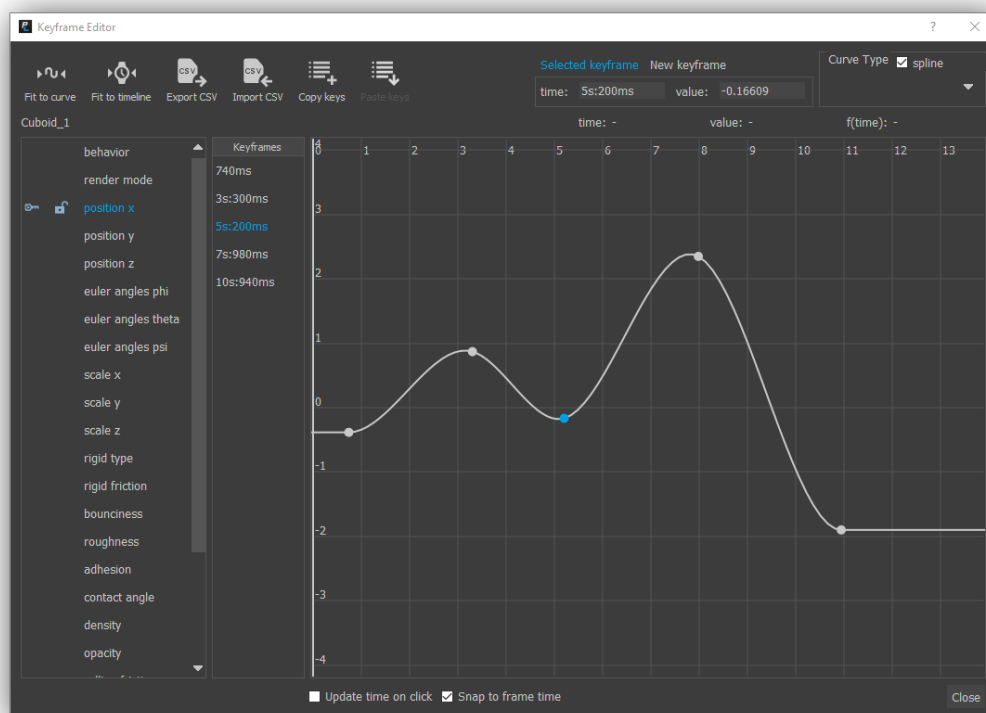


Figure 20: Keyframe editor.

The keyframe editor can be opened by selecting an object and clicking on the *Keyframes* button in the toolbar. The keyframe editor displays all properties of the object that can be keyframed in a list. Selecting a property shows all keyframes for this property and a plot of the corresponding function mapping time to values. The keys are visualized as small dots on the graph. Time is plotted on the x-axis while function values are plotted on the y-axis. Right above the plotting area, information about the current mouse cursor position is shown. *time* displays the current time at the mouse cursor position, *value* displays the value at the mouse position and *f(time)* displays the value of the keyframing function at the current time.

A new key can be created by double clicking in the plotting area. Keys can be moved by dragging them using the mouse and removed by pressing the delete key. You can zoom in and out using the mouse wheel and shift the plotting area by pressing the mouse wheel and moving the mouse. Pressing *Fit to curve* will reset the zoom so that all keys are visible, while *Fit to timeline* ensures that the time interval of the plotting area matches the interval displayed in the timeline.

By default, values between keys are interpolated using spline interpolation, which ensures that the property value changes smoothly over time without sudden changes in the first derivative. If you want more control over the interpolation between keys, you need to uncheck the spline checkbox next to the **Curve Type** label in the top right of the keyframe editor. After disabling spline interpolation, you can select segments between two keys in the keyframe editor and modify the interpolation used between them by choosing a different curve type.

In the top center of the keyframe editor, you will find tabs named *Selected keyframe* and *New keyframe*. If you select the first tab, *time* and *value* of a selected key will be shown. These values can be edited and will be directly applied to the selected key. If you select the second tab, you will be able to define *time* and *value* for a new key. The key will be added after clicking the *Add* button.

At the bottom, you will find a check box labeled with *Update time on click*. By default it is unchecked or disabled, meaning that a click on the graph will not update the current time. If enabled, the current time will be updated and everything related to the new point in time will be loaded, even in the main window.

By using the check box labeled with *Snap to frame time*, the cursor position is snapped to the nearest frame time on the time axis. Thus, keyframes that are added via double click are automatically snapped to the nearest frame time position. This applies to moving an existing keyframe, too. The checkbox is enabled by default.

It is possible to lock a property and, thus, prevent further editing of its keyframes. Whether a property is locked or unlocked is illustrated via an icon left of the property name. Only keyed properties are lockable. By right-clicking on a property, a context menu will give you the following options: a) *Lock* or *Unlock*, if an unlocked or locked property is selected, respectively b) *Lock all keyed properties* and c) *Unlock all keyed properties*. The last two options (lock or unlock all keyed properties) appear regardless of the current selection. For a locked property, its keyframes can still be inspected via selection of their nodes or curve intervals.

### 6.1.1 Copy and paste keyframes

Pressing the *Copy keys* button in the top center of the keyframe editor copies the keyframes of the selected properties to the clipboard. Multiple properties can be selected by pressing and holding the CTRL or SHIFT key and clicking on the respective property names. The copied keyframes can be pasted to another object by selecting this object in the scene inspector and then pressing the *Paste keys* button in the top center of the keyframe editor. When pasting multiple properties, the property names have to match exactly for keyframes to be pasted. When pasting the keyframes of a single property to one selected property, pasting works as long as the data types of the two properties are compatible and will print an info message otherwise.

*Hint:* The copy and paste functionality for keyframes works across scene instances, too.

### 6.1.2 CSV import / export

You can export all keyframes in the CSV file format by clicking on the respective button in the top of the keyframe editor or export keyframes of a single property by right-clicking on the respective property in the keyframe editor. Keyframes can also be imported from a CSV file.

The CSV import on the other hand can be useful to import sequences created in other applications. For a successful import, you need to use semicolons as separators and the column names need to be in accordance with the respective property names of the selected object (as, e.g., displayed in the property editor). Composite properties pose a special case because each component is listed separately in the keyframe editor and, thus, requires its own column in the CSV file. Therefore, an example CSV file with two position keys may look like this:

```
Time;position x;position y;position z
0;0;0;0
2.0;1.5;2.0;3.5
```

Note that imported keyframes cannot be manipulated by default, since the properties they belong to will be locked after the import process. These locks can be manually overwritten, as described above.

## 6.2 Best practices

Setting up a hierarchy of objects using the *Transform* connection can often simplify keyframing greatly. Consider a car that consists of many objects like wheels, engine and so on. The basic movement of the car could be keyframed using a single transform group that is connected to all parts of the car. Note that the objects could still be keyframed individually, for example to achieve spinning wheels.

### 6.2.1 Make the camera follow an object

The simplest way to keyframe a camera so that it follows a moving object is to use derived transformations. Just position the camera so that it looks on the object and connect the *Transform* slot of the object to the *Transform* slot of the camera using the connection editor. The camera will now move synchronously with the object. It will also rotate around the object if it rotates. If you need to avoid this, don't keyframe the position of the object directly. Instead, keyframe the position of a Transform group and connect it to the camera and to the object.

Another possibility to control the rotation of the camera is the *Lookat* connection slot. You can connect the *Transform* slot of any object into the *Lookat* slot of the camera and the camera will always look towards this object.

### 6.2.2 Shortcuts

The keyframe editor is not the only way to create keyframes. Pressing **CTRL + k** will create or overwrite a position, orientation and scale key (referred to as transformation key) for the selected object at the current time and object transformation.

It is also possible to create a new key using the property editor. Right click on a property that can be keyframed and click on *Set key* to create or overwrite a key for this property at the current time. You can also click on *Show curve* to open the property editor and view all keyframes for the property.

## 6.3 Known limitations

When exporting keyframes in the CSV file format, only the time/value combination is saved, but not the curve type. Furthermore, values are exported for all keyframed points in time for every selected property. If no keyframe exists at such a point in time for a particular property, it is interpolated using the specified curve type. If this is not suitable, you can use the PreonPy Python API to import and export keyframes (see Chapter 19 for more details).

## 7 Statistics and plots

Per default, PreonLab tracks and records (saves on disk) statistics for every object in the scene. Respective settings can be changed per object via the properties listed in the **Statistics** group. PreonLab distinguishes between simulation related statistics and timings.

### 7.1 Plots

All statistics can be plotted using the *plot dialog* which is accessible via the toolbar. To plot data for a specific object, first select the object, then click on the *Plots* button. To open multiple plot dialogs for the same object, keep the object selected and click multiple times on the *Plots* button. In the plot dialog, select one or multiple properties which you want to plot.

You can also plot data for multiple objects in one single plot dialog by selecting the objects in the scene inspector and clicking on the *Plots* button.

If you hover over a plot displayed in the plot dialog using the mouse, the time and value for the hovered position of the plot will be printed in the top right corner. Additionally, the plot dialog will display the average of the plotted data for the current time range.

Control	What it does
mouse wheel	Zoom in and out.
right mouse button + mouse move left or right	Zooming in or out of value range only (y-axis).
right mouse button + mouse move up or down	Zooming in or out of time range only (x-axis).
mouse wheel click	Move plot view (panning). Changes value and time range but does not zoom in or out.

**Table 11:** Controls for navigating in the plot dialog.

### 7.1.1 Plot color

The plot color is automatically selected by PreonLab. You can also choose a color manually by clicking on the displayed color in the property list. This will open a color dialog which lets you choose a color.

### 7.1.2 Plot range

The range of the x axis (time) and y axis (value) can be adjusted in the *Plot range* tab. For both axes, you can also reset the range using the respective *Reset* buttons. The checkbox *Auto update* switches automatic updating of plot ranges during the simulation or playback on or off.

### 7.1.3 Sampling

By default, the plot dialog will directly visualize the raw simulation data. Since PreonLab's time step may be adaptive, this means that the intervals between plot samples are usually not fixed. This often leads to noisy plots that are oversampled in certain regions, which can also cause performance problems when processing the data.

In the *Sampling* tab you can filter the raw simulation data to produce more meaningful plots. The checkbox *Fixed step* lets you use a fixed sampling rate for plot sample points. The checkbox *Smoothing* will enable or disable smoothing of sample points with a defined smoothing width. This is useful to eliminate noise and see trend lines more clearly.

### 7.1.4 Export

You can export statistics as a CSV file using the *Export CSV* button found in the *Export* tab. Note that only the statistics of the selected properties are exported. In order to export all statistics regardless of whether they have been selected or not, use the *Export all CSV* button. If you have specified additional post-processing steps in the *Sampling* tab, they are applied on the exported data, too.

The *Save image* button saves the displayed plots as an image.



## 8 Scene and basic objects

### 8.1 Scene

With the *User Preferences*, you can define a set of default values, so you do not have to redefine your desired preferences every time you create a new scene. However, the scene properties initialized with the *User Preferences* can still be modified later. Scene properties are always saved and restored per scene.

Property	What it does
Appearance→background color	Specifies the background color of the scene.
Appearance→background color 2	Specifies a second background color. If this color differs from the first background color, a color gradient between the two will be interpolated across the background hemisphere.
cache directory	PreonLab writes its simulation data to this path and reads existing data from it. The data path is relative to the location where the scene file is stored. By default, this is set so, that the simulation data is read from and written to the same folder as where the scene file is located. However, it can be adapted for example to read and write from a network location.
individual #threads	If enabled, this allows you to specify an individual number of threads to be used for the related scene in <i>#threads for this scene</i> .
#threads for this scene	The number of CPU threads used by PreonLab if <i>individual #threads</i> is enabled. Note that any number greater than the <i>maximum number of threads</i> specified in <i>User Preferences</i> will not affect the actual number of threads used by PreonLab.
specify thread affinity	If this property is enabled, PreonLab tries to maximize the overall performance by manually assigning threads to CPU cores instead of using the OpenMP and operating system scheduler for this.
up axis	Defines whether the z-axis or the y-axis is used as up axis. Per default the z-axis is up. Note that if you prefer to have the y-axis as up axis you probably want to adjust the gravity vector to point in negative y-direction. Other changes are not required.

<b>simulation frame rate</b>	The simulation frame rate decides how often the simulation data is saved to disk. It also governs the maximal time step to use. The time step can never be larger than the frame rate. It is possible to keyframe the simulation frame rate in order to resolve several parts of the simulation with different frame rates. Thereby, only the step interpolation function is available.
<b>view frame rate</b>	The view frame rate specifies the frame rate for post-processing operations like sensor measuring or rendering. By default, the view frame rate and simulation frame rate are set to the same value. However, there are also many applications for separate simulation and view frame rates. For instance, it allows you to process a 50 fps simulation at 25 fps, discarding every second frame and thereby saving performance. Like the simulation frame rate, the view frame rate can be keyframed.
<b>simulation start</b>	Denotes the start time of the simulation, specified in the time format <code>&lt; #months &gt; mo :&lt; #days &gt; d :&lt; #hours &gt; h :&lt; #minutes &gt; m :&lt; #seconds &gt; s :&lt; #milliseconds &gt; ms</code> . For example, write <code>1m:3s:500ms</code> to specify a start time of 1 minute and 3.5 seconds. This value is also represented by the simulation start marker in the timeline.
<b>simulation end</b>	Denotes the end time of the simulation, specified in the time format. This value is also represented by the simulation end marker in the timeline.
<b>playback/postprocess start</b>	Denotes the start time for playback and postprocessing, specified in the time format. This value is also represented by the playback/postprocess start marker in the timeline.
<b>playback/postprocess end</b>	Denotes the end time for playback and postprocessing, specified in the time format. This value is also represented by the playback/postprocess end marker in the timeline.
<b>update sensors at substeps</b>	Enables or disables sensor updates at simulation substeps. If this is turned off, sensors will only be updated at full frames (according to the view frame rate) during the simulation. This may improve performance, but it also might lead to inaccurate post-processing results if the view frame rate is not high enough. Note that for distributed computations, this property will be ignored, and sensors will only update at full frames, confirm Chapter 20.

**Table 12:** Properties for scene object.

The **Scene** gathers simulation-related statistics like total number of fluid particles and total computation times. A set of these statistics is printed in the upper-left OSD of the graphics window by default. All statistics can be accessed via the *Plot* tool. In order to print more fine-grained statistics in the OSD, the properties in group statistics need to be set accordingly.

Property	What it does
<b>show granular timings</b>	If enabled, granular timings are printed in OSD. See Table 14 for more details.

<b>show solid particles</b>	If enabled, the number of total and active solid particles is printed. See Chapter 13 for more details on solid particles.
-----------------------------	--

**Table 13:** Additional statistic properties for **Scene** in **Statistics**→**OSD Settings**.

Timing	What it measures
<b>Total Update</b>	The total accumulated computation time needed for updating the simulation and sensors, saving and loading data, and updating the user interface from start time/frame to current time/frame.
<b>Total Update Sim</b>	The total accumulated computation time needed for updating the simulation from start time/frame to current time/frame. This only accounts for the update times of the fluid and solid object physics, and the computation time for neighbor search (collider).
<b>Total Update Physics</b>	The total accumulated computation time needed for updating the physics from start time/frame to current time/frame. This only accounts for the update times of the fluid and solid object physics, and does not account for the computation time for neighbor search (collider).
<b>Collider</b>	The computation time to update neighbor lists for the current simulation step.
<b>Fluids</b>	The computation time to update the physical quantities of all fluid particles like forces, position and velocity for the current simulation step.
<b>Solids</b>	The computation time to update the physical quantities of solid objects like forces, position and velocity for the current simulation step.
<b>GUI Update</b>	Time required for updating the GUI in each simulation step or frame.
<b>Loading</b>	Time required to load the last set of simulation data from disk.
<b>Postprocess</b>	Time required for updating sources and sensors, as well as saving statistics and updating objects with respect to keyframes.
<b>Saving</b>	Time required to save the last set of simulation data to disk.

**Table 14:** Explanation of timings tracked and recorded by the scene.

## 8.2 Scene UI Settings

Property	What it does
<b>Appearance</b> → <b>show axes</b>	Draws the orientation axes as an overlay in the graphics window.

<b>Appearance</b> → <b>show grid</b>	Renders a grid at the origin of the scene. Each square has a side length of 1 m.
<b>Appearance</b> → <b>show osd</b>	If switched on, statistics and timings are printed as overlay text in the graphics window. For recording you might want to turn this off.
<b>General</b> → <b>gl auto sleep</b>	If enabled, PreonLab will disable the rendering of simulation substeps if you didn't interact with the graphics window for at least 10 seconds. This improves overall simulation performance.
<b>Recording</b> → <b>save frames</b>	For recording animations you can save the content of the graphics window (including all overlay text). The frames are recorded to the subfolder <i>Visualization/OpenGL/[CameraName]</i> of your scene data.

**Table 15:** Properties to set for Scene UI Settings.

### 8.3 Transform groups

A transform group is a non-physical object with a position and an orientation. Other objects can be connected to a transform group via the *Transform* slot using the connection editor. If you connect the outgoing *Transform* slot of the transform group to an incoming *Transform* slot of another object, the transform group acts as a so-called *parent* in a transform hierarchy and the connected *child* objects will inherit the position and orientation of the transform group. Conversely, the transform group could also be the child of another object and then will inherit its position and orientation. The individual position and orientation of a child is interpreted as a local transformation relative to the parent transform group. Note that this feature is not limited to transform groups (any object can act as a transform parent for other objects).

For example, the transform group can be employed to act as a rotation axis. Therefore, position and orient the transform group such that one of its principal axes represents the rotation axis. Add a second transform group, connect it as a child to the first one via the *Transform* slot and set the revolution around the axis as the respective **euler angles** (PHI, THETA or PSI) in the second transform group.

Sometimes it can be difficult to deduce the angles PHI, THETA and PSI that define a particular rotation axis, e.g., between parts of a windshield wiper, but two 3D points are known that lie on this axis, e.g. on the respective geometry. In this case, right-click on the transform group in the scene inspector and select *compute axis from points* to let PreonLab compute the euler angles. Note that the position of the transform group is suggested as the position of the first point in the opened dialog.

As an alternative to the solution described here, the rotation axis can be set directly as a property in many scene objects together with a revolution or revolution speed. However, the axis cannot be visualized in that case. For further details, see Section 4.3 and Table 8.

## 8.4 Point

A point is like a transform group without a rotation. Points are mainly used to specify seed points for the volume source (see 10.2). Also, a point can be useful together with the placement tool and a distance sensor to measure distances in the scene (see 16.3).

## 9 Fluids

### 9.1 Preon solver

The Preon solver is an implicit, point-based solver of the compressible Navier-Stokes equation. This formulation adds an extra term to the incompressible formulation which takes the current compression into account. The current compression in the fluid might stem from the initial setup or numerical errors from the previous simulation step. This compressible formulation enhances volume preservation even for numerically challenging simulations.

The solver update can be coarsely grouped into three steps: (i) velocity prediction, (ii) pressure projection and (iii) advection. For the prediction of the velocity field, all forces except the pressure force are computed, explicitly. These forces comprise viscosity, cohesion, and body forces. For simulating drag effects, e.g., from air to Preon solver, a drag force has to be added manually. Based on these forces, the velocity and density field for the next time step are predicted. In the second step, the pressure solver computes pressure forces which result in a quasi incompressible state. The tolerated compression can be specified by the user. Finally, the advection step updates the positions of the sample points (particles). The neighborhood information (sample point connectivity) is updated by the so called *Collider*. This is automatically performed in an efficient way by PreonLab.

Add one or multiple Preon solvers to your scene via *Add*→*Fluid*→*Preon Solver*. The solver settings are described below.

#### 9.1.1 General settings

Property	Unit/Type	What it does
spacing	m	The spacing controls the resolution of the fluid. The volume of a fluid particle is $\text{spacing}^3$ .
dimension	-	The default is <b>ThreeDimensional</b> which means that PreonLab simulates in 3D. If this property is set to <b>TwoDimensional</b> or <b>OneDimensional</b> , particle movements are restricted to two dimensions or one dimension.
rest density	kg/m <sup>3</sup>	The density of the fluid.

Table 16: Solver properties.

### 9.1.2 Pressure-solver settings

The pressure solver enforces a compression below the user-defined *density error* value. The solver automatically stops either when the compression is below this value or after a user-defined maximum number of iterations.

Property	Unit/Type	What it does
stable initialization	On/Off	If enabled, pressure-based velocity changes are discarded in the first simulation step for each particle that becomes part of the simulation. This avoids artificial collision responses due to imperfect initialization.
gradient correction	On/Off	If enabled, the pressure gradient is computed using a corrected kernel gradient that always ensures first-order consistency.
gc symmetric	On/Off	If on and gradient correction is enabled, the computed forces are symmetrized. This, however, introduces a certain amount of artificial kinetic energy. <i>This property is experimental and only visible if experimental mode is activated.</i>
solver type	-	<i>Please note that this feature is still highly work-in-progress and not intended for production use.</i> Allows to select the pressure solver formulation. <b>Density invariant</b> is the default solver. <b>DI/PS</b> is an experimental variant that first performs a normal density invariant (DI) solve before performing a position shift (PS) on all particles. The position shift step can be used to improve the particle sampling (e.g. in a closed domain to avoid void spaces) without influencing the velocity field of the fluid. When using <b>DI/PS</b> , the solver properties of the position shift solver can be set separately as shown in Table 19. <i>This property is experimental and only visible if experimental mode is activated.</i>

error relaxation	-	<i>Please note that this feature is still highly work-in-progress and not intended for production use.</i> Defines which error relaxation model is taken. <b>None</b> is the default implementation. <b>ImplicitEOS</b> allows to define a bulk modulus which changes the target density while solving based on the pressure. <b>TargetDensity</b> changes the target density while solving based on a reference time step. Using <b>ImplicitEOS</b> or <b>TargetDensity</b> can help stabilize the phase interface for multiphase simulations with high density ratios. There are additional properties available when choosing one of these two models as shown in Table 20. <i>This property is experimental and only visible if experimental mode is activated.</i>
------------------	---	---

Table 17: Solver properties in **Physics**→**Pressure Solver**

Property	Unit/Type	What it does
initial pressure value	-	Factor which determines how the pressure is initialized before solving the pressure - factor is multiplied with the previous pressure of the particle. <i>This property is experimental and only visible if experimental mode is activated.</i>
min. iterations	-	The solver always does at least this number of iterations in each simulation step. The density error gets smaller with more iterations.
max. iterations	-	The maximum number of iterations the solver does in each simulation step.
stopping criterion	-	The stopping criterion for solving the linear system of equations by the pressure solver. If set to <b>DensityErrorAvg</b> , the pressure solver does iteratively solve for the pressure field until the average density error is below the user-defined value (see <b>density error</b> ). If set to <b>DensityError-Combined</b> , the maximum density error of single particles is also taken into account with respect to the properties <b>high density threshold</b> and <b>max high density particles</b> .
density error	%	The tolerated volume compression given in percentage.
high density threshold	%	Particles with a higher density deviation than the given threshold are classified as high density particles.
max high density particles	%	If the stopping criterion is set to <b>DensityError-Combined</b> the pressure solver will iterate until the percentage of high density particles is below this value and the tolerated volume compression is below the specified <b>density error</b> .



adaptive RJ omega	On/Off	If enabled, the relaxed Jacobi omega parameter will be adjusted dynamically to improve convergence. This property is only visible in experimental mode.
-------------------	--------	---

**Table 18:** Solver properties in **Physics→Pressure Solver→Density Invariant Solver**.

Property	Unit/Type	What it does
clamp minimum pressure	On/Off	If enabled, the pressure is clamped to a user-specified minimum as given by <b>minimum pressure</b> . <i>This property is experimental and only visible if experimental mode is activated.</i>
minimum pressure	Pa	Specifies the minimum pressure value a particle can have during solving of the system. Specifying a value larger than 0 can be interpreted as applying a background pressure to the system and thus helps to minimize void spaces in the simulation. <i>This property is experimental and only visible if experimental mode is activated.</i>

**Table 19:** Additional solver properties in **Physics→Pressure Solver→Position Shift Solver** compared to **Physics→Pressure Solver→Density Invariant Solver**. These are experimental properties that are only visible if **solver type** is set to **DI/PS**. *Please note that this feature is still highly work-in-progress and not intended for production use.*

Property	Unit/Type	What it does
adapt residual	On/Off	If enabled, the solver residual is adapted based on the time step. Only relevant if the <b>Target-Density</b> model is chosen. <i>This property is experimental and only visible if experimental mode is activated.</i>
bulk modulus	Pa	Defines the bulk modulus. Only relevant if the <b>ImplicitEOS</b> model is chosen. <i>This property is experimental and only visible if experimental mode is activated.</i>

**Table 20:** Additional solver properties in **Physics→Pressure Solver→Error Relaxation**. These are experimental properties that are only visible if **error relaxation** is set to something other than **None**. *Please note that this feature is still highly work-in-progress and not intended for production use.*

Fluid	Ratio	Fluid	Ratio
Methyl alcohol	2.24	Linseed oil	2.3
Ethyl alcohol	2.05	Water	3.12
Propyl alcohol	1.88	Ethane diol	1.75
<i>n</i> -Butyl alcohol	1.81	Ethylene glycol	4.88
<i>n</i> -Amyl alcohol	1.5	Propylene glycol	1.56
<i>m</i> -Cresol	1.27	Diethylene glycol	2.23
Cyclohexanol	1.15	Triethylene glycol	1.1
Caster oil	1.3	Glycerol	2.07

**Table 21:** Ratio of bulk viscosity to shear viscosity for selected fluids, as given by Hirai and Eyring.<sup>4</sup>

### 9.1.3 Viscosity

PreonLab 3.2 included two different models for viscosity handling. The **Morris** viscosity model was mainly a shear viscosity model,<sup>1</sup> whereas the **Monaghan** viscosity model resembled a bulk viscosity model.<sup>2</sup> Although both models were available, it was not possible to use both of them simultaneously. Starting with PreonLab 3.3, these models are merged into a single **Newtonian** viscosity model. Now, instead of specifying a single **viscosity** coefficient, the user is able to provide **shear viscosity** and **bulk viscosity** coefficients individually, allowing to better match the genuine fluid behavior.

As in versions of PreonLab before 3.3 the specified viscosity coefficient had to compensate for the missing shear or bulk part, specifying the same property value in PreonLab 3.3 and onwards will result in a less viscous fluid. From PreonLab 4.0 onwards, the default value for the dynamic **bulk viscosity** property is therefore set to 3 mPa s which is three times the dynamic **shear viscosity** of water, as measured by He, Wei, Shi, Liu, Li, Chen, and Mo.<sup>3</sup> Table 21 lists the ratio of bulk and shear viscosity for other selected fluids.

In order to preserve calibration and parametrization determined with previous versions, PreonLab will automatically convert old scenes when loading. In these cases, the **legacy viscosity values** property is enabled, and **shear viscosity** will act as the single **viscosity** property of previous versions. For new scenes the property is disabled by default and has to be enabled explicitly if this behavior is desired. Note that the numerical viscosity added by the default solver of PreonLab 4.0 is significantly lower than for *PreonSolver (3.3 settings)*.

<sup>1</sup>J. P. Morris, P. J. Fox, and Y. Zhu, "Modeling low reynolds number incompressible flows using SPH," *Journal of computational physics*, vol. 136, no. 1, pp. 214–226, 1997.

<sup>2</sup>J. J. Monaghan, "Smoothed particle hydrodynamics," *Annual review of astronomy and astrophysics*, vol. 30, no. 1, pp. 543–574, 1992.

<sup>3</sup>X. He, H. Wei, J. Shi, *et al.*, "Experimental measurement of bulk viscosity of water based on stimulated Brillouin scattering," *Optics Communications*, vol. 285, pp. 4120–4124, 20 2012.

<sup>4</sup>N. Hirai and H. Eyring, "Bulk viscosity of liquids," *Journal of applied physics*, vol. 29, pp. 810–816, 5 1958.

Property	Unit/Type	What it does
viscosity model	-	The viscosity model to use. Can be either <b>Newtonian</b> , <b>PowerLaw</b> or <b>None</b> . In below, the models are described in detail.
shear viscosity	Pa s	The dynamic shear viscosity of the fluid.
bulk viscosity	Pa s	The dynamic bulk viscosity of the fluid. For water, this is typically around three times the dynamic shear viscosity.
flow behavior index	-	The flow behavior index for the power-law viscosity model, as described above. It is only available when the <b>PowerLaw</b> model is selected.
artificial viscosity	-	If enabled, an artificial viscosity term as discussed in the work of Monaghan <sup>5</sup> is added to yield an additional stabilization when modeling higher Reynolds numbers. This property is only visible when <b>legacy viscosity values</b> is enabled, and only applied for resolutions coarser than 1 mm.
legacy viscosity values	-	If enabled, the exactly same discretization and parametrization of the viscosity model as in releases of PreonLab before 3.3 is used. When loading old scenes, this is enabled automatically. For newly created scenes, the property is disabled by default.
implicit	-	If enabled, the implicit viscosity solver as described below is used.

Table 22: Properties in Physics→Viscosity.

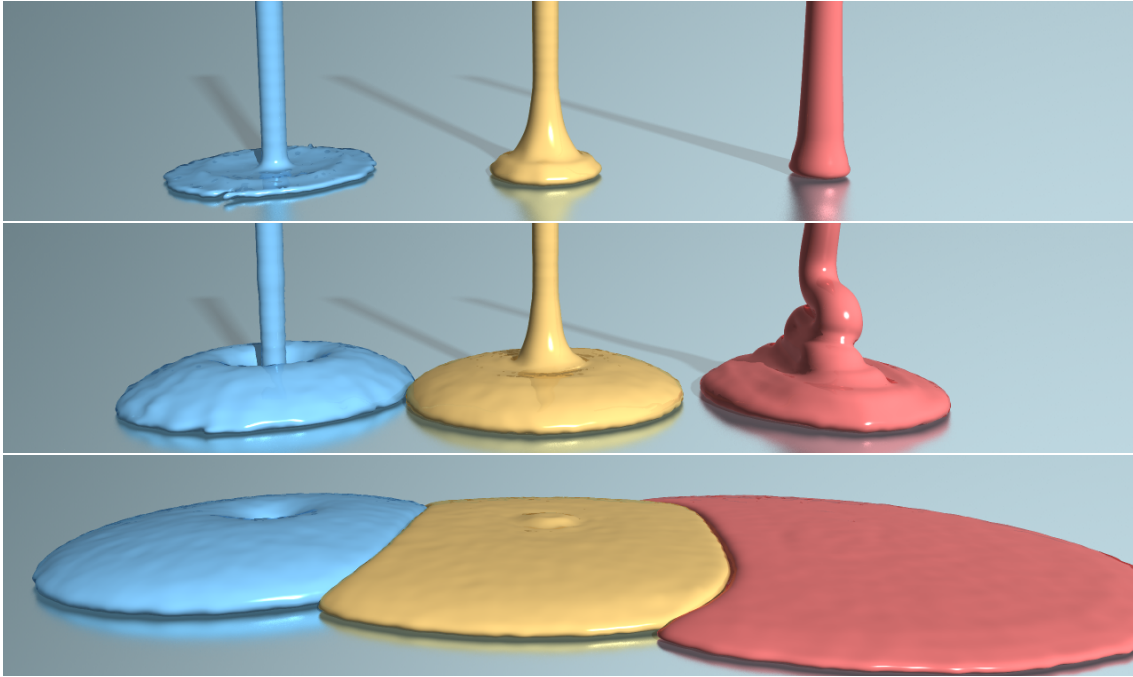
## Models

### Newtonian

Starting with PreonLab 3.3, this is the new default viscosity model. It combines the **Morris** and **Monaghan** models of previous PreonLab releases. Instead of specifying a single viscosity coefficient, it is now possible to set **shear viscosity** and **bulk viscosity** coefficients individually, allowing to better match the genuine fluid behavior. Note that while for divergence-free formulation in closed domains the bulk viscosity is typically zero and can therefore be neglected, it is important to include it when employing a density-invariant formulation as done by the PREON® solver. The Newtonian viscosity model is fully compatible with the implicit solver described below.

### Herschel-Bulkley

<sup>5</sup>J. J. Monaghan, "Smoothed particle hydrodynamics," *Reports on progress in physics*, vol. 68, no. 8, p. 1703, 2005.



**Figure 21:** Demonstration of the power-law viscosity model. The fluid in the middle has a flow behavior index of one and corresponds to a Newtonian fluid. The fluids on the left and on the right have flow behavior indices smaller than and greater than one, respectively. The yield stress is kept zero in all the cases.

The Herschel-Bulkley model is a generalized viscosity model which can be used to model non-Newtonian power-law fluids as well as Bingham plastics and a combination of both. It differs from the Newtonian model in the fact that the viscosity is not constant, but dependent on the strain rate. In this model, the apparent or effective viscosity is calculated as

$$\mu = K\varepsilon^{n-1} + \frac{\tau_y}{\varepsilon} (1 - e^{-m\varepsilon})$$

where  $K$  is the **flow consistency index** that corresponds to the **shear viscosity** and **bulk viscosity** properties of the Newtonian model,  $\varepsilon$  is the strain rate of the flow,  $n$  is the **flow behavior index**,  $\tau_y$  is the **yield stress** and  $m$  is the **stress growth exponent** factor. The factor  $m$ , which is similar to the one used in a common Bingham-Papanastasiou model as described in the work of Papanastasiou<sup>6</sup>, is a regularization parameter used to reduce the computational complexity. The higher the value of  $m$ , the better the model matches the theoretical Herschel-Bulkley model but at the cost of an increased computational complexity. Even though the best value of  $m$  is highly case dependent, a rough initial choice can be made based on the dimensionless number  $M$  of the form

$$M = m \frac{V}{L}$$

where  $L$  is a reference length and  $V$  is a reference velocity of the simulation set up. Deciding a value for  $m$ , such that  $M$  is greater than 50 could be a good choice to start with. A good practice still would be to start with a low  $m$  and increase it until there is a good trade off between a converged solution and the computational performance. Maintaining a  $\tau_y$  greater than zero, the behavior of viscoplastic or Bingham plastics

<sup>6</sup>T. C. Papanastasiou, "Flows of materials with yield," *Journal of Rheology*, vol. 31, no. 5, pp. 385–404, 1987. DOI: 10.1122/1.549926. [Online]. Available: <https://doi.org/10.1122/1.549926>.

can be simulated. Those are the fluids that behave like a plastic until a minimum shear stress  $\tau_y$  is applied to initiate the flow. Flows with a flow behaviour index greater than one represent shear thickening fluids, whereas those with flow behaviour index less than one represent shear thinning fluids. Keeping  $\tau_y = 0$ , the model reduces to a power-law model, whereas keeping  $n = 1$ , the model reduces to a Bingham model. When keeping both  $n = 1$  and  $\tau_y = 0$ , the model matches the **Newtonian** model.

It is also possible to use the Herschel-Bulkley model in a semi-implicit fashion by enabling the **implicit** property. When doing so, the apparent viscosity  $\mu$  is precomputed in each simulation step and kept constant for each particle. The resulting linear system is solved with the implicit viscosity solver. This is not a fully implicit formulation, as the apparent viscosity is assumed not to change while solving the system. However, the approximation typically is satisfactory for the simulation outcome at large.

## Implicit formulation



**Figure 22:** Highly viscous fluids can be simulated efficiently with the implicit viscosity solver.

PreonLab includes an implicit formulation for the viscosity force. The implicit formulation drastically improves the performance compared to the default explicit formulations when simulating highly viscous fluids, as it is stable for large time steps. For low-viscosity fluids such as water, however, the overhead of the implicit formulation, which requires a linear system to solve, is typically not needed. Therefore, the implicit formulation is disabled by default.

Property	Unit/Type	What it does
<b>implicit</b>	-	If enabled, the implicit formulation is used instead of the explicit one.
<b>min. iterations</b>	-	The solver always does at least this number of iterations in each simulation step.
<b>max. iterations</b>	-	The maximum number of iterations the solver does in each simulation step.

stopping criterion	-	The stopping criterion for solving the linear system of equations by the implicit viscosity solver. If set to <b>AvgResidual</b> , the solver does iteratively solve for the velocity field until the average residual of all particles is below the user-defined value (see <b>tolerance</b> ). If set to <b>MaxResidual</b> , the maximum residual of single particles is taken into account, i.e., it is ensured that the residual of each particle is below <b>tolerance</b> .
tolerance	-	The tolerated residual. A higher tolerance reduces the number of iterations and shortens the computation time.

Table 23: Solver properties in **Physics**→**Viscosity**→**Viscosity Solver**.

### Known limitations

For multiphase simulations where the involved fluids use different viscosity models or have different densities, momentum preservation due to the viscosity force is not guaranteed. This will be addressed in a future release.

#### 9.1.4 Surface tension

Property	Unit/Type	What it does
cohesion model	-	There are three models implemented to compute cohesive forces: (i) <b>PotentialForce</b> and the two legacy models (ii) <b>PairwiseForce</b> and <b>Preon-Cohesion</b> . The models are explained in more detail in below.
performance mode	-	Only available for the model <b>PotentialForce</b> . It allows to choose between <b>Speed</b> and <b>Quality</b> modes.
cohesion	N/m	Controls the cohesion (surface tension) of the fluid. Higher values result in larger forces. Zero means no cohesion.
adhesion	N/m	Controls the adhesion coefficient at the interface between this fluid and another fluid. For the interaction between particles of different phases, the average values of their respective <b>adhesion</b> value are used to compute the interaction force. This leads to symmetric forces as long as both fluids use the same <b>cohesion model</b> .

Table 24: Properties in **Physics**→**Cohesion**





**Figure 23:** Solid-fluid interaction. Left: A tomato is washed under a water faucet. In the beginning the outflow is low. Cohesion effects are clearly visible at the concave water jet and the droplets beneath the tomato. Due to the established adhesion model (see Section 9.1.4 and Chapter 13), the water flows around the tomato before dropping off. Right: With increasing flow rate, this effect is replaced by the water splashing off of the tomato on contact.

In PreonLab cohesive forces are computed as inter-particle forces between a fluid particle and its neighboring particles. This model is also known as pairwise force model (PF) and has been originally proposed by Tartakovsky and Meakin.<sup>7</sup> The model computes the phenomenon of surface tension by computing pairwise attraction forces between particles based on their distance. Thereby, the force acts in the direction of the density gradient which creates surface tension. For a given fluid, the magnitude of this force depends on the distance between particles and the cohesion parameter.

The total particle-particle interaction force acting on the fluid particles is non-zero only near fluid surfaces. The advantage of the pairwise-force model over the continuous surface force model is that it perfectly conserves momentum, and that it is insensitive to the quality of the computed surface normals. Furthermore, the parameter does not change with the dynamics of the fluid (opposed to the contact angle).

**PairwiseForce:** The pairwise-force model implemented in PreonLab has been calibrated and validated for water by the TU Dresden.<sup>8</sup> However, there is no direct mapping to the physical surface tension value in N/m. The default parameter of 1 should match water. For new scenes, we recommend to use the **PotentialForce** model.

**PotentialForce:** The potential force model is an advanced version of the pairwise-force model which can reproduce micro-fluidic behavior of a variant of fluids such as the oscillation frequency of a single droplet. The model offers two different modes, **Speed** and **Quality**, which allow to trade off speed for quality.

The third model, **PREON® cohesion**, is only kept for legacy reasons.

### 9.1.5 Solid-fluid interaction

The interaction of fluid with solid objects is handled in a unified particle-based manner. Therefore, solid objects are automatically sampled with particles which are transformed with the object by PreonLab. The pressure value of a fluid particle is mirrored onto solid object particles in close proximity while the velocity is set to the corre-

<sup>7</sup>A. Tartakovsky and P. Meakin, "Modeling of surface tension and contact angles with smoothed particle hydrodynamics," *Physical Review E*, vol. 72, no. 2, 026301:1–9, 2005.

<sup>8</sup>[Sprinkling simulation with PreonLab by TU Dresden.](#)

sponding solid velocity at that position, thereby realizing a no-slip boundary condition at the solid interface as proposed by, e.g., Hu and Adams.<sup>9</sup> Explicit forces, like adhesion and viscosity are computed similar to fluid-fluid particle interaction using the respective model set for the fluid solver.

Property	Unit/Type	What it does
<b>no gap</b>	On/Off	If switched off, fluid particles are kept in a distance to solid walls (geometries) of the length of particle spacing. This results in a visible gap in the size of half the particle spacing and a thickening of geometries reducing the inner void volume. By switching, the <b>no gap</b> property to on, the gap is eliminated. As the fluid particles distance to the geometry is reduced, the CFL time step is also smaller. Furthermore, more boundary samples are generated when the <b>no gap</b> option is on which might increase the computation time and memory consumption.
<b>Friction→distance correction</b>	On/Off	If switched on, a correction for the boundary sampling is employed that makes the compute friction force resolution-independent. However, the resolution independence is guaranteed only for solids with <b>roughness</b> set to 1.
<b>Adhesion→use cohesion as adhesion</b>	On/Off	If enabled, the effective adhesion of a solid in contact with this fluid is computed as adhesion of solid times cohesion. If disabled, the effective adhesion is computed taking the adhesion property of this fluid times the adhesion of the solid.
<b>Adhesion→solid adhesion</b>	-	Adhesion value used in contact with solids. This property is only shown if <b>Adhesion→use cohesion as adhesion</b> is enabled.

**Table 25:** Solver properties in group **Physics→Boundary handling** and further subgroups.

### 9.1.6 Timestep computation

The following parameters control how the timestep is determined during the simulation. Please note that usually there is no need to change any timestep related parameters. The solver will automatically choose a timestep that balances simulation accuracy, stability and performance.

<sup>9</sup>X. Y. Hu and N. A. Adams, "A multi-phase sph method for macroscopic and mesoscopic flows," *Journal of Computational Physics*, vol. 213, no. 2, pp. 844–861, 2006.



Property	Unit/Type	What it does
timestep	s	The timestep of the solver. If adaptive time stepping is enabled, this is automatically computed by the solver and the property will not be visible.
maximal timestep	s	If adaptive time stepping is enabled, this value determines the maximal timestep.
CFL number	-	The CFL number used for adaptive timestep computation based on fluid and solid velocities. Larger values allow for larger timesteps. Results will not be accurate for values greater 1.
adaptive	On/Off	Specifies whether adaptive time stepping is used.

Table 26: Properties in group Time Stepping.

### 9.1.7 Deletion criteria

In very rare instances, the solver may choose to delete individual particles in order to keep the overall simulation stable. The settings in the property group **Deletion criteria** specify the thresholds above which particles may be deleted. We highly recommend to not change these settings. If you experience a significant loss of volume, it is usually a better way to decrease the time step and check the overall simulation setup. For instance, a CFL number of 0.8 often results in less deleted particles compared to a CFL number of 1.

Property	What it does
on CFL violation	This property specifies what happens with particles that violate the CFL condition due to numerically challenging settings. <b>Ignore</b> does not adjust the particles. <b>DeleteParticles</b> deletes all particles violating the CFL condition and <b>AdaptTimestep</b> individually adapts the time step for each particle so they meet the CFL condition.
stuck prevention	Enables or disables the stuck prevention which may delete particles based on their density.
predefined stuck values	Enables or disables user-specified stuck thresholds.
density threshold	Specifies a density threshold above which particles are deleted. The density is given as a normalized value (1 refers to the rest density).
stuck threshold	Specifies a density threshold above which particles stuck between solid objects are deleted. For this density, only solid objects are considered, while other fluid particles are ignored. The density is given as a normalized value (1 refers to the rest density).

high pressure action	In very rare cases, compressed particles might get such a high pressure value that the value can not be represented accurately anymore for numerical reasons. This property specifies whether such a particle should be deleted ( <b>Delete</b> ), its pressure value should be clamped to a predefined maximum value ( <b>Clamp</b> ) or the pressure value is kept without adaption ( <b>None</b> ).
----------------------	--

Table 27: Properties in group **Deletion criteria**.

### 9.1.8 Thermodynamics

The Preon solver can simulate heat diffusion and thermal interaction with other fluids and solids. By default, thermodynamics computation is not performed. You have to activate it for each fluid object individually by setting the **system type** to a value different from **None**, see Table 28 for more details. For fluid-solid interaction, see Section 13.1 respectively.

**Current constraints:** The properties **heat capacity** and **thermal conductivity** are currently assumed to be constant and, thus, are no functions of temperature.

Property	Unit/Type	What it does
system type	-	Defines whether and how the fluid particles thermally interact with each other and with particles from other fluids or solids. If set to <b>None</b> , they are not considered in the equation at all. <b>Isolated</b> allows diffusion among the particles of this fluid, whereas <b>Closed</b> extends diffusion to particles of other fluids or solids.
temperature	°C	If changed, sets the temperature for all existing fluid particles. <b>Note: The change overwrites all simulation results up to this point. We recommend using the temperature property in the fluid sources (see Chapter 10) to set the initial temperature as soon as the particles are generated. Further, you can use the temperature of solid objects (see Chapter 13) as additional heat source or sink with respect to the fluid.</b>
heat capacity	J/(°C kg)	The specific heat capacity of the substance on a per mass basis, i.e., the isobaric mass heat capacity. The default is 4181.3 (liquid water at 25°C).
thermal conductivity	W/(m °C)	The thermal conductivity $\kappa$ is the property of a material to conduct heat. The default is 0.6 (liquid water at 25°C).

temp. based viscosity	On/Off	If enabled, viscosity is computed per particle based on the temperature of the respective particle. The mapping between temperature and viscosity can be keyframed using the Keyframe editor, see <i>Keyframing temperature-based viscosity</i> below for more details.
temp. based density	On/Off	If enabled, the rest density of each particle is based on the temperature of the respective particle.
implicit	On/Off	If enabled, heat conduction is computed using an implicit formulation which is computational more demanding but can handle larger time steps. Recommended if high temperature gradients are the limiting factor for the adaptively computed time step. The properties for the implicit solver are listed in Table 29.

**Table 28:** Properties of Preon solver in group **Physics** → **Thermodynamics**.

Property	Unit/Type	What it does
average conductivity	On/Off	If enabled, the arithmetic mean from the conductivity values $\kappa$ of the participating media (solid and fluid) are taken as the effective conductivity at the interface. If disabled, the effective conductivity is computed as $(4 \cdot \kappa_{solid} \cdot \kappa_{fluid}) / (\kappa_{solid} + \kappa_{fluid})$ .
distance correction thermo	On/Off	If enabled, the thermal diffusion at the interface is enforced to be resolution independent. Note that this can only be guaranteed if $\kappa_{solid} = \kappa_{fluid}$ .

**Table 29:** Properties for fine-tuning the interface handling of fluid and solid for thermodynamics in group **Physics**→**Thermodynamics**→**Interface Handling**.

Property	Unit/Type	What it does
min. iterations	-	The solver always does at least this number of iterations in each simulation step.
max. iterations	-	The maximum number of iterations the solver does in each simulation step.
stopping criterion	-	The stopping criterion for solving the linear system of equations by the implicit thermo solver. If set to <b>AvgResidual</b> , the solver does iteratively solve for the temperature field until the average residual of all particles is below the user-defined value (see <b>tolerance</b> ). If set to <b>MaxResidual</b> , the maximum residual of single particles is taken into account, i.e., it is ensured that the residual of each particle is below <b>tolerance</b> .

<b>tolerance</b>	-	The tolerated residual. A higher tolerance reduces the number of iterations and shortens the computation time.
------------------	---	--

**Table 30:** Solver properties in **Physics**→**Thermodynamics**→**Thermo Solver**.

The temperature of each particle can be visualized during simulation and playback by setting the **Appearance**→**Coloring**→**coloring** to **TemperatureBased**.

### Keyframing temperature-based viscosity

By default, you can keyframe the viscosity of the whole fluid over time using keyframing, see Chapter 6. If the property **Physics**→**Thermodynamics**→**temp. based viscosity** is enabled, the viscosity of each particle is instead determined based on its temperature. This mapping can be changed in the Keyframe editor. If the property is enabled, the x-axis in the Keyframe editor no longer displays time, but instead temperature. The keys and their curves therefore define the mapping between the temperature of a single particle and its viscosity. You can also import the temperature-to-viscosity mapping by using the *Import CSV* button in the Keyframe editor. In this case, the first column in the CSV file must have the following header: *temperature;viscosity*.

### 9.1.9 Evaporation

The Preon solver can simulate the evaporation of fluid into the air. The relevant properties are in group **Physics**→**Evaporation Solver**. By default, the evaporation solver is inactive but can be activated by setting **evaporate** of the solver to **On**, see Table 31 for more details.

Table 31 lists all properties exclusively related to the evaporation solver. The fluid temperature has to be provided in **Physics** of the Preon solver and all properties related to the air by inserting at least one **Air object** (see Section 11.3).

For a better understanding of the prefactor properties, consider that the evaporated water per hour  $g_h$  can be expressed as:

$$g_h = \theta \cdot A \cdot (\tilde{q}_s - \tilde{q}), \quad (1)$$

where  $A$  is the area on the free surface,  $\tilde{q}_s$  the specific humidity ratio in saturated air (at the temperature of the water surface) and  $\tilde{q}$  the humidity ratio in the air. Furthermore:

$$\theta = \text{prefactor} + \text{wind\_speed\_prefactor} * v_{wind} = 11 + 19 * v_{wind} \quad (2)$$

is the evaporation coefficient  $\theta$ , which depends on the wind speed  $v_{wind}$  at the water surface.

Property	What it does
evaporate	Enables / disables the evaporation solver. When you enable it, you should also change the <b>simulation frame rate</b> and <b>view frame rate</b> and make sure that <b>Thermodynamics</b> → <b>system type</b> is set to <b>None</b> .
prefactor	Sets the prefactor of the evaporation coefficient $\theta$ . By default, the prefactor is set to 11.
wind speed prefactor	Sets the wind speed prefactor of the evaporation coefficient $\theta$ . By default, it is set to 19.
max. evap. thickness per timestep	Defines the thickness of the fluid layer at the air interface that can be evaporated within the current simulation step w.r.t. particle mass. If less or equal to 1, at most one layer of particles could be evaporated at once at the free surface. Note that a value greater than 1 is not recommended if you want to employ the <b>equilibration phase</b> property (see further below).
anisothermal	If enabled, the results of thermodynamic computations preceding the evaporation phase are adopted as the starting point of the evaporation process of the fluid. If disabled, the evaporation solver employs the temperature set in <b>Physics</b> → <b>temperature</b> of the fluid solver and, thus, assumes an isothermal fluid.
EVP computation method	Sets the method with what the equilibrium vapor pressure is determined. You can choose between the Magnus formula (only valid for temperatures between -45 °C and +60 °C) and a data table from WebBook (valid for temperatures between 0 °C and +374 °C).
equilibration phase	If set to <b>DeletedParticles</b> , the fluid dynamics of remaining particles are computed after a certain amount of particles have evaporated. If set to <b>FrameWise</b> , the fluid dynamics are computed after each simulation frame. Thus, an equilibrium state can be re-established that might have been disrupted during the evaporation phase (where the fluid dynamics computation is switched off due to the large timesteps).
equilibration duration	Sets the duration of the equilibration phase in simulation seconds. When reached the evaporation phase continues. Note: This property is only visible if <b>equilibration phase</b> is enabled (i.e. not <b>None</b> ).

Table 31: Properties of Preon solver in group **Physics** → **Evaporation Solver**.

## Best practices

For the evaporation coefficient, the *Verein Deutscher Ingenieure* (VDI) recommends in his guideline <sup>10</sup> to consider a value of 5 for covered pools, 15 for a bathtub, 20 for indoor baths and 28 for outdoor baths.

We recommend to simulate the fluid dynamics until an equilibrium is reached. Then, the **simulation frame rate** and **view frame rate** should be lowered from 50 (default) to about 0.00027, which equals one frame per hour. Now you can switch to the evaporation solver (see Table 31).

Note that it is required to employ keyframing to perform the frame rate changes, because without keyframing the frame rates are changed globally (e.g., 50 frames that formerly represented one simulation second would be interpreted as 50 simulation hours after setting the frame rate to 0.00027). Because of this crucial difference, PreonLab will show a message to hint at the keyframing possibilities if a supposedly global change of the frame rates is detected.

Furthermore, the thermodynamics solver and the evaporation solver cannot be employed at the same time due to the huge differences in the timesteps that are chosen to evolve the simulation. Thus, we require to enable only one of these two solvers at a time, i.e. switch off one solver when enabling the other via keyframing.

Finally, an **anisothermal** starting point of the evaporation process is only possible for fluid particles. The film wetting feature described in Section 13.2 currently only allows an isothermal starting point.

The solver might not always be able to evaporate all of the remaining fluid particles or all of the wetting film of a solid object. One reason could be that the fluid particles are not located within an air object and, thus, can not be evaporated by design. Another reason might be that particles are stuck in very narrow spaces where it is difficult for the solver to detect them having an interface to the air phase. The wetting film can not be evaporated if it is covered by particles that cannot be evaporated themselves. In all these cases, PreonLab prints a respective warning which lists the name of the solver or solid object and the point in time at which the state is detected for the particular object.

### 9.1.10 Local refinement

In many applications, it can be beneficial to represent the fluid adaptively with different particle sizes. PreonLab supports simulations with two particle sizes with a ratio of 1:2.

To setup an adaptive simulation, insert two Preon Solvers and connect the *FluidRefinement* connection slot of one solver to the *FluidRefinement* slot of the other. The spacing property of the two solvers will be adjusted automatically, realizing a 2:1 ratio.

---

<sup>10</sup>VDI- Richtlinienausschuss 2089

[https://www.vdi.de/nc/richtlinie/vdi\\_2089\\_blat\\_2-technische\\_gebaeudeausruestung\\_von\\_schwimmbaedern\\_effizienter\\_einsatz\\_von\\_energie\\_und\\_wa/](https://www.vdi.de/nc/richtlinie/vdi_2089_blat_2-technische_gebaeudeausruestung_von_schwimmbaedern_effizienter_einsatz_von_energie_und_wa/)

The next step is to setup one or multiple domains that specify in which region (or regions) of the simulation a refinement is necessary. To do so, insert a Box or Cylindrical domain and set its condition type to **refine**. Then connect the *RefinementDomain* output slot of the domain to the Preon Solver with the greater spacing (this is the solver with an outgoing *FluidRefinement* connection). Once coarse fluid particles of the first Preon Solver enter the domain (or leave it depending on the **region** of the domain), they will be subdivided into finer particles that are transferred into the other Preon Solver.

## Automatic property synchronization

Once a *FluidRefinement* connection exists between two solvers, you will only be able to change properties like cohesion and viscosity in the coarser solver. These properties will also be automatically applied to the finer solver.

## Refinement buffer zone

Particles are always refined inside specified refinement domains, but the solver may also choose to refine additional particles outside that have a high likelihood to enter the refinement region in the near future. This can improve overall performance because it reduces the number of potentially costly refinement events. To achieve this, the solver will refine particles in a buffer zone around refinement regions once a refinement is necessary. This behavior can be adjusted using the collider options **refinement proximity factor** and **dynamic refinement proximity** (found in Settings → Collider). The latter enables or disables the dynamic adjustment of the buffer zone and the former sets the size of the buffer zone as a multiple of the fluid spacing. However, changing these properties is not recommended.

## Limitations

- Cohesion forces between coarse and refined particles can create visible seams at the fluid surface.
- Pathlines do not work correctly yet when tracking a particle that gets refined or merged.
- Height fields can only measure either the refined or the coarse phase, but not both.
- If you want to render adaptive fluids with Preon Renderer, make sure to set the **sdf method** property (located in the Fluid Rendering group of a **Preon renderer** object) to **Density\_Adaptive**. However, seams between the different phases may still be visible in the rendering.
- **Preon Mesher** can only mesh the refined or the coarse phase, but not both.

- Local refinement may not be employed in a multiphase simulation.

### 9.1.11 Rendering of particles

The particles represent partial volumes of the fluid which are rendered as volumetric spheres. Most field values carried by the particles, e.g., velocity (default) or pressure, can be mapped as a color onto the particle. Rendering-related properties of the solver are listed in group **Appearance**.

Property	What it does
<b>render mode</b>	Particle rendering can be set to <b>smoothShaded</b> or <b>invisible</b> .
<b>color</b>	The base color of particles if the coloring is not set to any field variable <b>None</b> .
<b>opacity</b>	Controls the opacity of the particles. If 1, the particles are completely opaque. If 0, they are completely transparent.
<b>render scripted particles</b>	Enables or disables the visualization of scripted particles. This is only relevant in combination with simulation boundary domains that script particles.
<b>Coloring</b>	Lists a variety of properties which control the field variable used for coloring the particles, the value range, and the respective colors for minimum, medium, and maximum value.

Table 32: Properties in group **Appearance**.

### 9.1.12 Serialization

The properties in the group **Serialization** control how fluid data is written to disk in each frame. You can save disk space by enabling compression. You can also save disk space by disabling the storage of certain fluid attributes if you don't need them for your intended post-processing.

Property	What it does
<b>use compression</b>	Enables or disables particle compression. Compression saves disk space, but can be computationally demanding.
<b>max compression error</b>	The maximum allowed compression error as a factor of the spacing. Higher compression errors allow more efficient compression. This is only relevant when compression is enabled.
<b>serialize ID</b>	Enables or disables the serialization of particle identifiers. Particle identifiers are required for all applications that require the tracking of particles over time such as pathlines.
<b>serialize Velocity</b>	Enables or disables the serialization of particle velocities. Particle velocities are required in post-processing for most sensors.



<b>serialize Pressure</b>	Enables or disables the serialization of particle pressures. Particle pressures are required in post-processing if you want to plot pressure on a sensor plane or on meshes using the force sensor.
---------------------------	---

**Table 33:** Properties in group **Serialization**.

### 9.1.13 CSV export

You can export the particle data at the current point in time using the right-click action *Export state as csv*.

## 9.2 Snow solver

The snow model implemented in PreonLab is built upon the model proposed by Stomakhin, Schroeder, Chai, Teran, and Selle.<sup>11</sup> For small deformations the snow stress response will be purely elastic. If it exceeds the range set by **critical stretch** and **critical compression** the response will be modified plastically by hardening. In short, it features a principal-stretch-based yield for plasticity.

This model lets you for example simulate car snowing scenes as shown in Figure 24. Note that, a Preon solver and a snow solver cannot be present in a scene at the same time. In general, sources, sensors and force fields can be used the same way as with a Preon solver.

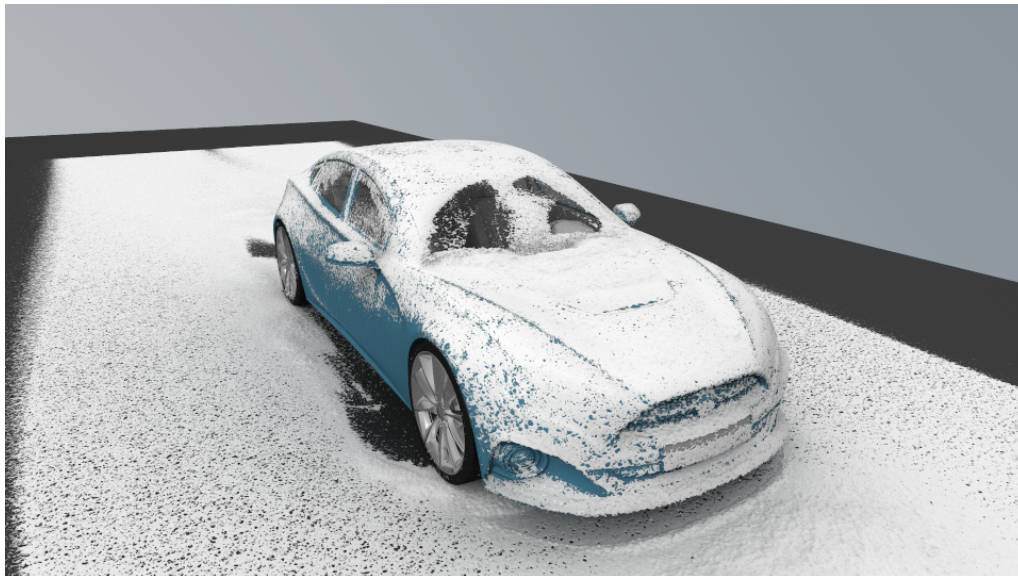
Please refer to Section 9.2.1 to learn what needs to be considered when simulating snow in PreonLab. The properties which define the behavior of the snow are listed and explained in Section 9.2.2.

### 9.2.1 Best practices

The following points need to be considered when using the snow solver in PreonLab:

- A force sensor measuring the force of snow might show over- or underestimated results if the snow solver does not solve the system perfectly. If the force sensor results are important to you, you might change the **stopping criterion** to **MaxResidual**.
- The viscosity and cohesion inside the snow phase are handled by the properties described in Table 34. However, the friction and adhesion to the boundary are

<sup>11</sup>A. Stomakhin, C. Schroeder, L. Chai, *et al.*, "A material point method for snow simulation," *ACM Trans. Graph.*, vol. 32, no. 4, 102:1–102:10, Jul. 2013, ISSN: 0730-0301. DOI: 10.1145/2461912.2461948. [Online]. Available: <http://doi.acm.org/10.1145/2461912.2461948>.



**Figure 24:** Car snowing simulation with moving wipers at a particle spacing of 5 mm visualized with Preon renderer using global illumination and the material preset for snow.

handled the same way as by the Preon solver. Accordingly, the values in the property group **Physics**→**Boundary Handling**→**Friction** are important.

- If the snow exhibits poor resistance to external forces (for the default configuration) increase the elastic iterations or reduce the time step. An example would be the default snow settings can't resist gravity. As a heuristic move the elastic iterations up an order (**min. iterations**) and half the **timestep**.

## 9.2.2 Properties

Table 34 shows the properties of the snow solver. Additional properties are shared with Preon solver. The shared properties are for example the spacing of the snow as well as friction and adhesion at the boundary.

Property	Unit/Type	What it does
rest density	kg/m <sup>3</sup>	The initial density of the snow. Together with the <b>Poisson's ratio</b> and <b>rest density</b> , this determines the purely elastic stress response. A lower <b>rest density</b> makes the snow stiffer, thus requiring a higher <b>Young Modulus</b> to counter it. During the simulation, the density of the respective snow particles may change drastically since snow is compressible in contrast to fluid. The default initial density of snow is set to 400 kg/m <sup>3</sup> .

Young modulus	Pa	Together with the <b>Poisson's ratio</b> and <b>rest density</b> , this determines the purely elastic stress response. Higher values lead to snow that is stiffer / more icy while the opposite is true for watery snow. An order of magnitude change of this value should cover all realistic snow behavior.
Poisson's ratio	-	Together with the <b>Young modulus</b> and <b>rest density</b> , this determines the snow properties of a purely elastic stress response. Higher values lead to a higher shearing stress response and enable splashing and bow-wave behavior.
critical stretch	-	The critical stretch determines when the snow starts to deform plastically when stretching. Essentially, this determines when the snow starts breaking. The default value is $7.5 \cdot 10^{-3}$ . Larger values result in more chunky snow while smaller values result in more powdery snow. Changing this value one order of magnitude (at most) in either direction should cover all realistic snow behavior.
critical compression	-	The critical compression determines when the snow starts to deform plastically when compressing. The default value is $2.5 \cdot 10^{-2}$ . Inversely, larger values result in more powdery snow while smaller values result in more chunky snow. Changing this value one order of magnitude (at most) in either direction should cover all realistic snow behavior.
hardening coefficient	-	The hardening coefficient defines how the elastic response parameters change depending if the deformation becomes plastic. The default value is 10 and a range of 5 in either direction results in plausible behavior. Higher values will lead to more breaks and fractures making the snow more brittle.

**Table 34:** Properties of the snow solver. Except **rest density**, these can be found in the property group **Physics**→**Snow Solver**.

Property	Unit/Type	What it does
solver	-	Allows to choose the solver which is used for solving the system of equations. Since the system is not symmetric, the <b>Bicgstab</b> solver is recommended to ensure convergence. <i>This property is experimental and only visible if experimental mode is activated.</i>
min. iterations	-	The elastic solver always does at least this number of iterations.
max. iterations	-	The elastic solver always does at most this number of iterations.

<b>stopping criterion</b>	-	Defines if the average or the maximum error is checked against the tolerance.
<b>tolerance</b>	-	Defines the error stopping criterion used by the elastic solver.

**Table 35:** Properties of the linear elastic solver of the snow solver in the group **Elastic Solver**. This group is only shown if **Physics**→**Snow Solver**→**Numerics**→**solve type** is set to either **Semimplicit** or **Implicit**.

Property	Unit/Type	What it does
<b>corr. for vel. grad.</b>	On/Off	Decides if the velocity gradient is computed using the corrected kernel gradient.
<b>corr. for Cauchy accel.</b>	On/Off	Decides if the Cauchy acceleration is computed using the corrected kernel gradient.
<b>recalculate plastic det.</b>	On/Off	If enabled, the plastic deformation is computed based on the current configuration instead of being accumulated over time.
<b>only use vol. grad.</b>	On/Off	If enabled, the boundary particles only contribute to the volume change of a particle but not to the shear deformation.
<b>clamp pressure</b>	On/Off	If enabled, the pressure mirrored at the boundary is clamped to be positive.
<b>boundary model</b>	Stress/ Impulse	Impulse features a boundary model enabling coefficient of restitution behavior. All solids in the scene will get a new property <b>Impulse Boundary Handling</b> → <b>Coefficient of Restitution</b> to model elastic response on boundary contact.

**Table 36:** Properties of the snow solver regarding the numerical computations. These properties can be found in the property groups **Physics**→**Snow Solver**→**Numerics**, **Physics**→**Boundary Handling** and **Physics**→**Deletion Criteria**. All of these properties are experimental and only visible if experimental mode is activated.

### 9.2.3 Snow Parametrization

A structured workflow of parametrizing the snow to your specific setup follows:

1. External factors: Change **rest density**, **solid adhesion**, **shear friction factor** such that the snow behaves correctly under external dependent force fields. If you are using **Boundary Handling**→**boundary model**→**Impulse** (*experimental*) modify **Impulse Boundary Handling**→**Coefficient of Restitution** on the solids.
2. Elastic properties: Model the resistance to forces of the snow with **Young Modulus**. Accordingly, the **Poisson's ratio** allows sideways expansion as a reaction

to a force.

3. Plastic properties: **Critical stretch** and **critical compression** control when the plastic behavior sets in. Modify the **hardening coefficient** to change the response caused by plasticity.

Table 37 provides some parametrizations we found helpful. The corresponding snow behaviors are also shown in Figure 25.

Snow type	Parameter	What it does
icy	Young modulus = $5 \cdot 10^5$ Pa	Snow resists with strong elastic response, i.e. snow strongly resists any compression/stretching.
powdery	Critical stretch = $5 \cdot 10^{-3}$ Critical compression = $5 \cdot 10^{-2}$	Snow is able to compress more but will fail much faster when exposed to stretch. Thus, any stretch will immediately lead to plastic behavior.
chunky	Critical stretch = $1.5 \cdot 10^{-2}$ Critical compression = $1.9 \cdot 10^{-2}$	Snow is able to stretch more and will harden much faster when exposed to compression.
watery	Young Modulus = $1.4 \cdot 10^4$ Pa Poisson's ratio = 0.42 hardening = 5	Snow resists with weaker elasto-plastic response. This makes it much easier to compress/stretch. Poisson's ratio will enable splashing behavior.
muddy	Young Modulus = $4.8 \cdot 10^4$ Pa Poisson's ratio = 0.42 Critical stretch = $1.5 \cdot 10^{-2}$ Critical compression = $1.9 \cdot 10^{-2}$ hardening = 5	Snow resists with weaker elasto-plastic response. Poisson's ratio will enable splashing behavior. Snow is able to stretch more but will fail much faster when exposed to compression resulting in chunky snow.

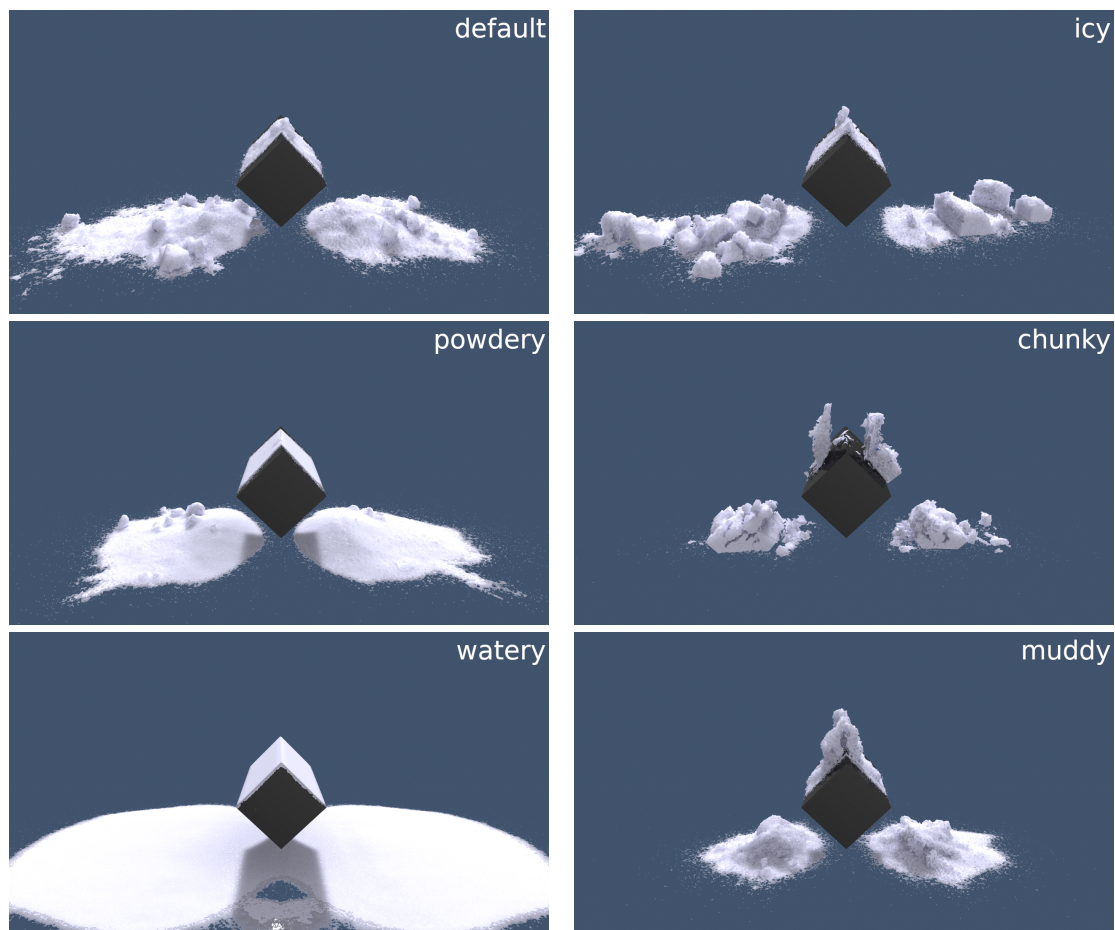
Table 37: Some helpful snow parametrizations.

## 9.3 Preon mesher

The Preon mesher creates a triangle mesh that represents the surface of the fluid.

### 9.3.1 First steps

Insert a Preon mesher. If there are multiple fluids in the scene, you need to connect the *Particles* slot of the fluid to be meshed to the *Particles* slot of the mesher using the connection editor. If there is only a single fluid, the mesher is already connected to the fluid by default.



**Figure 25:** A block of snow falls onto a wedge. The snow used here has a minor deviation from default settings by using a rest density of  $350\text{kg/m}^3$ . The different snow types out of Table 37 are compared.

Right click on the mesher in the scene inspector and click *Mesh frame* to generate the mesh for the current frame. To save the mesh to disk, right click and choose **Save mesh**. To mesh a sequence of frames, right click and choose *Mesh sequence*. A dialog will pop up, asking you for the start and end frame of the sequence. Clicking Ok will start the meshing process.

By default, the mesher will also generate and save meshes when simulating. If the mesher should be used as a post-process, it is recommended to set the **behavior** property of the mesher to **cache**. This will disable meshing during the simulation.

### 9.3.2 Parameters explained

The properties of the Preon mesher are explained per group in the following tables. Note that it is usually not required to change any parameters.



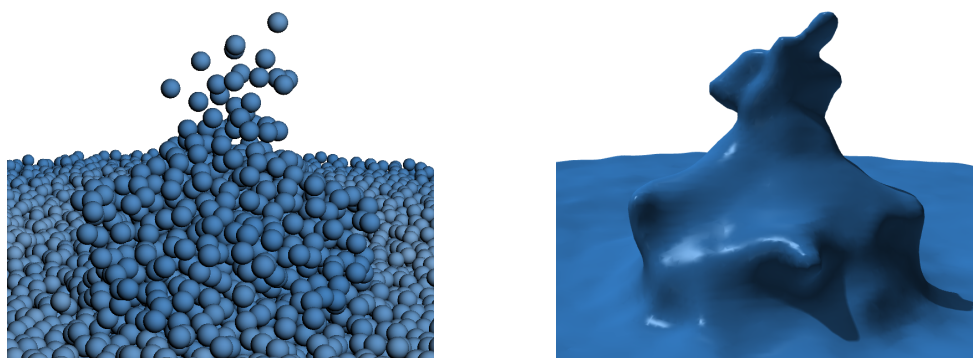


Figure 26: Generating a surface mesh from particles.

Property	What it does
LOD max error multiplier	Sets the maximal allowed error during mesh simplification as a multiplier of the particle radius. Higher errors will result in more adaptive, but possibly also less detailed meshes. The recommended range for this parameter is between 0.1 (high visual quality) and 0.5 (smaller meshes). Note that increasing this parameter has no effect when simplifications are prevented by <b>LOD max normal deviation</b> .
LOD max normal deviation	Sets the maximal allowed deviation between surface normals during mesh simplification. Higher values will result in more adaptive, but possibly also less detailed meshes. The recommended range for this parameter is between 0.001 (high visual quality) and 0.01 (smaller meshes). Note that increasing this parameter has no effect when simplifications are prevented by <b>LOD max error multiplier</b> .
Mesh export format	Set the mesh export format. Note that choosing a text-based format like .obj will result in much bigger file sizes.
Live Preview	Enables or disables live preview after changing parameters. For performance reasons, this is only recommended for small or medium-sized fluids.

Table 38: Properties in group **Meshing**.

Property	What it does
Isosurface threshold multiplier	Higher multipliers will result in thicker fluid meshes. A value between 1 and 1.5 is recommended.
Minimal cell size multiplier	Influences the level of detail of the mesh. Lesser multipliers will lead to more detailed meshes, but will also require more memory and computational power. A value between 0.5 and 2 is recommended (1 being the default).
Smoothing radius multiplier	Controls the smoothness of the generated mesh. A value between 4 and 6 is recommended. Higher multipliers will result in smoother meshes, but will also require more computational power.

Table 39: Properties in group **Distance field**.

### 9.3.3 Common issues

#### Resulting meshes have too many triangles

Multiple parameters influence the triangle count of the final mesh. Increasing **Minimal cell size multiplier** will reduce the mesh size, however it will also decrease the visual quality. Increasing **LOD max error multiplier** and **LOD max normal deviation** are another way to tune the tradeoff between mesh size and quality (see the table above).

A great way to reduce the mesh size without decreasing visual quality is to increase the **Smoothing radius multiplier**. This will lead to smoother meshes that can usually be represented with less triangles. However, it will also increase meshing time and tends to shrink meshes a bit. The shrinking can be countered by increasing **Isosurface threshold multiplier** carefully. It is recommended to tune meshing parameters for a single mesh before meshing an entire sequence. Also note that the default parameters are not a bad choice for most cases and should only be changed if necessary.



## 10 Sources

There are two types of sources, area sources and volume sources. Area sources emit fluid over time from a two-dimensional area, whereas volume sources emit fluid at one specified point in time. The following properties are shared by both volume and area sources:

Property	Unit/Type	What it does
emission particle limit	-	Sets the maximum number of generated particles per emission to prevent allocation of too much RAM. The maximum is given in mega particles, so that a limit of 1 means 1 million particles. For the user, it is sometimes difficult to predict the number of particles a source will generate, especially when using area or volume sources. Generating a massive amount of particles may cause a system freeze or crash on some systems. To prevent this, an artificial particle limit was introduced. Before generating the actual particles, PreonLab will estimate how many particles will be generated. If this estimation is higher than the artificial particle limit, the warning <i>Estimated number of generated particles exceeds user-specified limit</i> is printed in the message window and no particles are generated.
temperature	°C	Sets the initial temperature for the generated particles.

**Table 40:** Properties for all types of sources.

### 10.1 Area source

The Area source emits fluid over time from a defined area that can be specified in multiple ways. For all area types, it supports two types of emission: Continuous emission creates a coherent outflow of fluid while rain emission creates many individual droplets. The following properties control the emission and are available for all area types:

Property	What it does
area type	Determines how the source area and the outflow direction is specified. You can choose between <b>Seedpoint</b> , <b>Rectangle</b> , <b>Circle</b> , <b>FlatJet</b> and <b>Cone</b> . These area types are described in more detail later in this chapter.
emit type	Specifies whether the source should use continuous emission or rain emission.
finite volume	Enables / disables the generation of a finite amount of fluid volume controlled by the <b>volume</b> property.
volume	Sets the total volume that is emitted by the source (m <sup>3</sup> ). Only available if <b>finite volume</b> is enabled.
inflow unit	Sets the unit to be used for the inflow when using continuous emission. You can choose either <b>velocity</b> or <b>volume flow rate</b> . Please note that the magnitude of the selected unit will not be updated if you change the scale of the source.
rain inflow unit	Sets the unit to be used for the inflow when using rain emission. You can select either <b>precipitation height</b> or <b>volume flow rate</b> . Please note that the magnitude of the selected unit will not be updated if you change the scale of the source.
emit velocity	The initial velocity of generated particles (m/s). When using continuous emission, this is only available if <b>inflow unit</b> is set to <b>velocity</b> . When using rain emission, you can always specify the velocity.
volume flow rate	The input fluid volume generated by the source per second (m <sup>3</sup> /s). Only available if <b>inflow unit</b> is set to <b>volume flow rate</b> .
precipitation height	The precipitation height in mm per min which equals L per m <sup>2</sup> per min. This parameter defines the amount of volume generated per square meter. Only available when using rain emission and <b>inflow unit</b> is set to <b>precipitation height</b> .

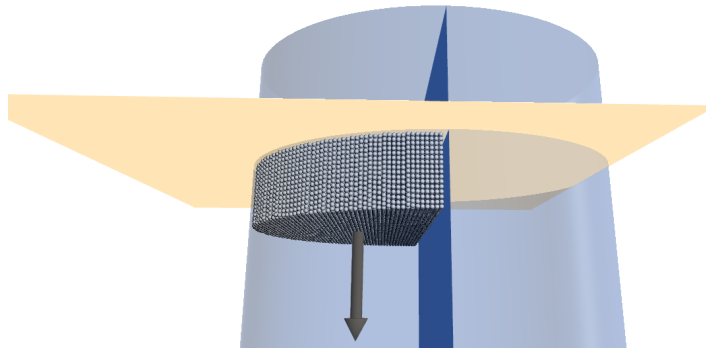
Table 41: Area source properties in group **Settings**.

### 10.1.1 Specifying the source area

The **area type** property controls how the source area is specified. The options **Rectangle** and **Circle** do not introduce any additional properties, the area size is directly given by the scale of the source. When using rain emission, the **Circle** area type can also be used to create an ellipsoid area by scaling the X and Y component separately.

#### Seedpoint

When using the **Seedpoint** option, the area will be limited by solids in the scene. Without solids intersecting the source, the behavior is identical to the rectangular area type. Otherwise, fluid is only emitted from the area that can be reached from



**Figure 27:** In this example, the emit area is bounded by a cylinder and a plane.

the the seedpoint without going through solid boundary. Thereby, only solid objects connected to the area source via the *TriangleMesh* connection are considered. Figure 27 shows an example in which the area is limited by a cylinder and a plane. By default, all solids are connected to the source and the seedpoint is located at the center of the source. It is also possible to connect one or more custom **Point** objects to the source using the *Seedpoint* connection, which will replace the implicit default seedpoint at the center. To do so, you can use the connection editor or the right-click action *Create and connect seedpoint*.

Property	What it does
<b>static source area</b>	If this property is enabled (which is the default), the area is only updated once when starting the simulation. This can improve performance in some scenes, but it must be guaranteed by the user that the shape of the emission area won't change over the simulation time. When using this option, the whole source can still move or rotate during the simulation without restrictions. If the option is disabled, the area of the source is automatically updated (even during the simulation) if necessary, for instance when connected solid objects move or rotate in relation to the source.

**Table 42:** Properties in group **Seedpoint area**.

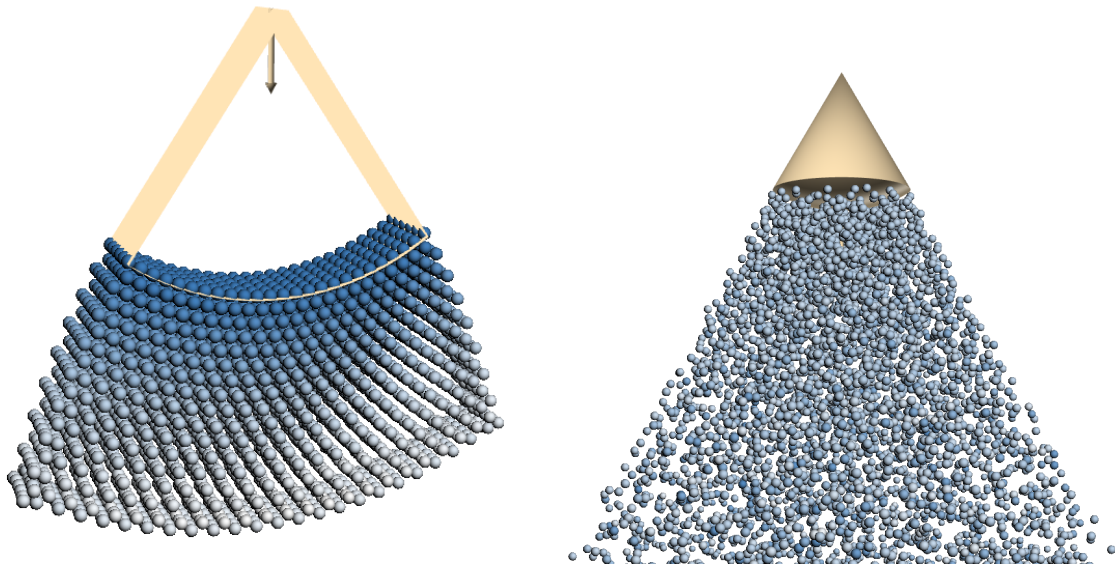
## Flat jet

The **FlatJet** area type allows to emit fluid in a defined angle (see figure Figure 28. The y- and z-scale can be manipulated via the scale dragger. While the y-scale defines the **emission radius**, the z-scale defines the thickness of the flat jet. Additionally, the source area can be specified using the following properties:

Property	What it does
<b>fan angle</b>	The spray angle of the source in degrees. Has to be a value between 0 and 180 degrees.

<b>emission radius</b>	The fluid is emitted at this radius from the center of the source. The larger the radius, the more particles are generated for the given <b>fan angle</b> .
------------------------	---

**Table 43:** Properties in group **Flat jet area**.



**Figure 28:** Two area sources with different emission and area types. Left: continuous emission and area type set to **FlatJet**. Right: rain emission and area type set to **Cone**.

## Cone

The **Cone** option allows to emit fluid from a cone with a defined opening angle.

Property	What it does
<b>cone angle</b>	The opening angle of the cone in degrees. Has to be a value between 0 and 180 degrees.

**Table 44:** Properties in group **Cone area**.

## 10.2 Volume source

The volume source is used to fill a user-specified volume with particles.

Property	What it does
source spacing	Distance between generated particles. In almost all cases, this should be equal to the solver spacing. Please note that changing this does not automatically change the solver spacing.
emit frame	The frame at which the volume source emits its particles. The resulting time depends on the simulation frame rate.
emit velocity	The initial velocity of generated fluid particles.
fill method	Specifies <i>how</i> the volume is sampled with particles. Fill method <b>uniform</b> will generate particles aligned in a uniform grid. Fill method <b>hexagonal</b> will align the particles in a hexagonal grid. This method is optimized for filling a box and generates one layer of particles at the top that is not strictly located within the volume. The fill method <b>quality</b> uses a combination of techniques to fill the volume as accurately as possible, minimizing void spaces due to discretization. It is the recommended method for most scenarios, but it also takes more time to generate the fluid. Fill method <b>poisson_hybrid</b> is only recommended when filling objects using fill type <b>inside</b> or <b>seedpoint</b> and aims to capture the surface of filled objects as accurately as possible. It first samples the surface using maximal poisson disk set sampling and then fills the rest of the volume using uniform sampling. <b>poisson_full</b> uses poisson disk set sampling not only for the surface, but for the whole volume. This is usually not recommended, because it requires a lot of time to compute and results in a low density compared to other fill methods. Finally, <b>poisson_surface</b> only samples the surface and generates no other particles.

Table 45: Volume source properties in group **Settings**.

Property	What it does
fill type	Defines the volume in which particles are generated. In any case, particles are never generated outside the source box. Fill type <b>all</b> will fill the source box completely with particles. Fill type <b>inside</b> will generate particles only inside of objects. This only works properly with volumetric and closed meshes. For meshes that contain holes or self-intersections, it may give unexpected results. The same applies to fill type <b>outside</b> , which generates particles outside of objects. Finally, fill type <b>seedpoint</b> can be used to fill all regions that can be reached from one or multiple user-specified seedpoints. Read more about seedpoints in Section 10.2.2.
manual border size	Enables or disables manual specification of the border size. By default, this is disabled and the border size will be set automatically to ensure that generated fluid does not overlap with geometries.

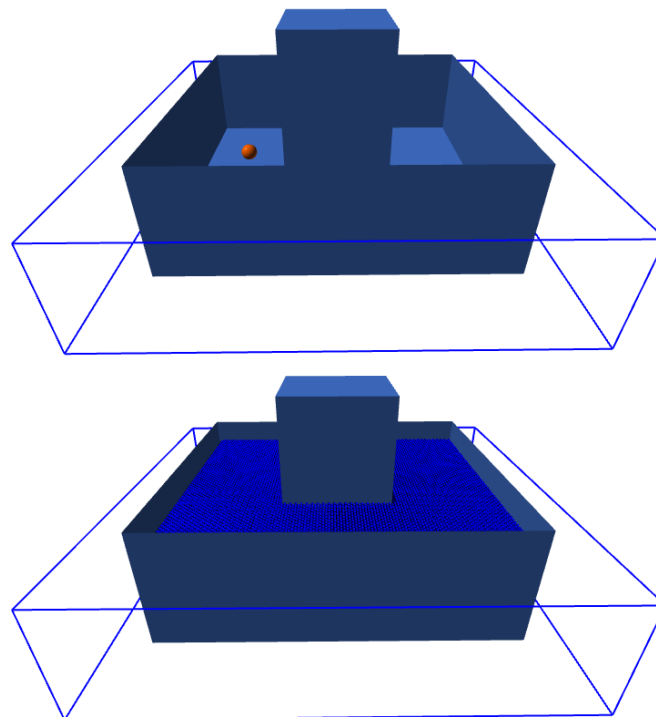
<b>border size</b>	Specifies a border between the surface of objects and particles generated by the source. This has no effect if fill type all is used. The border size is typically used to avoid collisions between fluid and boundary particles when starting the simulation. In this case, it should be set to the spacing of the fluid. When using the no-gap option in the fluid solver, it should be set to half the spacing of the fluid.
--------------------	---

**Table 46:** Volume source properties in group **Volume settings**.

### 10.2.1 Preview of generated particles

Before starting the simulation, you should check whether the fluid particles are generated as expected. To view the particles, right-click on the volume source in the scene inspector and choose *Regenerate volume*. You can clear the particles by right-clicking on the volume source again and choosing *Clear preview*. The preview will also clear automatically when you start the simulation and the volume source emits its particles.

### 10.2.2 Filling containers with a volume source



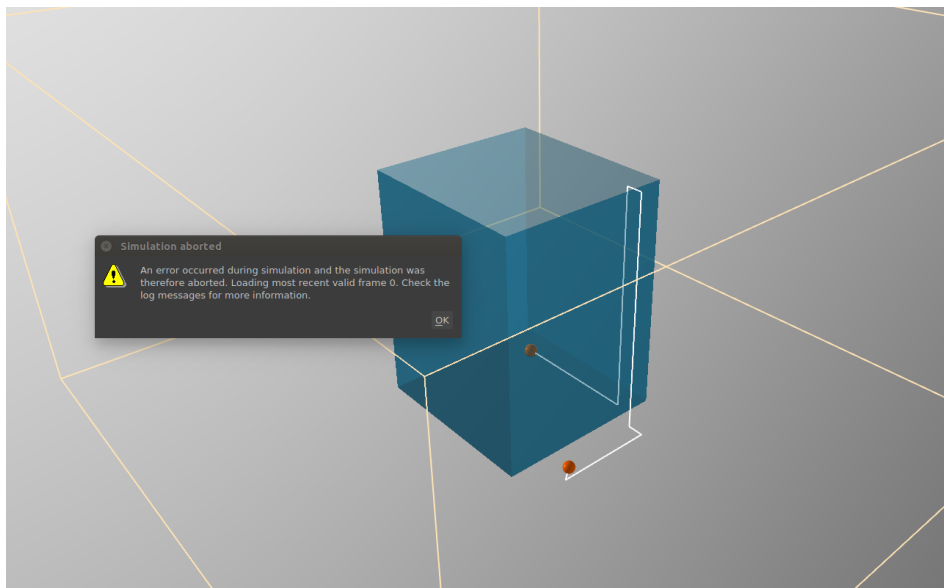
**Figure 29:** Top: Seedpoint (orange dot) placed in a box with obstacle in the middle. Bottom: Box filled with particles.

Seedpoints are a powerful tool to fill arbitrary containers with fluid. The volume source will fill all regions that can be reached from the seedpoints without passing

through an obstacle or the boundaries of the volume source itself. Figure 29 shows a box filled with fluid using seedpoint filling.

To activate filling from seedpoints, set the **fill type** property to **seedpoint**. If no custom seedpoint is connected to the volume source, a single implicit seedpoint located in the middle of the source is used for filling. To insert a custom seed point, click *Add→Transform→Point*. If there is a single volume source in the scene, inserted seedpoints are connected automatically to it. Otherwise, you need to setup the connection manually using the connection editor (slot *Seedpoint*).

### Finding holes in your geometry



**Figure 30:** Error after starting the simulation because the badpoint was reached during filling. A white line indicates the connection between seedpoint and badpoint.

Sometimes, seedpoint filling can lead to unexpected results. Consider the following example: You want to fill a complex gearbox geometry with fluid and you setup a volume source and seedpoint accordingly. Now you start the simulation and notice that the volume source did not only generate fluid inside the gearbox, but everywhere inside the volume source box. Usually, this means that there is a tiny hole somewhere in your geometry which connects the inside of the gearbox with the outside and therefore prevents a sensible filling. In order to find this hole, you can use the *badpoint* functionality offered by the volume source. A badpoint represents the opposite of a seedpoint: You place it where you *don't* expect any fluid. If the volume source reaches the badpoint during the seedpoint filling process, it does not generate any particles but instead plots a path between a seedpoint and the badpoint. By tracing the path you can usually quickly find the unexpected hole in the geometry.

To specify a badpoint, insert a **Point** object and connect it to the *Badpoint* input slot of the volume source. Figure 30 shows how a started simulation terminates if the badpoint is reached during filling. Without the badpoint, the entire volume source box would be filled with particles.

## Make the volume source ignore geometries

The volume source also takes inactive geometries (solids) into account. You can make the volume source ignore certain geometries by manually deleting the connection *TriangleMesh* from geometry to volume source which is per default automatically added by PreonLab.

## Special notes on box filling

To ensure optimal filling of a **Box**, make sure to follow these rules:

- The size of the box should align with the particle spacing. Right-click on the box in the scene inspector and choose *Adjust scale to spacing* in order to round the scale to the next aligned value.
- Connect the *Align* slot of the box to the volume source to ensure optimal particle alignment.
- Use hexagonal filling to avoid splashes during the first simulation steps.

### 10.2.3 Alignment

The volume source is internally rasterized using a grid. By default, the position of the volume source defines the alignment of particles, i.e., determines the particle positions relative to the volume source. It is also possible to specify an alignment independent from the position of the source by connecting an object to the Volume source via the *Align* slot. For instance, the **Box** offers an *Align* output slot that ensures an optimal alignment of particles inside the box. It is also possible to specify a manual alignment by connecting a **Point** to the Volume source via the *Align* slot. The position of the point will now determine the particle alignment.

Note that alignment has no effect when using *poisson* particle sampling. When playing around with alignment, make use of the particle preview to get feedback.

### 10.2.4 Specifying initial velocities

By default, the **Volume Source** assigns a constant velocity given by the **emit velocity** property to all particles. It is also possible to specify a velocity field via the *VelocityField* connection input slot of the source. The velocity field can be given by a moving solid or **Point Cloud Resource** that stores three-dimensional samples imported from a CSV file (see Section 17.8.2). For instance, in order to use the velocity field defined by a moving solid object, just connect the *VelocityField* output slot of the solid to the input slot of the source. If a velocity field is connected this will override the **emit velocity** property.



## 10.3 Rain source

The rain source mimics a rain like inflow.

Property	What it does
emit velocity	The initial velocity of generated drops. Note that this value does not change the volume flow rate. In order to mimic real rain, you should use the terminal velocity of a raindrop (2 mm radius) which is $\approx 9$ m/s.
precipitation height	The precipitation height in mm per min which equals L per m <sup>2</sup> per min. This parameter defines the amount of volume generated per square meter.
shape	The shape of the rain source. Either rectangular or ellipsoid.
specify drop diameters	Defines whether diameters of emitted drops can be set manually or not. If disabled, all emitted drops have a fixed diameter equal to the fluid spacing. If enabled, the rain source emits drops of fluid across a rectangular or ellipsoid area. These raindrops can have a fixed or a varying diameter distributed with a normal distribution. Let $d_{min} = \text{min drop diameter}$ and $d_{max} = \text{max drop diameter}$ . The mean of the normal distribution is $d_{mean} = \frac{d_{min} + d_{max}}{2}$ . The standard deviation is $\frac{d_{max} - d_{min}}{3}$ while a drop will never be smaller than $d_{min}$ or bigger than $d_{max}$ .
min drop diameter	The minimum diameter of emitted drops in meters.
max drop diameter	The maximum diameter of emitted drops in meters.

Table 47: Properties of rain source.

## 11 Boundary Domains and Conditions

PreonLab offers possibilities to define a boundary domain, i.e., a domain where the behavior of fluid particles or boundary particles is prescribed by the user.

### 11.1 Box and cylindrical domain

The **Box domain** defines a box-shaped boundary domain, while the **Cylindrical domain** defines a cylindrical-shaped domain. Except for the shape, both domains behave identically. There exist three control types: **deletion of fluid or rigid particles**, **scripting the velocity of fluid particles** and **refinement** of fluid particles.

The user can define whether this condition holds **inside** or **outside** the boundary domain. The color indicates which region is set for the domain. A green color means that the simulation domain is inside the boundary domain, particles outside are handled by the boundary domain, while a red color indicates that particles inside the domain are treated by the boundary domain.

A typical use case for these objects is to define a simulation domain. For example, by using a **Box domain** with region set to *outside* and type set to **deleteFluidParticles**, the user restricts the simulation domain to the extend of the **Box domain**.

Property	What it does
region	Specifies whether particles outside (the default) or inside the box should be removed.
types	Specifies the type of the boundary condition. If you want to delete particles, there are three options: Choose <b>deleteFluidParticles</b> to delete only fluid particles, <b>deleteRigidParticles</b> to delete only rigid particles or <b>deleteAll</b> to delete both fluid and rigid particles. If you want to delete rigid particles you need to additionally connect the <i>DeletionDomain</i> slot to the respective solid. If you want to specify a constant fluid velocity inside the domain, choose <b>scriptFluidVelocity</b> . To refine coarse fluid particles into finer particles select <b>refine</b> (See Section 9.1.10 for more details).

**Table 48:** Properties of the Box domain and Cylindrical domain.

Property	What it does
fluid velocity	Specifies the velocity of fluid particles controlled by the domain. This velocity is given as global value independent of the rotation or <i>Transform</i> parents of the domain.

**Table 49:** Properties in the group *Velocity Condition*. This group is only shown if the control type is set to **scriptFluidVelocity**.

### 11.1.1 Using meshes to define the domain volume

For the **Box domain**, it is also possible to restrict the volume in which fluid is influenced by the domain with custom meshes. To do so, connect the *TriangleMesh* output slot of one or multiple solids to the corresponding input slot in the domain. Then right-click on the domain and select *Regenerate volume*. The properties in the group **Volume settings** specify how the meshes are considered when the volume is generated. This works mostly the same way like it does for the volume source. The domain will only be updated once automatically (on first use, for example after loading a scene). In all other cases, it is required to explicitly trigger the recomputation using the right-click action.

Property	What it does
fill type	See Table 45 for more information.
manual border size	See Table 45 for more information.
border size	See Table 45 for more information.
volume generation frame	Defines the (view) frame in which the volume is generated. The volume will never be regenerated during post-processing or simulation, however it will be transformed dynamically according to the domain position and orientation.

**Table 50:** Box domain properties in group **Volume settings**.

### 11.1.2 Using multiple domains

PreonLab follows the following rules when combining the conditions of multiple domains:

- A particle is deleted if it is inside a deleting domain with region **inside** or if its velocity is above the threshold of any maximum velocity condition.
- A particle is also deleted if it is outside a deleting domain with region **outside** and not inside a deleting domain with region **outside**.
- A particle is never scripted if it is deleted.

- A particle is scripted if it is inside a scripting domain with region **inside**. If this is true for multiple domains with different scripting velocities, the resulting velocity is undefined.
- A particle is also scripted if it is outside a scripting domain with region **outside** and not inside a scripting domain with region **outside**. If this is true for multiple domains with different scripting velocities, the resulting velocity is undefined.

### 11.1.3 Using boundary domains to improve performance

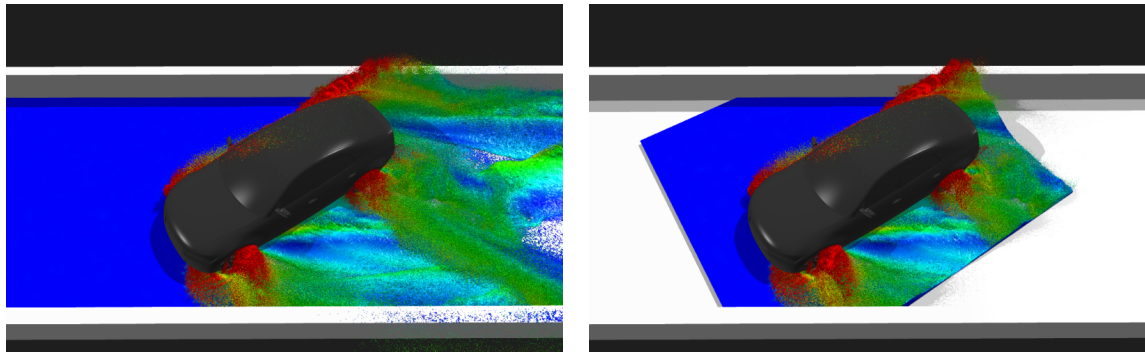
A box domain scripting the velocity of fluid particles can not only be used to achieve a specific fluid behavior, it can also be used to improve performance in certain scenes. This is possible because scripted particles require almost no simulation effort by the solver and are therefore processed very efficiently. Whenever there are parts of your simulation where fluid particles move with a constant velocity or do not move at all, there may be an opportunity to speed up the simulation using a boundary domain. To make optimal use of the domain, it may be necessary to keyframe its scale and / or position.

#### Example

Consider a simulation in which a car drives through water on a street. If you are only interested in the water interacting with the car, you can restrict the simulation to a box around the car. To achieve this, insert a box domain and set its **type** to *scriptFluidVelocity* and its **region** to *Outside*. Then connect the *IOTransform* slot of the car to the corresponding slot of box domain and position and scale the box domain so that it contains the car. Now, only fluid in the box surrounding the car will be simulated by the solver. Note that you should leave a generous safety gap between the front of the car and the boundary domain, or else fluid particles might pile up. The safety gap should at least be as wide as the car's length. Also note that particles leaving the box will be frozen, which leads to unrealistic results for water behind the car. You can disable the visualization of these particles by turning the solver property **render scripted particles** (located in the Appearance group) off. If you need a full drive-through simulation including water behind the car, you can still get a speedup using boundary domains by scripting fluid in front of the car.

### 11.2 Maximum velocity condition

The **maximum velocity condition** deletes fluid particles based on their velocity and not based on their position. This can be used to remove very fast particles from the simulation in order to improve performance and stability.



**Figure 31:** Use-case for simulating a restricted domain around a car to save memory and increase performance. Comparison of unrestricted (left) to restricted simulation domain (right).

Property	What it does
velocity magnitude	Specifies the maximum velocity magnitude, fluid particles above that will be deleted.

**Table 51:** Maximum velocity condition.

### 11.3 Air Objects

Air Objects provide boundary conditions that can be employed to model the interaction of fluids and solids with an air phase for two scenarios: thermodynamics and evaporation. They have to be connected to all fluids and solids that should interact with it via the *Air* slot.

Property	Unit/Type	What it does
Temperature	°C	Sets the temperature (can be varied over time via keyframes).
system type	-	Defines whether and how the air phase thermally interacts with fluid or solids. If set to <b>None</b> , it is not considered in the thermodynamics computation at all. Setting the <b>system type</b> to <b>ClosedFixed</b> allows to thermally interact with fluids that have their <b>system type</b> set to <b>Closed</b> . However, the temperature of an air object is set back to a fixed (or keyframed) temperature after the diffusion update. Thus, the air object behaves like a constant heat source (or sink).

**Table 52:** General properties in group **Physics**.

### 11.3.1 Evaporation with air objects

The speed at which water vapor is evaporated into the air depends on a set of properties which are all defined in Table 53. An additional degree of freedom comes with **water vapor control mode**. If set to **Humidity**, the **relative humidity** is set manually (either as a constant or keyframed over time). With modes **Volume** or **TriMesh**, the **relative humidity** defines the initial value at the start of the evaporation phase. Together with the **volume** PreonLab derives the initial amount of water vapor of the air object. During the evaporation the relative humidity increases automatically based on the amount of evaporated water vapor mass until the air is completely saturated. **Note:** The **volume** property is not automatically derived from the connected meshes or the bounding box of the air object itself. It has to be set by the user manually.

Multiple air objects can be defined within the scene. Therefore, it is important to understand how the interaction of particles and air objects can be specified. First, you have to connect the air objects to the solvers and solids they should interact in general. Second, you have to specify a region in the scene where fluid or film particles have to reside to be considered. This depends on the **water vapor control mode**. If set to **TriMesh**, the bounding boxes of triangle meshes connected to the *TriangleMesh* slot are employed for assignment. Otherwise, the bounding box of the air objects is employed.

Property	Unit/Type	What it does
pressure	Pa	Sets the air pressure. The default is 101325 Pa. This equals the pressure for the ICAO Standard Atmosphere at 0 meter above mean sea level (MSL) at 15 °C.
wind speed	m/s	The wind speed tangential to the free surface of the fluid.
water vapor control mode	-	Defines how the relative humidity of the air is determined. If set to <b>Humidity</b> , it is defined by the user (either as a constant or keyframed over time). If set to mode <b>Volume</b> or <b>TriMesh</b> , it is computed based on the system state (see Section 11.3.1).
relative humidity	%	Sets the relative humidity of the air. The default is 0 percent.
volume	m <sup>3</sup>	Defines the volume that is occupied by the air object.

Table 53: Properties in group **Physics** → **Evaporation**.

### 11.3.2 Thermodynamics with air objects

The property group **Physics** → **Thermodynamics** becomes visible as soon as **system type** is set to **ClosedFixed**. Its members are listed in Table 54.

Property	Unit/Type	What it does
heat capacity	J/(°C kg)	The specific heat capacity of the substance on a per mass basis, i.e., the isobaric mass heat capacity. The default is 1012.0 (air at 25°C).
thermal conductivity	W/(m °C)	The property of a material to conduct heat. The default is 0.026 (air at 25°C).

Table 54: Properties in group **Physics** → **Thermodynamics**.

## 11.4 Experimental: Open boundary plane

An open boundary plane is a plane on which a boundary condition of zero velocity gradient is applied. The fluid particles exiting this plane are deleted and simultaneously a zero velocity gradient condition is maintained. This feature may be applied on boundaries where the velocity fluctuations along the direction normal to the plane is expected to be small which ensures that the velocity upstream of this plane is almost undisturbed by its introduction. In order to maintain a zero velocity gradient on the plane, the velocities and positions of the particles contained within a box upstream to this plane are used and a group of so-called ghost particles is created with correspondingly similar velocities in a similar box downstream to this plane. The thickness of each of the two boxes adjacent to the plane is controlled by the **thickness** property. The particles exiting the plane are deleted and are replaced by those ghost particles in the downstream box. The positions of the ghost particles are found by mirroring the positions of the corresponding real fluid particles in the upstream box with the open boundary plane as the mirroring axis, whereas the velocities of the ghost particles are calculated from the velocities of the real fluid particles based on the type mentioned in the property **velocity**.

Property	What it does
thickness	Defines the thickness of each of the boxes on either side of the plane. This is specified as a factor of the particle spacing.
rest density	If switched on, the density of all the ghost particles are set to rest density.
velocity	Defines the way the velocity of the ghost particles in the downstream box should be calculated from those of the particles from the upstream box. The types can be <b>copy</b> , <b>project</b> or <b>interpolate</b> . The type <b>copy</b> copies the complete velocity of the particle to be mirrored, whereas the type <b>project</b> just uses the part pointing in normal direction of the plane. Finally, the type <b>interpolate</b> does an SPH interpolation of the velocity instead of using the velocity of the single particle to be mirrored.
spacing factor	The factor in terms of particle spacing that decides the gap between a real fluid particle and its correspondingly mirrored ghost particle.

collision factor	The factor in terms of particle spacing that decides the distance the mirroring axis can be shifted to avoid a fluid particle and a ghost particle sharing the same position.
------------------	---

Table 55: Properties of **Open boundary plane**.

## 11.5 Outflow domain (Prototype)

The **Outflow domain** allows to prescribe a fix outflow of fluid. For this, place the **Outflow domain** at the location where the fluid should flow out. The type of outflow condition can be specified by changing the **General**→**outflow type** (see Table 56).

**Note:** Currently, you need to manually make sure that the **Outflow domain** is scaled correctly in the z-direction. To guarantee correct behavior, **Transformation**→**scale**→**Z** should be at least three times the particle spacing of the solver. This scaling will be done automatically in a future version.

Property	What it does
outflow type	Defines the type of boundary condition of the outflow domain This can be <b>velocity</b> , <b>flowrate</b> or <b>continuative</b> .
min target speed	Only relevant if <b>outflow type</b> is set to <b>velocity</b> . Particles with lower speed will get this speed prescribed, keeping the direction of the velocity.
max target speed	Only relevant if <b>outflow type</b> is set to <b>velocity</b> . Particles with higher speed will get this speed prescribed, keeping the direction of the velocity.
direction deletion threshold	Only relevant if <b>outflow type</b> is set to <b>velocity</b> or <b>continuative</b> . Value between 0 and 1 encoding how many percent of the particle velocity has to be coherent to the direction of the outflow. For lower coherence, the particle is deleted.
target flow rate	Only relevant if <b>outflow type</b> is set to <b>flowrate</b> . The maximum flow rate of fluid the domain deletes from the simulation. This value is reached, when the complete area of the domain is filled with fluid. This means, that if only 50% of the area is filled with fluid, only half of the target flow rate will be pumped out. The unit of this property is m <sup>3</sup> /s.
shape	Allows to change the shape of the outflow domain to either <b>box</b> or <b>cylinder</b> .

Table 56: Properties of **Outflow domain**.



### 11.5.1 Connection to sources

The **Outflow domain** has a *FlowRate* output slot. This slot can be connected to one or multiple sources. The connection then results in the sources emitting the same amount of fluid as is deleted by the **Outflow domain**. Note, that if there is a *FlowRate* connection, the **outflow type** of the outflow domain is automatically set to **flowrate**. Furthermore, the **emit type** of the connected sources is automatically set to **rain**. Please note that adding or deleting these connections during simulation might lead to an erroneous behavior.

## 12 Force Fields

### 12.1 Gravity

The gravity object applies a force to all objects it is connected to based on a constant and global acceleration defined by the property **gravity**. By default, each scene contains a gravity object that is automatically connected to all physical objects including fluids and rigids. Its gravity property is always in the global reference frame. By default, it is set to  $9.81 \text{ m/s}^2$  directed into the negative z-direction to approximate earth's gravity. You need to change this direction if the z axis is not your up-axis or objects will fall to the side and not down.

### 12.2 Drag Force

The interaction between a fluid and the surrounding air phase is in reality normally not visible but can influence the overall flow of the fluid significantly. Especially the path and terminal velocity of single fluid droplets is dependent on the surrounding air velocity. By default, the air phase is not simulated in PreonLab. In order to approximate the effects of the air onto the fluid, a drag force can be used. The drag force object is used to simulate air resistance acting on fluids. It has no effect on rigid objects. The drag force allows to specify a single global air flow velocity. If you want to specify an air flow field where the air velocities differ in space, see Section 12.3.

Be aware that if you insert a drag force as well as an air flow force field without restricting their area of effect, both force fields will act on the fluid, resulting in a doubling of the force. This is probably incorrect for your scene. See Section 12.3.7 for more information.

The force applied to a fluid particle depends on the relative velocity difference between the air and the particle at this position. Furthermore, for each fluid particle, its area exposed to the air is computed. This means that the drag force does not affect particles that are located inside a fluid volume but only these on the surface. A typical effect of the drag force on fluid simulations is a reduction of splashes.

Property	What it does
<b>velocity</b>	The velocity of the air in the local coordinate system. Note that the zero vector is a perfectly valid choice, because the applied force depends on the velocity difference between fluid and air and not only on the air velocity.
<b>apply force everywhere</b>	Defines if the force should be applied everywhere or just in the defined box region.
<b>drag model</b>	Specifies how the drag coefficient and the (unobstructed) particle area is calculated. The different options are discussed in more detail in the subsections below.
<b>Cd</b>	The constant drag force coefficient. Only relevant if <b>drag model</b> is set to <b>Constant</b> .
<b>terminal velocity</b>	Specifies the terminal velocity a single fluid particle should achieve. Only relevant if <b>drag model</b> is set to <b>TerminalVelocity</b> .
<b>force factor</b>	Factor multiplied to the resulting drag force. This is only shown and applied when as drag model either <b>Constant</b> , <b>TerminalVelocity</b> or <b>AutomaticViaTerminalVelocity</b> is selected.
<b>density</b>	Density of the air modeled by the drag force. This is only relevant and shown in the <b>Liu Model Settings</b> group when <b>drag model</b> is set to <b>LiuModel</b> .

**Table 57:** Properties for the Drag force.

In the following subsections, the different drag models are discussed in more detail.

### 12.2.1 Constant

If this drag model is selected, the used drag coefficient is given by the user. The base area of the fluid particle is assumed to be the squared spacing.

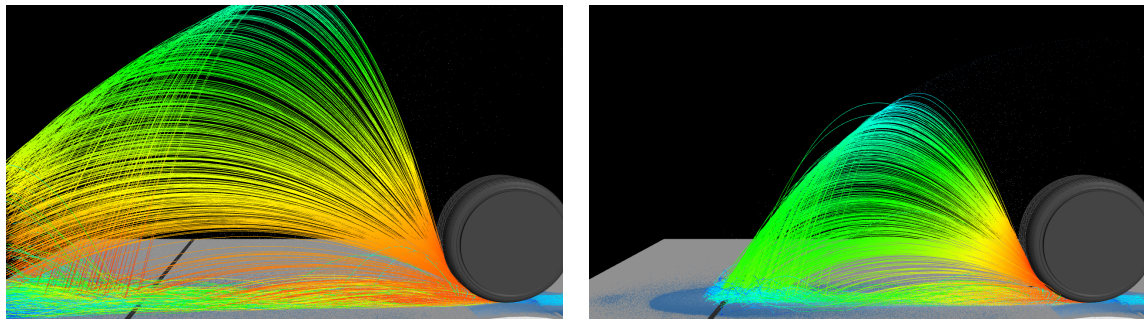
### 12.2.2 Terminal Velocity

The user can specify a terminal velocity a single fluid particle should reach in free fall. Depending on this, a constant (independent of the relative velocity or other factors) drag coefficient is computed. The base area of the fluid particle is assumed to be the squared spacing.

### 12.2.3 Automatic Terminal Velocity

A formula is used to compute the terminal velocity based on the particle spacing. This terminal velocity is then used as described in the **TerminalVelocity** model to compute

a constant drag coefficient. The base area of the fluid particle is assumed to be the squared spacing.



**Figure 32:** Fluid spray for a wheel rotating with a speed corresponding to 20 km/h displayed using path lines. In the left image no drag force is enabled while in the right image the Liu Model is used. The result from the right image matches experiments.

#### 12.2.4 Liu Model

This drag model is based on a paper by Liu et al.<sup>1</sup> In this model, the deformation of a single fluid particle is taken into account. Depending on the relative velocity between air and fluid, the fluid particle is modeled to deform. This influences the drag coefficient as well as the cross-sectional area of a particle.

### 12.3 Air Flow

The **Air Flow** field is an extension of the drag force and inherits most of its properties. However, instead of specifying a single global velocity, velocities are stored in a 3D grid and are interpolated during the simulation. The purpose of this object is to couple PreonLab fluid simulations with air flow fields simulated in another software.

PreonLab supports the import of an air flow field via the csv format or the EnSight Gold format. On import, PreonLab will parse the original file, convert it to the binary *.prairflow* format and finally create an air flow force field. You have to specify the **Output Path** to the location where you want to store the converted file. The dimensions of position and velocity data can be re-interpreted on import via the **Position Scaling** and **Velocity Scaling** input parameters. Note that PreonLab uses SI units, i.e., positions are given in meter and velocity in meter per second. Thus, if the positional data in your original file is given in millimeter, **Position Scaling** should be set to 0.001 to convert to meter. **Velocity Offset** can be set to a non-zero vector to transform the input velocity field from a moving to a fixed reference frame.

---

<sup>1</sup>Modeling the Effects of Drop Drag and Breakup on Fuel Sprays, Liu A., Mather D., Reitz R., 1993

### 12.3.1 Static air flow data import via the csv format

For files in csv format, each line in the csv file should contain position and velocity for one sample point. The first line should contain a text per column which describes the corresponding value. The order of position and velocity values is not important and can be adjusted when importing the data. Any additional data will be ignored. To import your csv data, click *File*→*Import*→*Import Airflow*. You can select your csv file by setting the path. PreonLab will then try to automatically detect the correct separator and fill the drop down lists for position and velocity components. If the order is not correct, you can reassign it, such that the order matches your csv data. PreonLab will convert the csv file into a binary file with the file-ending *.prairflow*.

### 12.3.2 Static air flow data import via the EnSight Gold format

To import files in EnSight Gold format, provide an EnSight Gold *.case* file in the path field. The import dialog then displays the respective input fields (see Figure 33). First, the **Time** frame has to be specified that should be imported. For transient data the drop-down list contains more than one entry. Second, the **Velocity File** has to be selected. Note that only variable files with vector entries are listed in the drop-down list. All other input fields are similar to the ones provided for *.csv* file import. The input fields are pre-filled if the *.case* file could be parsed successfully.

### 12.3.3 Transient air flow data import

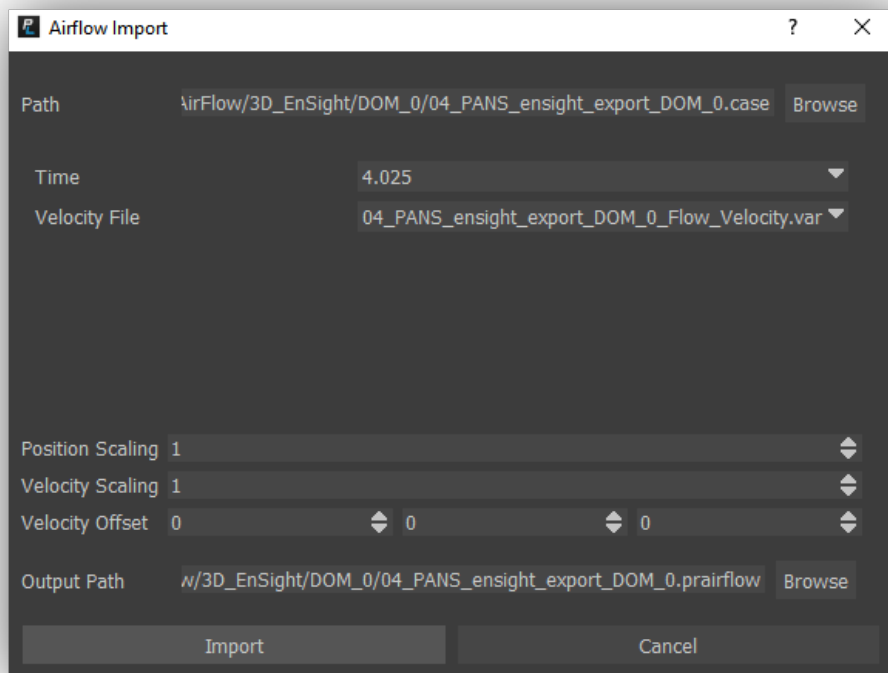
The import dialog for transient airflow data requires the EnSight Gold format as input format. The input fields are pre-filled if the *.case* file could be parsed successfully (see Figure 34). The time range which is covered by the transient data is presented with **Start Time** and **End Time**. The range can be narrowed down. Next, a **Velocity File** has to be selected. Note that only variable files with vector entries are listed in the drop-down list. Finally, an output directory has to be chosen. The EnSight Gold data is converted to an internal data format for PreonLab (i.e., *.prairflow*).

After clicking the *Import* button, the selected variable file is converted for all time frames available within the given time interval and an **Air Flow** object is created. For every instance in time of the simulation, the **Air Flow** object will load the appropriate airflow files and interpolate accordingly to compute the air velocity for positions inside the **Air Flow**.

### Limitations

Currently, the following limitations are known:

1. Structured geometries (i.e. blocks) are not yet supported and may not be used in geometry files. In contrast, all unstructured geometries (e.g. polygons and polyhedra) are supported.



**Figure 33:** The import dialog for a static **Air Flow** field represented with the EnSight Gold format.

2. Individual timeset numbers for variable files are not yet supported. Therefore, we suggest to only have one timeset specified in the .case file.
3. The keywords "conn\_change", "coord\_change" and "no\_change" are not yet supported for "part" sections in the geometry file.
4. Variable files must be of type "scalar per node", "scalar per element", "vector per node" or "vector per element". Other types are not yet supported.
5. Placeholders (i.e. wildcards) for the frame numbers may only be used as part of the file or folder name, but not in the file extension. See the asterisks in the .case file example.

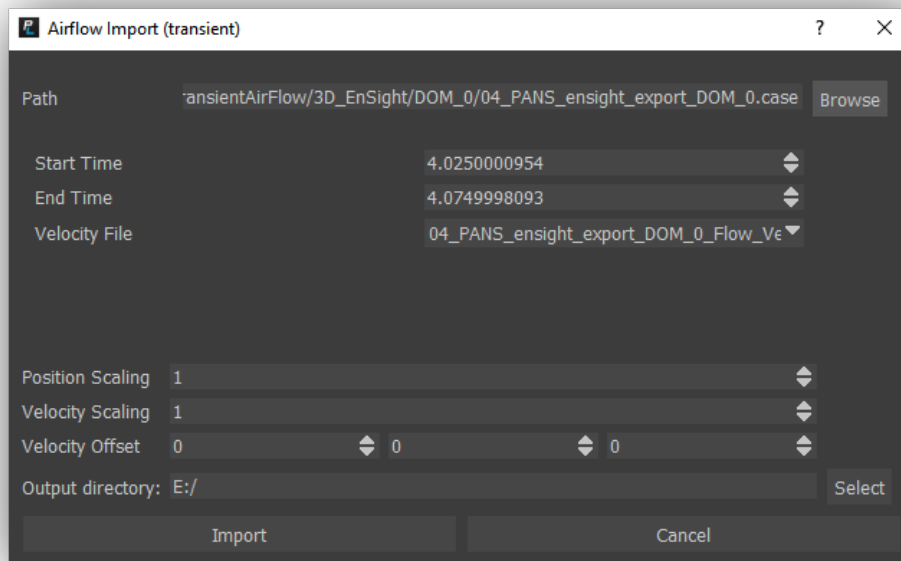
### Valid .case file example

A .case file may look like follows:

```

FORMAT
type:  ensight gold
GEOMETRY
model: 1 *****/my_example.geo

```



**Figure 34:** The import dialog for transient airflow data represented with the EnSight Gold format.

```
VARIABLE
vector per element:      1 Flow_Velocity *****/my_example_Flow_Velocity.var
scalar per element:      1 Flow_Fk_ratio *****/my_example_Flow_Fk_ratio.var
TIME
time set:                 1
number of steps:          3
filename start number:    1
filename increment:       1
time values:
  0.1025000000E+01
  0.1050000000E+01
  0.1075000000E+01
```

### 12.3.4 Viewing the air flow field

You can view the imported sample vectors by enabling the **show sample points** property. It will simply display the imported samples as colored arrows. Be careful though, because for very large data sets this visualization may require too much GPU memory for your system. Another way to visualize the air flow field is the **Air flow visualizer** object which displays air flow velocities projected on a plane. This visualization does not show you the raw samples, but the interpolated data that is used by PreonLab during simulations. Only this visualization allows you to tune properties like **cell size**. For further information, see Section 16.14.

### 12.3.5 Air flow parameters

Property	What it does
air flow path	Sets the path to the <i>.prairflow</i> file created by PreonLab that stores the sample points in binary form.
velocity offset	This allows to add an offset to all imported velocity samples. This is for example helpful if the air flow was computed in a moving reference frame (moving objects are kept at the same position) and in PreonLab the reference frame is fixed (objects move).
num lod levels	Imported air flow fields may be sampled adaptively. To interpret the imported adaptive point cloud correctly, PreonLab needs to know the number of detail levels. Note: A high number of levels will result in slower grid construction time. This property is only relevant when using the grid-based velocity storage.
discard samples	Sets whether samples should be rejected if they are classified as outliers.
maximal deviation	This parameter is only visible if standard deviation is selected for discarding samples. It sets the maximal allowed deviation of a sample from the mean.
max. vel. length	This parameter is only visible if fixed value is selected for discarding samples. It sets the threshold above which samples will be discarded.

Table 58: Properties in group **Point cloud import**.

Property	What it does
cell size	Sets the minimal cell size in the airflow grid. This property is only relevant when using the grid-based velocity storage.
cell limit	Sets the maximum number of cells (in millions) for the top grid level to prevent allocation of too much RAM. This property is only relevant when using the grid-based velocity storage.
velocity storage implementation	Specifies how the air flow velocity field is represented internally. <b>Grid</b> will represent the velocity field using a uniform grid, that allows fast access but can consume a lot of memory. Also, the imported air flow may not be represented accurately if the cell size is not small enough. <b>Gridless</b> uses a particle-based representation for the velocity field which accurately captures the imported air flow and typically requires less memory.

Table 59: Properties in group **Air Flow Storage**.



Property	What it does
air velocity	Sets the velocity that is used in areas where the grid stores no data.

**Table 60:** Properties in group **General**.

Property	What it does
show sample points	When enabled, velocities at the positions of the imported sample points will be visualized. The velocities displayed at each respective location is not taken from the imported file but instead interpolated from the constructed velocity grid.

**Table 61:** Properties in group **Appearance**.

In case **show sample points** is enabled, the subgroup **Sample Points** becomes visible for additional options on how to visualize the velocities at the sample point positions:

Property	What it does
normalize arrows	When enabled, the velocities shown at the positions of the imported sample points will be normalized.
arrow scale	The velocities displayed are scaled by this factor.

**Table 62:** Properties in subgroup **Sample Points**.

### 12.3.6 Air Flow box

By default, the air flow will determine its size automatically based on the input data. The air flow box lets you specify the box covered by the air flow field manually. Note, that the air flow box is only useful if your input data contains samples you want to ignore. Only consider using the air flow box if you cannot correct the original input data. To use the air flow box, just insert one and position it as required. Note that the air flow object will not update itself automatically. You need to trigger the update manually by right clicking on the air flow object in the scene inspector and choosing *Resample grid*.

### 12.3.7 Best practices

If you insert an air flow force field, you typically do not need an additional drag force. By default the **Air Flow** applies a drag force based on the imported velocities in the

regions where the imported velocities are given and additionally a drag force based on the user-defined **velocity** in all other regions. This means that adding an additional **Drag Force** is not needed and would double the applied force. If you do want to apply the force from an **Air Flow** or **Drag Force** only in a specific region you can either toggle the property **apply force everywhere** or, for air flows, you can use an **Air Flow box** (see Section 12.3.6).

## 12.4 Air Pressure

The **Air Pressure** object allows to simulate the effect of ambient pressure on a fluid. In the region defined by the object, a force acts which depends on the pressure difference of the air compared to the fluid. The force is achieved by filling the empty neighborhood region of a fluid particle with virtual air particles. These virtual air particles carry a user-defined pressure value, which is taken into account when computing the pressure gradient for the fluid particles. If **force type** is set to **AirForce**, the resulting force points into the fluid, whereas with **VacuumForce**, the force direction is inverted.

Table 63 shows the properties of an **Air Pressure** object.

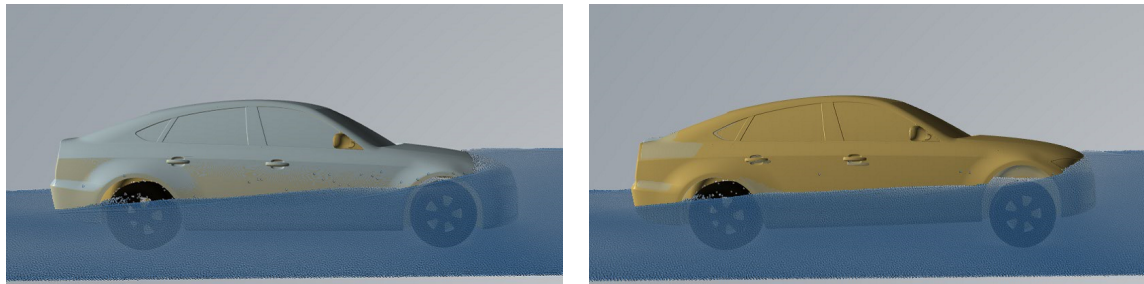
Property	Unit/Type	What it does
shape	-	Defines the shape in which the force field acts. Can either be <b>Box</b> or <b>Cylinder</b> .
air pressure	Pa	The pressure of the air in Pascal.
force type	-	The type of the force. Defines if the force acts into ( <b>AirForce</b> ) or out of the fluid ( <b>VacuumForce</b> ).
dist. based air pressure	-	If enabled, the air pressure can be keyframed depending on the distance from the position of the air pressure object.

**Table 63:** Properties of the **Air Pressure** object.

## 12.5 Car suspension model

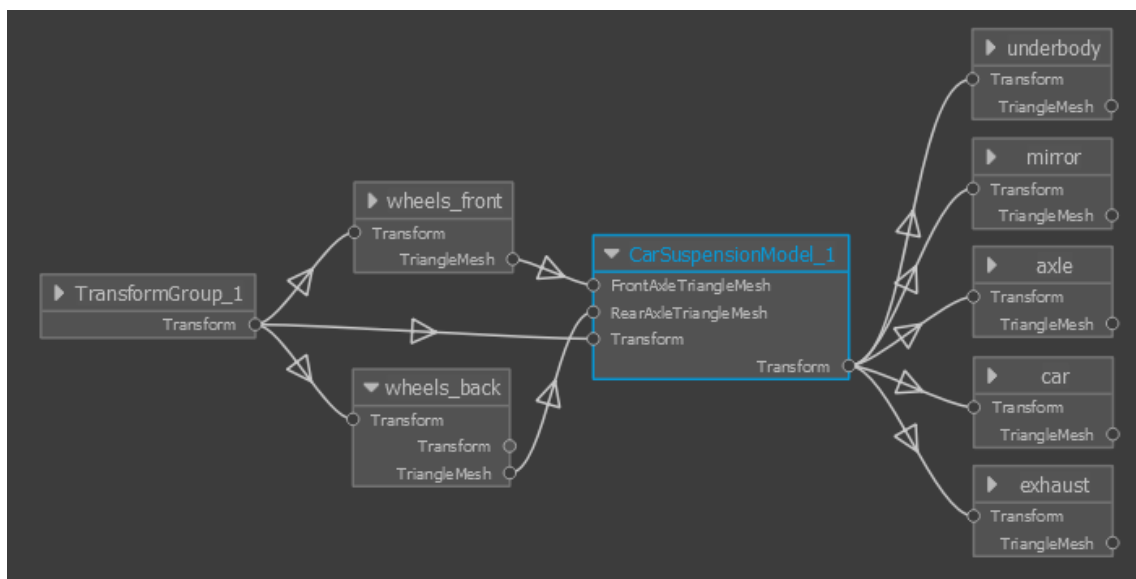
In wading simulations, the precise modeling of the car movement becomes increasingly important the faster the car moves and/or the higher the water level in the wading channel is. The position, orientation and velocity of the car when hitting the water pool determine the wave pattern in front of the car, height of the water splashes and the location of water, e.g., whether water flows across the engine hood or not (as illustrated in Figure 35). Therefore, engineers would have to provide a velocity profile across the channel drive-through as well as the exact positioning and orientation of the car. Considering the latter, the profiles of springs and dampers play an important

role, since they show how far the sprung mass of the car is pushed upward by forces exerted on the underbody of the car.



**Figure 35:** Comparison of a wading scenario computed without (left) and with car suspension model (right). For the simulation with car suspension model, there is no fluid on the front lid of the car, as the fluid forces acting on the car are pushing the sprung car parts upwards until the front damper is fully deflected.

As the profiles of springs and dampers are dependent on the interaction of the car with water, they are mostly not known beforehand. PreonLab provides a so-called half-car suspension model that computes the deflection of the springs based on the pressure forces exerted by the water and derives a re-positioning of the sprung car parts relative to the unsprung parts, i.e., the wheels and axles. Accordingly, the car suspension model can be considered as a special type of transform group which adds a transformation to the sprung geometry of the car. Note that the car suspension model might be part of a possible hierarchy, such as when the general movement of the car is keyframed or defined by a velocity profile as is illustrated in Figure 36. The car suspension model requires the connection of wheel or axle geometry to two



**Figure 36:** Example connection graph for scene with car suspension model. The **TransformGroup\_1** defines the overall transformation due to velocity and orientation keyframes of the car and the wheels. The **CarSuspensionModel\_1** applies forces acting from fluid onto the connected sprung car parts.

connection input slots in order to be able to automatically derive various properties of the car required for the suspension computation, e.g., the wheelbase of the car, see Table 64.

Property	What it does
<b>Transform</b>	The sprung car parts have to be connected via the <i>Transform</i> output slot. The general transform, i.e., the movement of the car through the wading channel, should be defined via a transform group that has to be connected to the <i>Transform</i> input slot.
<b>FrontAxleTriangleMesh</b>	The meshes belonging to the front axle, i.e, wheels and axle, have to be connected here. Use filter type <i>TriangleMesh</i> if you need to filter the connection graph for this type.
<b>RearAxleTriangleMesh</b>	The meshes belonging to the rear axle, i.e, wheels and axle, have to be connected here. Use filter type <i>TriangleMesh</i> if you need to filter the connection graph for this type.

**Table 64:** Connection slots of the car suspension model.

### 12.5.1 Half-car suspension model

The half-car suspension model splits the car into two parts, front and rear. By defining the **weight distribution** you can shift the center-of-mass along the wheelbase of the car. **weight distribution** and **center-of-mass height** together define the position of the center of mass. The wheelbase is automatically computed based on the front and rear axles meshes connected to the car suspension model.

Tables 65 to 68 list the properties of groups **General** and **Suspension** and respective subgroup properties.

Property	Unit/Type	What it does
<b>car body mass</b>	kg	The mass of all sprung parts of a car and possible additional weight, i.e, passengers or luggage.
<b>weight distribution</b>	-	Gives the front-to-rear weight distribution. The value has to be in the range of (0,1). For example, set 0.6 for a distribution of 60 : 40. This would move the center-of-mass more to the front.
<b>center of mass height</b>	m	Defines the height of the center of mass relative to the wheelbase, i.e. the straight line between front and rear axles. You have to subtract the wheel radius in case you have the center of mass height with respect to the floor.

**Table 65:** General properties.

Property	Unit/Type	What it does
parameter input	-	Specifies whether the suspension parameters are provided uniformly ( <b>Uniform</b> ) or per-axle ( <b>PerAxle</b> ).
nonlinear springs	On/Off	If enabled, the spring coefficient can be keyframed based on the deflection of the spring. Based on the <b>mapped unit</b> , the deflection is mapped to a spring rate or to a force, see Table 69.
reference point	-	If set to <b>axleLoad</b> , a spring deflection of zero corresponds to the spring configuration under axle load (given the gravity). If set to <b>uncompressedSpring</b> , the axle load will cause a deflection of the spring. In both cases, the geometric configuration of the car must reflect this initial state.
nonlinear dampers	On/Off	If enabled, the damper coefficient can be keyframed based on the velocity of the damper. Thus, a nonlinear relation between applied force and resulting damper velocity can be modeled.

**Table 66:** General suspension properties.

**Note:** If additional weight is added to the car, i.e., by adding passengers, you have to either provide the adapted geometry, or provide the relative translation and rotation of the sprung car geometry caused by this additional weight via a transform group which has to be connected to the *Transform* input slot of the sprung car geometry.

The **parameter input** property determines which subgroup of group **Suspension** will be shown. If set to **Uniform**, all springs in the car have identical properties, confirm Table 67. If **parameter input** is set to **PerAxle**, the springs for front and rear axle can be parametrized differently, confirm Table 68. Regardless the chosen **parameter input**, the values must refer to the suspension of one wheel.

Property	Unit/Type	What it does
max. spring compression	m	Defines the maximum possible compressive deflection of the springs relative to the static deflection
max. spring expansion	m	Defines the maximum possible expansive deflection of the springs relative to the static deflection.
spring constant	N/m	Defines the spring constant $k$ for the suspension and is only visible if <b>nonlinear springs</b> are disabled.

nonlinear spring coefficient	N/m or N	Defines the spring coefficient dependent on the spring's deflection via a deflection-to-coefficient mapping. Click on the icon to the right of the property to enter the keyframe editor and import this mapping. The icon is shown in red as long as the keyframe sequence is empty.
damper coefficient	N s/m	Defines the constant damper coefficient $c$ and is only visible if <b>nonlinear dampers</b> are disabled.
nonlinear damper coefficient	N/m or N	Defines the damper coefficient dependent on the dampers velocity via a velocity-to-coefficient mapping. Click on the icon to the right of the property to enter the keyframe editor and import this mapping. The icon is shown in red as long as the keyframe sequence is empty.

**Table 67:** Suspension properties if **parameter input** is set to **Uniform**. By this, the parameter values are applied to all springs uniformly.

Property	Unit/Type	What it does
max. spring compression (front)	m	Defines the maximum possible compressive deflection of the front axle springs relative to the static deflection.
max. spring expansion (front)	m	Defines the maximum possible expansive deflection of the front axle springs relative to the static deflection.
spring constant (front)	N/m	Defines the spring constant $k$ for the front left and right suspension and is only visible if <b>nonlinear springs</b> are disabled.
nonlinear spring coefficient (front)	N/m or N	Defines the spring coefficient dependent on the spring's deflection via a deflection-to-coefficient mapping for the front left and right suspension. Click on the icon to the right of the property to enter the keyframe editor and import this mapping. The icon is shown in red as long as the keyframe sequence is empty.
damper coefficient (front)	N s/m	Defines the constant damper coefficient $c$ for the front left and right suspension and is only visible if <b>nonlinear dampers</b> are disabled.
nonlinear damper coefficient (front)	N s/m	Defines the damper coefficient dependent on the dampers velocity via a velocity-to-coefficient mapping for the front left and right suspension. Click on the icon to the right of the property to enter the keyframe editor and import this mapping. The icon is shown in red as long as the keyframe sequence is empty.
spring constant (rear)	N/m	Defines the spring constant $k$ for the rear left and right suspension and is only visible if <b>nonlinear springs</b> are disabled.

<b>nonlinear spring coefficient (rear)</b>	N/m or N	Defines the spring coefficient dependent on the spring's deflection via a deflection-to-coefficient mapping for the rear left and right suspension. Click on the icon to the right of the property to enter the keyframe editor and import this mapping. The icon is shown in red as long as the keyframe sequence is empty.
<b>damper coefficient (rear)</b>	N s/m	Defines the constant damper coefficient $c$ for the rear left and right suspension and is only visible if <b>nonlinear dampers</b> are disabled.
<b>nonlinear damper coefficient (rear)</b>	N s/m	Defines the damper coefficient dependent on the dampers velocity via a velocity-to-coefficient mapping for the rear left and right suspension. Click on the icon to the right of the property to enter the keyframe editor and import this mapping. The icon is shown in red as long as the keyframe sequence is empty.

**Table 68:** Suspension properties if **parameter input** is set to **PerAxle**.

In case you enable **nonlinear springs**, the subgroup **Nonlinear Springs** is shown with properties described in Table 69.

Property	Unit/Type	What it does
<b>mapped unit</b>	-	Select <b>NewtonPerMeter</b> if the spring deflection is mapped to a spring rate (in N/m). Alternatively, select <b>Newton</b> if the spring deflection is mapped to a force (in N).

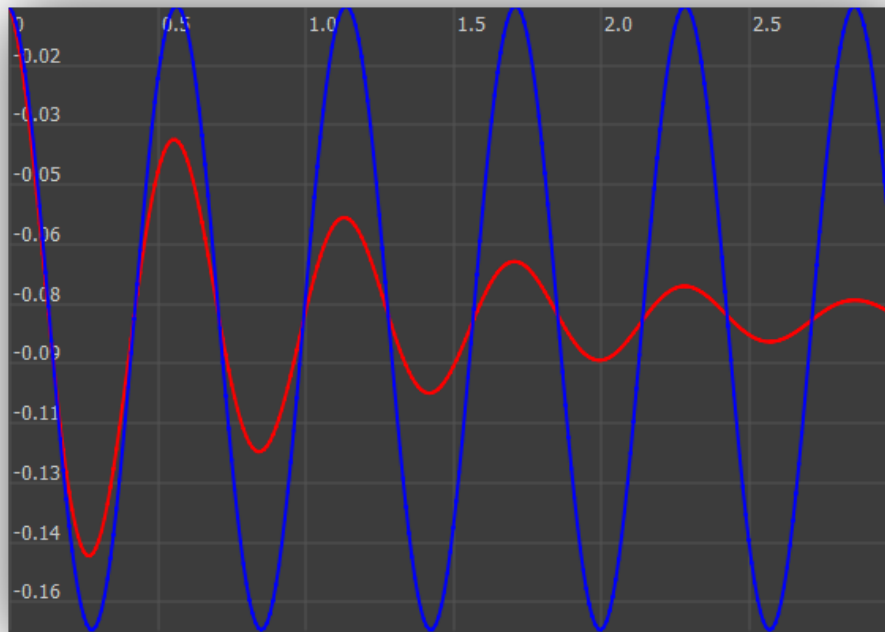
**Table 69:** Properties for nonlinear springs.

We have compared the implemented model to an analytical solution of a harmonic oscillator, once with the damping coefficient set to 0 and once to a value  $> 0$ . The graphs can be seen in Figure 37. It matches the analytical solution perfectly.

### 12.5.2 Best practices

Please consider the following preconditions and hints when setting up a wading scene that includes a Car Suspension Model (CSM):

1. The car has to be set up such that its wheelbase is parallel to one of the coordinate axes.
2. The car has to be set up such that the gravity direction is orthogonal to the wheelbase.



**Figure 37:** Gravity forces act on a car suspension model with uniform parameters at all four wheels. The graphs show the deflection of the sprung mass (in meter) for an undamped (blue) and a damped (red) configuration as a function of time (in seconds).

3. The maximum spring compression and maximum spring expansion is considered regardless of whether you provide a linear spring constant or nonlinear spring coefficients.
4. Connect all objects defining the sprung parts of the car (e.g. the car body) to the CSM by connecting the outgoing *Transform* slot of the CSM with the ingoing *Transform* slots of the sprung objects. Thus, they will be rearranged based on the deflections computed by the CSM.
5. Connect the (unsprung) parts of the front axle to the *FrontAxleTriangleMesh* slot and the (unsprung) parts of the rear axle to the *RearAxleTriangleMesh* slot. The CSM will automatically derive the wheelbase from the connected geometry.
6. Add a transform group (TG) to the scene and connect its outgoing *Transform* slot to the **Car suspension model** (and, thus, indirectly to the sprung parts of the car) and directly to the ingoing *Transform* slot of all unsprung car parts.
7. Animate the car movement by creating transform keyframes in TG.

The resulting connection graph is illustrated in Figure 36. Please consider reading the wading tutorial provided separately from this manual that assists you further in setting up your wading scenario.



## 13 Solid Objects

Solid objects can be added using the *Add* → *Solid*. There are different standard geometries implemented, e.g., cuboid, box, sphere. Arbitrary geometries can be imported either via *File* → *Import* → *Import Mesh* or per drag-and-drop.

PreonLab automatically samples the solids with particles in the resolution of the Preon solver (spacing). The sampled surface of the solid acts as an interface to the fluid, i.e., the sample points define a boundary condition which is included in the pressure system. The velocity of the solid particles matches the velocity of the solid at the corresponding position. The Preon solver computes inter-particle forces (adhesion and friction) between the solid interface and the fluid particles in proximity.

PreonLab can also be used to simulate rigid body dynamics including two-way coupling with a fluid. See Chapter 14 for more information.

**CAUTION:** Please make sure that the geometric center of the object is set correctly as this is mandatory for PreonLab in order to compute the desired and correct velocity of the solid particles. Therefore, please use *Coordinate system* and *Compute System* in the toolbar, see Section 13.8 for more information. In particular, this applies for setups with complex rotations such as gearbox and planetary gears.

Property	What it does
particle limit	Sets the maximum number of generated boundary particles to prevent allocation of too much RAM. The maximum is given in mega particles, so that a limit of 1 means 1 million particles. If this limit is exceeded, a warning will be printed to the message window. In this case, you either need to increase the fluid spacing or increase this limit to ensure a correct simulation.
dynamic sampling	Switches between static and dynamic sampling of the solid surface. If dynamic sampling is disabled, PreonLab will sample the entire surface of the solid with particles when starting the simulation (or performing other tasks that require boundary particles). If it is enabled, PreonLab will partition the solid into blocks and only sample the blocks if it is necessary, for instance if there is fluid nearby. In many cases, this greatly reduces the amount of solid particles, saving memory and potentially also performance (if the solid moves). However, enabling this property does not guarantee that dynamic sampling is actually used because some components of PreonLab do not support it yet. In these cases, PreonLab will automatically fall back to static sampling or emit a warning message.

<b>dynamic particle deletion</b>	Enables or disables dynamic particle deletion during the simulation based on boundary domains that are connected to this solid via the slot <i>DeletionDomain</i> . This is only required for domains that move in relation to the solid over time, which is in our experience very rare. Note that this option may cost a lot of performance, so use it with caution. If disabled, the boundary domains will only delete particles of this solid if there is no relative movement between the solid and the boundary domains over time. This option is currently incompatible with dynamic sampling and enabling it will enforce static sampling.
<b>sample triangle mesh</b>	If enabled, the particle sampling will operate on the mesh and not on the mathematical underlying shape (like cube or sphere). This property should be enabled when using dynamic sampling to get the best performance. The property does not exist if the solid surface is loaded from a mesh file.

**Table 70:** Properties of solid objects in group **General**.

Property	What it does
<b>rigid type</b>	Defines whether the solid acts as <b>scripted</b> (animated) object, or as <b>dynamic</b> in which case the physics of the object are computed, i.e., gravity acts on the solid, and it collides with other solid objects. The solid can also be defined as <b>stationary</b> . In this case, the solid does not move in world space, but you can pre-scribe a velocity for the object. The object will interact with the fluid according to this pre-scribed velocity. In this way, moving objects can be simulated without really moving them. The value <b>guided</b> is used in conjunction with the car suspension model, see Section 12.5.
<b>roughness</b>	Defines the roughness of the solid surface. If this parameter is 1 (default), the viscosity computed between the virtual fluid film and the fluid particles matches the viscosity of fluid-fluid interaction. By setting the roughness to larger values, surfaces with larger frictional effects can be modeled, e.g., a felt seal. The inter-particle force used to compute the force is defined by the viscosity model of the fluid solver (see Section 9.1).
<b>adhesion</b>	Controls the adhesion effect of the rigid onto the fluid. The employed model for computing the force is defined by the cohesion model of the fluid solver. When using the <b>PotentialForce</b> model, the adhesion value is a factor. The effective adhesion is computed by multiplying this factor with the cohesion of the fluid in contact. For the <b>PairwiseForce</b> model it is an absolute and independent value. For this model, a larger adhesion than cohesion value results in stronger forces between fluid and solid than the cohesion forces of fluid to fluid particles.
<b>density</b>	The density of the object. Together with the volume, this defines the mass.

**Table 71:** Properties of solid objects in group **Physics**.

## 13.1 Thermodynamics

By default, thermodynamics computation between solids and fluids is not performed. You have to activate it for each solid object individually by setting the **system type** to **ClosedFixed**, see Table 72 for more details. Solid objects can act as heat sources or sinks for fluids as described in Section 9.1.8. Please note that neither the diffusion of temperature inside solid volumes nor the thermal interaction of solids and air is currently computed.

Property	Unit/Type	What it does
system type	-	Defines whether the solid thermally interacts with the fluid. If set to <b>None</b> , it is not considered in the thermodynamics computation at all. In order to have the solid act as a heat source or sink, set the system type to <b>ClosedFixed</b> .
temperature	°C	Sets the temperature of the particles. Note that this property is only available if <b>system type</b> is set to <b>ClosedFixed</b> and there is no temperature field connected via the <i>TemperatureSamples</i> slot that provides a temperature distribution (see Section 17.8).
heat capacity	J/(°C kg)	The specific heat capacity of the substance on a per mass basis, i.e., the isobaric mass heat capacity. The default is 460.0 (cast iron at 25°C). Note that this property is only employed for rigid-rigid particle heat diffusion.
thermal conductivity	W/(m °C)	The property of a material to conduct heat. The default is 55.0 (cast iron at 25°C). Note that this property is only employed for rigid-rigid particle heat diffusion.

Table 72: Properties of solid objects in group **Physics** → **Thermodynamics**.

### 13.1.1 Known limitations

Heat diffusion within the solid is not performed. However, with **ClosedFixed** and keyframes you can utilize the rigid as a variable heat source (or heat sink) with respect to the fluid. Furthermore, the properties **heat capacity** and **thermal conductivity** are currently assumed to be constant and, thus, are no functions of temperature.

## 13.2 Film wetting

PreonLab introduces first steps towards the simulation of wetting films which are a sub-class of thin liquid films, i.e., fluid particles "stick" to the surfaces of solid objects.

These films are often thinner than can be depicted by the particle size actually employed for the simulation of the fluid dynamics. Therefore, we introduce the concept of *wetting film particles*. They reside on a solids' surface. The area they cover on the surface is in the resolution of the Preon solver (spacing). However, their height, i.e., their thickness of the wetting film they represent can be arbitrarily smaller than the Preon solver (spacing). It is derived from *wetting film particle's* mass. The mass of such a particle is zero at first, but increases as soon as it gets in contact with a fluid particle and a mass exchange is performed between those two particles.

Currently, this exchange is simply a function of time, but can be enhanced with arbitrary constraints in the future.

### 13.2.1 Parameters explained

Property	What it does
film max	Defines the maximum amount of wetting film per particle on a solid in relation to a fluid particle's mass. If set to 1, the wetting film could have a thickness of the size of the Preon solver (spacing).
film absorption rate	Defines the absorption rate per second of the wetting film in relation to a fluid particle's mass. If set to, e.g., 1, the mass of one fluid particle would be absorbed by the wetting film within 1 second.

Table 73: Properties of Solids in group Physics.

**Note:** If **film max** is set to a value greater than zero, the wetting film feature is automatically enabled while the default value of 0 disables film wetting computations for this solid object.

### 13.2.2 Visualizing the wetting film

Since the wetting film might be smaller than the radius of a fluid particle, we cannot visualize it similar to the fluid. Instead, we visualize its thickness using the coloring possibilities of a **solid**.

Property	What it does
coloring	WettingFilmBased
automatic range	Off
minimum range	0
maximum range	This value is discussed further below.
minimum color	Any color, e.g. light blue.

maximum color	Any color, e.g. dark blue.
enable mesh coloring	On

**Table 74:** Properties and values of **Solids** in group **Appearance**→**Coloring** for visualizing the wetting film.

## Best practices

Setting the **maximum range** of the **WettingFilmBased** coloring requires some thought. For the following discussion, we consider an area on the surface of size **PreonSolver**→**General**→**spacing**, i.e. 0.03. Since the film thickness is determined by **film max**, the film volume is

$$0.03 \cdot 0.03 \cdot (\text{film max} \cdot 0.03) = 0.03^3$$

assuming we have set **film max** to 1.0. Accordingly, with  $mass = volume \cdot density$ , we end up with a maximum possible mass of  $\approx 0.027\text{kg}$  per sample point as the fluid density is  $\approx 1000\text{kg/m}^3$ . Thus, we set **maximum range** to 0.027.

The **film absorption rate** determines how fast this maximum mass is reached for the wetting film. For example, if we set it to 0.1 and assume that the wetting film only absorbs fluid mass for one second, it is best to reduce **maximum range** to  $\approx 0.005$  in order to take advantage of the full coloring range.

### 13.2.3 Evaporation of film

We apply the same mathematical background to the wetting film as to the fluid particles for evaporation. The mass of the wetting film particle decreases over time until all its mass has evaporated. You have to connect an **Air** object to the solid for the evaporation of a wetting film (see Section 11.3).

## 13.3 Visualization of solid objects

Property	What it does
show particles	Enables / disables visualization of the boundary particles of the rigid. Note that only particles that are already created are visualized. When <b>dynamic sampling</b> is <b>On</b> , this depends on the proximity to fluid particles.
two-sided lighting	Enables / disables two-sided lighting for the rigid.

<b>Coloring</b>	The properties of this property subgroup control the coloring of the rigid boundary particles. They work just like the coloring for sensors explained in Section 16.2 by specifying three key colors and values. Coloring the solid requires the solid to be sampled with particles. Accordingly, this only works if a solver exists in the scene and either fluid is close to the solid or <b>dynamic sampling</b> is <b>off</b> . Note: Use coloring type <b>WettingFilm-Based</b> to visualize the wetting film of this solid (if you have defined the respective properties to allow for the evolution of a wetting film).
-----------------	--

**Table 75:** Properties of solid objects in group **Appearance**.

### 13.3.1 Random coloring for solids

You can colorize a set of rigids randomly by selecting them in the scene inspector and choosing the right-click action *Auto-colorize*. Note that you will get a different coloring each time you trigger the action, so if you don't like the colors it can be worth it to just try again.

## 13.4 Primitive shapes

PreonLab includes some common geometric shapes including **cuboid**, **sphere**, **capsule** and **cylinder**. Another simple object provided by PreonLab is the **Box** which is usually employed to quickly setup a basin for fluid. These objects have few special parameters - just scale them to get the shape you want.

Property	What it does
<b>segments</b>	Sets the number of horizontal and vertical segments for the sphere. Higher values will result in a smoother surface. A value between 18 and 50 is recommended.

**Table 76:** Properties for sphere.

Property	What it does
<b>top cap</b>	Disabling this property will remove the cap at the top of the cylinder.
<b>bottom cap</b>	Disabling this property will remove the cap at the bottom of the cylinder.

**Table 77:** Properties for cylinder.

## 13.5 Mesh

Using the mesh object, you can integrate arbitrary geometries into your simulation. You can add meshes either using *File* → *Import* → *Import Mesh* or by dragging a mesh file into PreonLab. You can import them with or without a mesh resource object (see Section 13.5.1), which allows you to treat multiple meshes stored in a single file separately.

The following file types for geometry meshes are supported by PreonLab:

*.dae* (Collada), *.blend* (Blender 3D), *.3ds* (3ds Max 3DS), *.ase* (3ds Max ASE), *.obj* (Wavefront Object), *.ifc* (Industry Foundation Classes (IFC/Step)), *.xgl* (XGL), *.zgl* (XGL), *.ply* (Stanford Polygon Library), *.dxf* (AutoCAD DXF), *.lwo* (LightWave), *.lws* (LightWave Scene), *.lxo* (Modo), *.stl* (Stereolithography), *.bstl* (Stereolithography binary), *.x* (DirectX X), *.ac* (AC3D), *.ms3d* (Milkshape 3D), *.cob* (TrueSpace), *.scn* (TrueSpace), *.assbin* (AssBin).

Property	What it does
path	The path to the mesh file. This is ignored if a Mesh resource is connected (see below).
particle sampling	In order to compute solid-fluid interactions, PreonLab needs to sample the mesh surface with particles. This property chooses the method that is used to generate the particles. The default is <b>Mesh_Independent</b> , which generates a sampling independent of the mesh structure. As an alternative, there exists the <b>Mesh_Thinned</b> method, which samples the given mesh directly and thins oversampled regions if necessary. This method is faster but can (in rare cases) lead to fluid leakage issues if the mesh contains many small triangles in comparison to the fluid spacing.

Table 78: Properties for Mesh.

### 13.5.1 Mesh resource

A Mesh resource object loads a mesh file from disk and provides an output slot in the connection editor for each contained submesh (or one slot for the whole mesh, if you have selected "Single mesh" in the import dialog). These slots can be connected to mesh rigids which will then ignore their mesh file (if specified) and instead use the mesh provided by the connected Mesh resource object. There are multiple use cases in which this is useful:

1. Some mesh files contain multiple submeshes and it may be necessary to represent these submeshes as individual rigids, for instance in order to keyframe them separately. This can not be achieved using plain Mesh rigids, but it is possible using a single Mesh resource connected to the individual Mesh rigids.
2. Mesh resource objects allow sharing the same mesh between multiple rigids. This avoids loading the same mesh from disk multiple times and saves memory.

3. Mesh resource objects can be rescaled comfortably according to a specified unit in which the mesh was modeled in.

If you import a mesh file into PreonLab, you can choose if you want to create a mesh resource and if you want to expose all submeshes separately. During the import, a Mesh object is created for each exposed submesh.

Property	What it does
mesh file	The path to the mesh file.
units	Rescales the mesh if anything different than <i>Meter</i> is specified. Choose the unit that was used to model the mesh in order to rescale it for PreonLab.

Table 79: Properties for Mesh resource.

## 13.6 Alembic Mesh

An Alembic mesh represents a single object from an Alembic file. It allows to use a (deforming) mesh and its animation from an Alembic file in PreonLab. More information on the import of an Alembic file can be found in Section 17.5. Please note, that beginning with version 3.2, PreonLab only supports Alembic files using the Ogawa data format. More information can be found in Section 17.5.1. Table 80 lists the properties of an **Alembic Mesh**.

Property	What it does
path	The path to the Alembic file.
load mesh samples	Specifies if the mesh samples should be loaded from the Alembic file. This allows to use deforming meshes in PreonLab. If this property is set to <b>off</b> , only the first mesh sample is loaded from the file. This property is automatically set to <b>off</b> or <b>on</b> when creating an Alembic object based on the number of mesh samples found in the Alembic file. Note, that only Alembic objects with this property set to <b>off</b> can be used in an MPI simulation.
load transformation samples	Specifies if the transformation for this object should be loaded from the Alembic file. If you want to keyframe this object, you must either set this to off, or use a Transform group parent object.
alembic object path	The path of the object inside the Alembic file. As a single Alembic file can contain more than one object, the exact path to a single object inside the file must be specified.
flip triangle orientation	Some meshes have their triangles orientated in the wrong way which leads to wrong lighting. Toggle this property to change the triangle orientation.

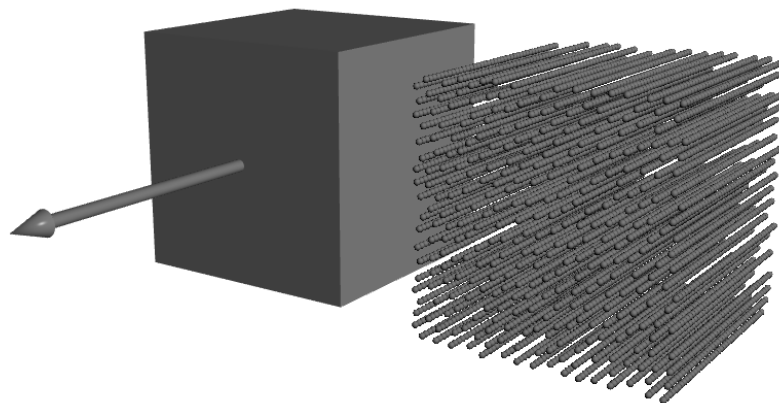


<b>animation time offset</b>	This property allows to specify a time offset (in seconds) for the animation loaded from the Alembic file.
<b>animation time factor</b>	The animation time factor can be used to speed up or slow down the animation loaded from the Alembic file. A factor smaller than 1 slows the animation down while a factor of larger than 1 speeds the animation up.

**Table 80:** Properties for Alembic Mesh.

## 13.7 Porous Rigid

The porous rigid can be used to mimic objects with a certain sub-scale porosity, i.e, porosity due to small channels or holes that cannot be captured by the spacing of the Preon solver. This object is similar to the **Cuboid** solid, but with the additional property **porosity**. The porosity is modeled by directed (void-space) channels which is indicated by the arrows. Note that for a **Porous Rigid** object the default value for both, **roughness** and **adhesion**, is 0. This way the porous media models a pressure gradient which automatically results from the resistance induced by the solid samples points taken into account in the pressure computation by the Preon solver.



**Figure 38:** Porous rigid with indicated direction of porosity/channels (left). A porosity of 0.8 visualized by setting **Appearance**→**show particles** to **On** (right).

Property	What it does
<b>porosity</b>	Defines the porosity of the object. A porosity of 1, is totally porous, while 0 means no porosity.

**Table 81:** Properties for porous rigid.

### 13.7.1 Best practice

The porous rigid allows fluid particles to pass a geometry that would otherwise be impenetrable due to the chosen fluid particle spacing. The dimensionless porosity has to be calibrated manually. A desired flow rate can be chosen as frame of reference. We suggest to start of with a high porosity and then decrease it until the desired flow rate is reached. If no porosity value, i.e., channel sampling, can be found that matches the desired flow rate choose one that overpredicts it and fine-tune with the **roughness** property of the porous rigid.

## 13.8 Changing the pivot / center-of-mass

By default, PreonLab assumes that the center-of-mass of a rigid is located at the origin. If this is not the case, it may lead to the following problems:

- Keyframing an object so that it rotates without moving is close to impossible.
- Simulated dynamic rigids may interact incorrectly with other rigids.

Both problems can be solved by adjusting the pivot position (also called center-of-mass). To change the pivot, select the object, click on the *Coordinate system* button in the toolbar and select the *Translate pivot* dragger. Moving the dragger changes the pivot position and also adjusts the object position so that the entire object does not move. If it is necessary to specify an exact pivot position, it is also possible to specify the pivot using the property editor. However, this position is only intuitive if no rotations are applied to the object. It is highly recommended to set the pivot before rotating it or attaching it to a transform group. If possible, it is recommended to use the dragger which should always work as expected. In any case, the dragger should be used to check if the pivot is located at the desired position.

### 13.8.1 Rotating around a custom axis

Even after adjusting the pivot position, it is still very difficult to rotate objects around a specific non-unit axis. In order to solve this problem, it is necessary to specify a local coordinate system in which the rotation is applied.

If you know exactly around which axis you want to rotate, you can specify the revolution around a given axis by changing the **orientation control mode**. See Section 4.3 for more information on this option.

It is also possible to compute the local coordinate system automatically by clicking on *Compute System* in the toolbar. This will also set the pivot position. However, it only works correctly for symmetric objects.

### 13.8.2 Solid velocities for meshes with pivot

Depending on the time step of the simulation, meshes that are children of a transform group and rotate around this transform group but have a position that lies outside the rotation center may compute wrong velocities for their solid particles. To prevent this, it is recommended to use the *Coordinate system*→*Compute System* option in the tool bar for these meshes to move their center-of-mass to the rotation center.

## 14 Rigid body simulation

PreonLab can simulate rigid body dynamics, i.e., dynamic rigid body objects that collide with each other. Additionally, two-way coupling between rigid body objects and fluids can be simulated.

PreonLab includes a particle-based rigid body solver which is used by default. This solver is especially well suited for the simulation of concave and complex geometries. More information regarding this particle-based solver can be found in Section 14.3. Alternatively, PreonLab additionally includes the open-source library Bullet for rigid body simulation.<sup>1</sup> More details on settings regarding Bullet can be found in Section 14.2.

You can switch between the rigid body solvers by clicking on *Settings*→*Rigid body settings* and selecting the respective solver in the drop-down on the top.

By default, all rigid body objects have their property **Physics**→**rigid type** set to **scripted** which means that they are not simulated as a rigid body but just stay at a fixed position or move based on transformation keyframes. However, simulated rigid bodies can still collide with scripted objects. When setting **Physics**→**rigid type** to **dynamic**, the object is simulated as rigid body, i.e., gravity acts on the solid, and it collides with other solid objects. For dynamic objects, their center-of-mass is relevant for the simulation, see Section 14.1 for more information.

In Table 82, general properties of a rigid body object are shown which are also relevant for scripted objects. In Table 83, object properties are shown that are only relevant for dynamic rigid bodies and which are independent of the used rigid body solver. More object-specific rigid body settings that are dependent on the rigid body solver are shown in the respective sections Sections 14.2 and 14.3.

Property	What it does
<b>rigid friction</b>	The friction coefficient of the rigid relevant for interactions with other rigid objects. The friction values of two interacting rigid objects are multiplied to get the value that is used to compute the frictional forces between them.
<b>linear velocity</b>	The linear velocity of the object. For scripted objects using keyframes, this value is automatically computed. For dynamic objects, the user-defined linear velocity is the initial linear velocity of the object. For stationary objects, the user-defined value is used as the linear velocity of the non-moving solid.

---

<sup>1</sup>[Bullet Physics Library](#)

<b>angular velocity</b>	The angular velocity of the object. For scripted objects using keyframes, this value is automatically computed. For dynamic objects, the user-defined angular velocity is the initial angular velocity of the object. For stationary objects, the user-defined value is used as the angular velocity of the non-moving solid.
<b>ignore in rigid body simulation</b>	If enabled, the solid will not be considered when simulating rigid body dynamics, no matter what rigid type or behavior state it has. It will still be considered by fluid though if its behavior is not inactive.

**Table 82:** General rigid body settings in the group **Rigid Body Settings**.

Property	What it does
<b>two-way coupling</b>	Defines whether the object is influenced by fluids.
<b>volume</b>	Shows the current volume of the object.
<b>mass</b>	Shows the current mass of the object. This is a read-only property which depends on the <b>density</b> and <b>volume</b> of the object.
<b>use custom inertia tensor</b>	If enabled, a custom inertia tensor can be defined using the <b>custom inertia tensor</b> property.
<b>inertia tensor</b>	The unrotated inertia tensor of the object. Note that this is only correctly calculated when starting the simulation. This property is only shown if <b>use custom inertia tensor</b> is disabled.
<b>custom inertia tensor</b>	The unrotated inertia tensor of the object. Note that PreonLab does not ensure that the values of <b>mass</b> and <b>custom inertia tensor</b> have a correct relation to each other. Accordingly, if you adapt <b>custom inertia tensor</b> , you probably also need to adapt <b>manual volume</b> and <b>density</b> of the rigid object to get the correct corresponding <b>mass</b> . This property is only shown if <b>use custom inertia tensor</b> is enabled.
<b>Constrain DOF</b>	These properties allow you to restrict forces applied to the rigid body object. You can freeze movement along the three unit axis and you can freeze rotation around unit axis.
<b>use manual volume</b>	Only shown if the solid is of type Mesh. If true, the user can manually specify a volume using property <b>manual volume</b> instead of using the automatically calculated one.
<b>manual volume</b>	Only shown if the solid is of type Mesh. Allows to specify the volume of the object. The specified volume is for the current scale of the object and is automatically adapted whenever the scale is changed.

**Table 83:** Properties of rigid body objects in group **Dynamic Settings** which are independent of the used rigid body solver (only relevant for dynamic rigids).

## 14.1 Center-of-mass

The center-of-mass of a dynamic rigid body object influences its behavior in a simulation. The center-of-mass of an object is defined by its pivot. See Section 13.8 for more information.

## 14.2 Bullet

Bullet is an open-source physics library which can be used in PreonLab to simulate rigid body dynamics. Bullet is especially well suited to simulate a large amount of convex geometries. Note, that by default, Bullet computes a convex hull for objects in order to do a fast collision detection. This may be problematic for concave meshes. See the property **collision shape** explained in Table 84.

Table 84 shows the basic properties of rigid objects related to the rigid body simulation. General settings related to the Bullet simulation can be accessed over the menu entry *Settings*→*Rigid body settings*. The properties of the Bullet solver are described in Table 85.

Property	What it does
<b>bounciness</b>	Physical bounciness of the object. The resulting bounce is dependent on the bounciness of both colliding rigids.
<b>rolling friction</b>	Defines the rolling friction of a rigid body object. This is a factor for the normal <b>rigid friction</b> . The final rolling friction value between two objects is computed by first multiplying each rolling friction value with the respective <b>rigid friction</b> value and the adding these two values of both interacting rigid objects.
<b>collision margin</b>	Defines the collision margin added to the collision shape. Note that different collision shape handle the collision margin differently.
<b>collision shape</b>	This property allows to choose the type of the internal collision shape that is used to compute interactions between rigids. This property is only relevant if the object is a dynamic rigid or interacts with other dynamic rigids. It has no effect on the interaction with fluid. <b>meshConvexHull</b> uses the convex hull of the rigid. <b>meshConvexHullReduced</b> uses the simplified convex hull, which usually improves stability and performance compared to <b>meshConvexHull</b> . <b>obb</b> uses a simple oriented bounding box as collision shape, which gives good stability and performance, but also poor accuracy. <b>bvhTriangleMesh</b> and <b>glm-pactMesh</b> are the most accurate shapes, but they are usually quite slow and tend to have stability issues. <b>bvhTriangleMesh</b> only works for static objects.

<b>external collision shape</b>	Setting this property to on allows to specify a mesh (by setting <b>collision shape mesh</b> ) that is used as base for creating the collision mesh instead of the original mesh. Furthermore, if the mesh provided by <b>collision shape mesh</b> contains submeshes, these submeshes are separately converted to collision shapes and are then combined. This allows to create concave collision meshes built out of multiple convex collision meshes. The algorithm used to create collision shapes out of the submeshes is the same as provided by <b>collision shape</b> .
<b>collision shape mesh</b>	The path to the external collision shape mesh. See <b>external collision shape</b> for more details.
<b>show collision shape</b>	Enables / disables visualization of the internal collision shape.

**Table 84:** Properties of solid objects which are only relevant when using Bullet as rigid body solver. These properties are located in the **Bullet Settings** property group.

Property	What it does
<b>substep size</b>	This is the time step size of a single sub step done by the Bullet solver. The size of this time step should always be chosen smaller than a single simulation time step done by PreonLab. Furthermore, this value times <b>nr sub steps</b> should be larger than a single simulation time step.
<b>nr sub steps</b>	This is the maximum number of sub steps the Bullet solver does. This number times the <b>substep size</b> should always be larger than a single simulation time step by PreonLab or otherwise the rigid body simulation may be incorrect.
<b>split impulse</b>	If on, the Bullet solver solves positional and velocity constraints separately. Contact handling is usually more accurate when enabled.
<b>split impulse threshold</b>	Only relevant when <b>split impulse</b> is on. When enabled, the split impulse position correction is only used when the penetration is larger than this value, otherwise the regular velocity/position constraint coupling is used. This value is normally negative since it represents the penetration of two objects.
<b>solver iterations</b>	Maximum number of iterations the Bullet constraint solver does.

**Table 85:** Bullet solver settings.

### 14.2.1 Collision margin

When simulating scenes with dynamic rigids and the Bullet rigid body solver, you may notice a small gap between rigid objects. The reason for this gap is Bullet's internal collision margin. PreonLab automatically chooses the collision margin based on the fluid spacing. This should ensure that no fluid can pass through these gaps. If the gap between rigids is disturbing, you can reduce it by using a smaller fluid spacing.

## 14.3 Particle-based rigid body solver

The particle-based rigid body solver was introduced in PreonLab 3.1. It computes rigid body dynamics and resolves rigid-rigid contacts based on the rigid particles that are already used by the fluid solver. It is therefore especially well suited for complex and concave geometry since it does not use a concave approximation of the collision shape as done by Bullet.

Settings of the particle-based rigid body solver can be accessed using the *Settings*→*Rigid body settings* dialog. The properties of the solver are explained in Table 86.

Property	What it does
CFL	The CFL number is used to compute the maximum time step based on the maximum velocity of a rigid particle and the spacing. CFL numbers larger than 1 may not result in accurate simulations.
relaxation factor	This is the relaxation factor used by the relaxed Jacobi solver to solve the system of equations. This influences the convergence. Larger values than 0.5 may not converge while lower values than 0.5 may converge more slowly, i.e. need more iterations.
max. iterations	The maximum number of iterations the iterative solver does.
min. iterations	The minimum number of iterations the iterative solver does.
max. avg. error	The solver iterates until all rigid particles have at most this average error or until the maximum number of iterations is reached.
adaptive time step	If on, the time step is adaptively estimated based on the CFL condition. If off, the time step specified as <b>max. time step</b> is used.
max. time step	The maximum time step used by the rigid body solver. When there is a fluid solver in the scene, the minimum of the time step of both is used. However, if there is no dynamic rigid body in the scene, this maximum time step of the rigid body solver is not taken into account.

Table 86: Settings of the particle-based rigid body solver.

### 14.3.1 Collision margin

When simulating scenes with dynamic rigids and the particle-based rigid body solver, you may notice a small gap between rigid objects. This is due to the size of the rigid particles and them being sampled on the surface of the rigid mesh. Accordingly, this gap is reduced when reducing the fluid spacing.



## 15 Rendering

PreonLab does not include a video encoder, so it can not directly generate videos. However, you can write the individual video frames as pictures to disk and create a video from these frames using a third party tool like FFmpeg (see Section 17.12).

PreonLab contains two renderers that can output video frames. The first one is the same that displays the scene in PreonLab's graphical user interface. It is based on OpenGL and is optimized for realtime rendering so that the user can interact with the scene. You can output OpenGL frames by simply enabling GL recoding as described in section 8.2 and clicking on playback. The second option is **Preon renderer**, a custom raytracer capable of high-quality fluid rendering.

### 15.1 Cameras

By default, there are three cameras with orthographic projection and one camera with perspective projection in every scene. The cameras with orthographic projection let you see the scene from  $x$ ,  $y$  and  $z$  direction. You can change the active camera either by clicking on the button in the top right corner of the graphics window or via the context-menu of the respective camera. Therefore, right-click on the camera in the *Scene Inspector* and select *Set as active camera*. Additional cameras can be added using the *Add* menu. You can also animate cameras like other objects using keyframing. You can read more about it in Section 6.2.1.

The camera position, orientation and look-at can be manipulated via the mouse controls listed in Table 1. Note that the camera manipulations via mouse control can not be undone using the **Undo** action. The best practice to not have your camera transformation accidentally changed is to set the property **locked** to on.

Property	What it does
field of view	The vertical field of view of the camera specified by an angle in degrees. Only relevant for perspective projection.
near clip	The distance between the camera and the near clipping plane. Objects in front of the near clipping plane will be clipped and are therefore invisible. You may want to decrease this value when working with very small objects. However, picking a small value can cause flickering artifacts in large scenes, because it decreases the precision of the internal depth buffer.

<b>far clip</b>	The distance between the camera and the far clipping plane. Everything behind the far clipping plane will be clipped. Changing the far clipping plane has similar tradeoffs like changing the near clipping plane.
<b>focus distance</b>	The distance from the camera position to the camera pivot (only relevant when you rotate the camera using the mouse).
<b>orthographic</b>	Toggles between orthographic and perspective projection.
<b>fixed up axis</b>	Defines whether the up axis of the camera is fixed. Fixing the up axis is recommended for more stable camera control.
<b>locked</b>	If on, the camera can not be manipulated per mouse. This property can also be changed by right-clicking the camera in the <i>Scene Inspector</i> and clicking on <i>Lock control</i> .

Table 87: Properties for **Camera**.

Property	What it does
<b>inverse direction</b>	Inverses the direction of the camera and changes the camera position by mirroring it on the projection plane.
<b>viewport width</b>	Sets the half width of the cubic volume viewed by the orthographic camera.
<b>fixed grid</b>	If on, the scene grid will rotate with the camera and act as a background.

Table 88: Properties in group **orthographic projection**.

## 15.2 Clipping object

The clipping object can be used to disclose areas that would be otherwise hidden by other objects. The clipping object can be either set to an infinite plane which clips all objects behind or in front of it, or it can be set to a finite box-shaped area which clips objects inside or outside the area. You can position and orient the clipping objects like any other object using either one of the transform dragger or the property editor. Using the connection system, you can control what is clipped as only objects connected via the *Clip* slot are accounted for by the clipping object. Likewise, the clipping object is only active for cameras connected to it via the *Clip* slot. Note that by default PreonLab connects all objects and cameras with the clipping object. Please also note that currently only up to five active clipping objects are supported.

Property	What it does
<b>shape type</b>	Defines the shape of the clipping object which can be either an infinite <i>Plane</i> or a finite <i>Box</i> .

<b>flipped</b>	Turning this flag on, inverses the clipping area. For a plane this has the same effect like rotating the plane around 180 degrees. For a box-shaped clipping object the area outside of the box is clipped. Default setting is flipped off which clips the inside area defined by the box.
----------------	--

**Table 89:** Properties for a Clipping object.

## 15.3 Lights

PreonLab supports directional and point lights. By default, each scene contains a directional light.

Property	What it does
<b>color</b>	Sets the color of the light, located in the Appearance group.
<b>intensity</b>	Scales the intensity of the light.
<b>casts shadows</b>	Enables or disables casting of shadows. In OpenGL, this might have no effect because only one shadow casting directional light is supported and only solids without two-sided lighting may receive shadows. Preon renderer supports shadows for an arbitrary number of lights.

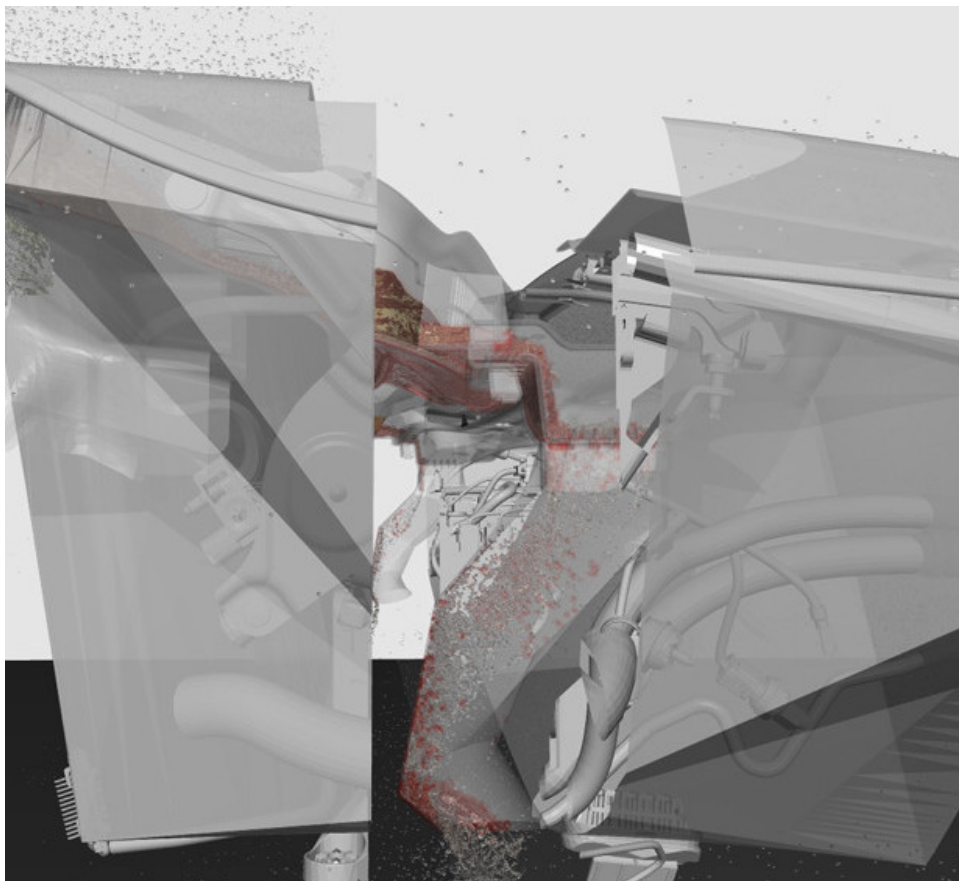
**Table 90:** Common light properties.

### 15.3.1 Directional light

A directional light emits light from a single given direction and illuminates the whole scene. The arrow of the directional light indicates its direction. The direction can be changed by rotating the directional light using the property editor or the rotation dragger.

Property	What it does
<b>photon mapping</b>	Enables or disables photon mapping for this light when using photon mapping with Preon renderer. Has no effect for OpenGL rendering and for Preon renderer objects that have disabled photon mapping.
<b>shadow softness</b>	Sets the softness of the shadow. Only relevant for materials with enabled monte-carlo light transport and only relevant for Preon renderer. Should be in the range between 0 and 0.2.

**Table 91:** Directional light properties.



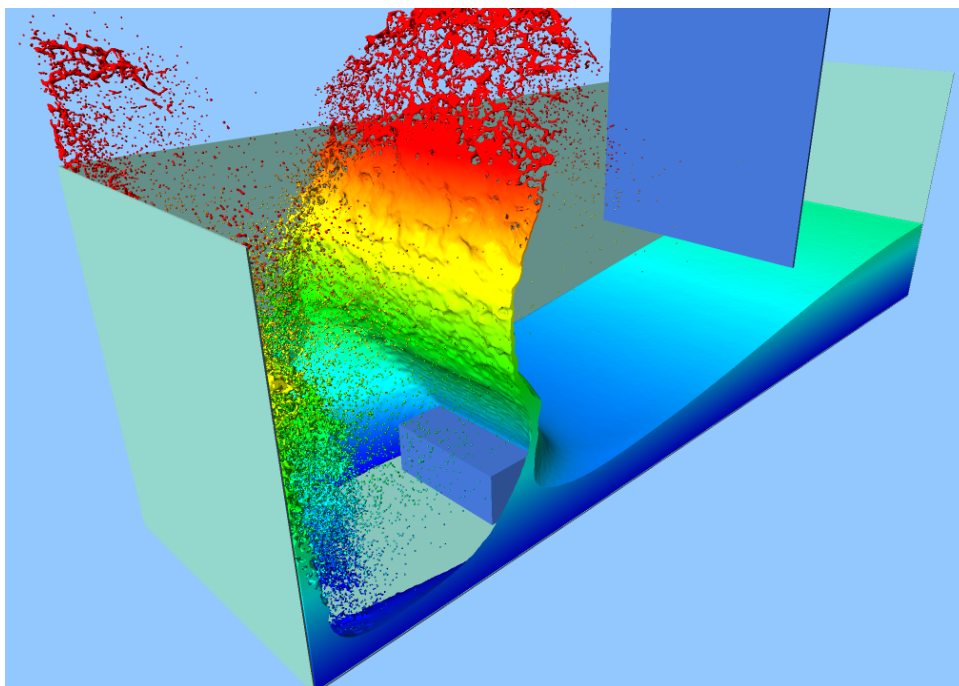
**Figure 39:** *Box-shaped clipping object* clips some parts of the engine compartment. Water is not clipped. Wet surfaces are highlighted in red by wetting sensor. Image is rendered with Preon renderer.

### 15.3.2 Point light

A point light emits light from its position in all directions. Point lights have no special properties.

## 15.4 Preon renderer

**Preon renderer** does not visualize fluid particles as small spheres, but instead renders a smooth surface. This means that when using Preon renderer, there is no need for meshing using **Preon mesher**. Also note that OpenGL rendering is not available via the command-line version of PreonLab, so Preon renderer is the only choice there. To get started, insert a renderer object (located in the group *Cameras*). You can also just click on the *Rendering* button in the toolbar, which will create a new renderer object automatically if necessary.



**Figure 40:** Rendering for a breaking dam scenario colored by fluid height.

### 15.4.1 The rendering dialog

The simplest way to render images is using the rendering dialog. The dialog can be opened by clicking on the *Rendering* button in the toolbar. If a renderer object is selected, the dialog will be opened for the selected renderer. If no renderer is present in the scene, a new one will be automatically created. In the dialog, you can select a camera and render an image for it by clicking on *Render frame*. You can render a sequence of images by clicking on *Render sequence*. It is also possible to render a sequence of frames for multiple cameras at once. Rendered images will always be saved directly to disk. If you jump in the timeline or close and reopen the dialog, the dialog will always try to load a previously rendered image for the current frame from disk. Click on *Open image folder* to open the folder where images are stored for the current camera.

The size of the rendering dialog can be adjusted and scrollbars will appear if necessary. Click on *Fit size* to rescale the dialog so that the rendered image is shown without scrollbars.

### 15.4.2 Rendering during simulation

Sometimes, it is beneficial to render automatically during simulation, e.g., when simulating on a cluster or computer without graphic card. To enable automatic rendering, make sure that the behavior of the renderer is set to *active*. Furthermore, you need to connect the cameras for which you want to render frames to the renderer via the *Camera* slot. The rendered frames will be located in the scene directory in the subfolder:

Visualization/[NameOfRenderer]/[NameOfCamera].

### 15.4.3 Parameters

Property	What it does
<b>transparent background</b>	If enabled, the background will be transparent. Do not use this if you intend to generate a video from the rendered images later.
<b>background color</b>	Specifies the background color.
<b>background color 2</b>	Specifies a second background color. If this color differs from the first background color, a color gradient between the two will be interpolated across the background hemisphere.
<b>background dithering</b>	Specifies the amount of dithering applied to the background. Dithering prevents color banding. The value should be between 0 and 5.
<b>max. solid recursion depth</b>	Sets the maximum recursion depth for solid ray tracing. You may need to increase this in scenes with many transparent layers.
<b>max. reflection bounces</b>	Sets the maximum number of times a ray may be reflected from a solid surface. Only relevant for reflective materials.
<b>max. fluid recursion depth</b>	Sets the maximum recursion depth for fluid ray tracing.
<b>artificial secondary hemisphere</b>	If enabled, an artificial hemisphere will be used for secondary reflections. This will improve the visual quality for scenes with monoton or dark backgrounds.
<b>show color legends</b>	If set to true, color legends for fluids and sensors are included in the rendered images.
<b>show time</b>	If set to true, the elapsed time will be drawn in the rendered images.
<b>samples per fragment</b>	Sets the number of stochastic samples per pixel fragment for materials with enabled monte-carlo raytracing. More samples reduce noise but also require more time to compute.

Table 92: Properties in group **Rendering**.

Property	What it does
<b>surface smoothing radius factor</b>	Controls the smoothness of the fluid surface. A value between 2 and 4 is recommended. Higher values mean a smoother surface, but take longer to compute.
<b>iso surface scale</b>	Values above 1 will thicken the fluid, while values below 1 will shrink it. This value should always be below 1.5 or rendering artifacts may appear.

<b>min culling neighbor count</b>	Sets the neighbor count threshold above particles inside the fluid volume may be selected for efficient culling. In normal cases it should never be necessary to change this, but you may try to increase it when experiencing holes in the rendered fluid surface.
<b>fluids cast shadows</b>	Enables or disables translucent shadows casted by fluids. Disabling this saves performance. Only relevant if there are lights in the scene that cast shadows.
<b>sdf method</b>	(Experimental) The method used for computing the signed distance field that defines the fluid volume and surface. To render adaptive fluids, <b>Density_Adaptive</b> should be chosen.

Table 93: Properties in group **Fluid rendering**.

Property	What it does
<b>width</b>	Specifies the width of the rendered image in pixels.
<b>height</b>	Specifies the height of the rendered image in pixels.
<b>anti-aliasing</b>	Specifies whether anti-aliasing is used. Improves image quality, but takes longer to compute.

Table 94: Properties in group **Resolution**.

## Photon mapping

Preon renderer supports the rendering of caustics using a technique called photon mapping. This is a computationally demanding process, but it can also greatly increase the realism of the visualization. It is recommended to try photon mapping for a single frame and then decide whether it is worth the effort or not. To enable photon mapping, check the **enable photon mapping** property in Preon renderer. You can also enable or disable photon mapping for individual light sources, but in any case the property in Preon renderer must be checked. Note that currently, only directional lights support photon mapping.

A crucial parameter for photon mapping is the **photon cell size** property in Preon renderer. It determines the size of individual photons and greatly influences quality and performance. A good value for the cell size depends on the scale of the scene. If photon mapping takes very long to compute, then you may increase the cell size to improve performance. If the caustics are too blurry then you need to decrease the photon cell size. PreonLab will warn you if the cell size leads to a number of photons that exceeds the limit specified in **max number of photons**. In this case you should increase the cell size.

Property	What it does
<b>max number of photons</b>	Specifies the maximum number of photons. If this is exceeded, the photon spacing will be decreased automatically.

<b>photon spacing</b>	Specifies the spacing between individual photon samples. Higher values will result in less photons and better performance, while lower values will result in more simulated photons and sharper caustics.
<b>photon mapping</b>	Enables or disables photon mapping.

Table 95: Properties in group **Photon mapping**.

## 15.5 Materials

Materials determine the appearance of solids and fluids. By default, every object is rendered using the *DefaultMaterial* which is present in every scene. You can insert and modify new materials just like other objects. To assign a material to a solid or fluid, connect the material to it via the *Material* slot. Thereby, it may be necessary to remove the old material connection first.

Note that starting with PreonLab 3.1, materials can be added and assigned with just one click via the context menu. For this, right-click the selected object(s) either in the scene inspector or in the graphics window. In the context menu, expand submenu *Set material* and either generate a new material or assign an existing one. The material which is currently assigned is emphasized with bold letters.

### 15.5.1 Shared material parameters

The following parameters exist for all materials:

Property	What it does
<b>override color</b>	If enabled, the color of the connected object is overridden by the <b>color</b> property.
<b>color</b>	The material color, only matters if <b>override color</b> is enabled.
<b>normal noise amplitude</b>	Sets the maximum amplitude for surface normal noise generation. Currently, noisy surface normals are only supported for fluids and not for solid objects.
<b>normal noise frequency</b>	Sets the lowest frequency for surface normal noise generation.

Table 96: Properties of Volumetric material in group **Material properties**.



## 15.5.2 Surface material

The Surface material is based on the Phong illumination model. It consists of an ambient, a diffuse and a specular term. The diffuse term models diffuse reflections of the material, which are independent from the viewer direction. In contrast, specular reflections are dependent from the viewer direction. Finally, the constant ambient term models indirect lighting not captured by the diffuse and specular term. In most cases, it is not recommended to tweak the parameters of a Surface material manually. Instead, one of the available presets should be used which are optimized for quality and performance (see section Section 15.6).

Property	What it does
flat shading	Enables or disables flat shading, which determines how surface normals are computed.
fade factor	This factor is multiplied to the overall opacity of the material, useful for realizing fade-in and fade-out effects.
ambient factor	Scales the ambient term of the Phong illumination system (should be between 0 and 1).
glass reflection	Enables glass reflections, which adjusts the specular to diffuse ratio based on the incident viewing angle. This property does not influence the opacity of the material, which can be adjusted separately. As an example, enabling this property and setting the material <b>color</b> to (255, 0, 0, 125) realizes a material for red tinted glass. Please note that the Surface material can only model glass reflections properly and ignores refraction and absorption effects. These effects are only supported by the Volumetric material, which can only be applied to fluids or meshes that form a closed volume.
specular to diffuse ratio	Sets the ratio between specular and diffuse reflection, must be between 0 and 1. Only available if <b>glass reflection</b> is disabled.
specular tint	Sets the specular tint factor, must be between 0 and 1. This regulates how the material color contributes to specular reflections.
specular exponent	Sets the specular exponent which determines the shininess of the material. Low values result in rough reflections, while high values result in clearer reflections. Typically, values between 50 and 5000 are used.
light exponent multiplier	This multiplier is applied to the specular exponent when computing specular reflections for light sources. This is motivated by the fact that light sources are usually not perfect point sources, but instead have an area. Lower values result in more specular highlights. Typical values range from 0.1 to 1.0.
receives shadows	Determines whether the material receives shadows.
enable reflections	By default, the surface material only takes light sources into account when computing the specular component. Enable this property to take the whole scene (including solids and fluids) into account.

<b>enable diffuse reflections</b>	By default, the surface material only takes light sources into account when computing the diffuse component. Enable this property to take the whole scene (including solids and fluids) into account.
<b>monte carlo light transport</b>	Enables or disables monte carlo light transport. This is necessary to render rough reflections or soft shadows.
<b>subsurface scattering weight</b>	Sets the weight of the subsurface scattering approximation as a value between 0 and 1. Please note that this currently only works for fluids and snow, but not for solids.
<b>subsurface scattering radius</b>	Sets the spatial radius used for approximating subsurface scattering. Please note that this currently only works for fluids and snow, but not for solids.
<b>light sources factor</b>	Sets a multiplier that is applied to lighting received by light source objects in the scene.
<b>skylight factor</b>	Sets a multiplier for the hemisphere skylight. To achieve good visual quality, it is recommended to also enable monte carlo light transport and diffuse reflections.
<b>headlight factor</b>	Sets a multiplier for the headlight that illuminates the material from the camera's point of view.
<b>render single particles</b>	This option is only relevant for fluids or snow and enables or disables the rendering of single particles. If disabled, the smoothed surface of the fluid will be rendered. If particle rendering should be used, it is recommended to use one of the provided presets ( <i>particles_fast</i> or <i>particles_quality</i> ) that also adapt lighting parameters in order to achieve good visual quality.

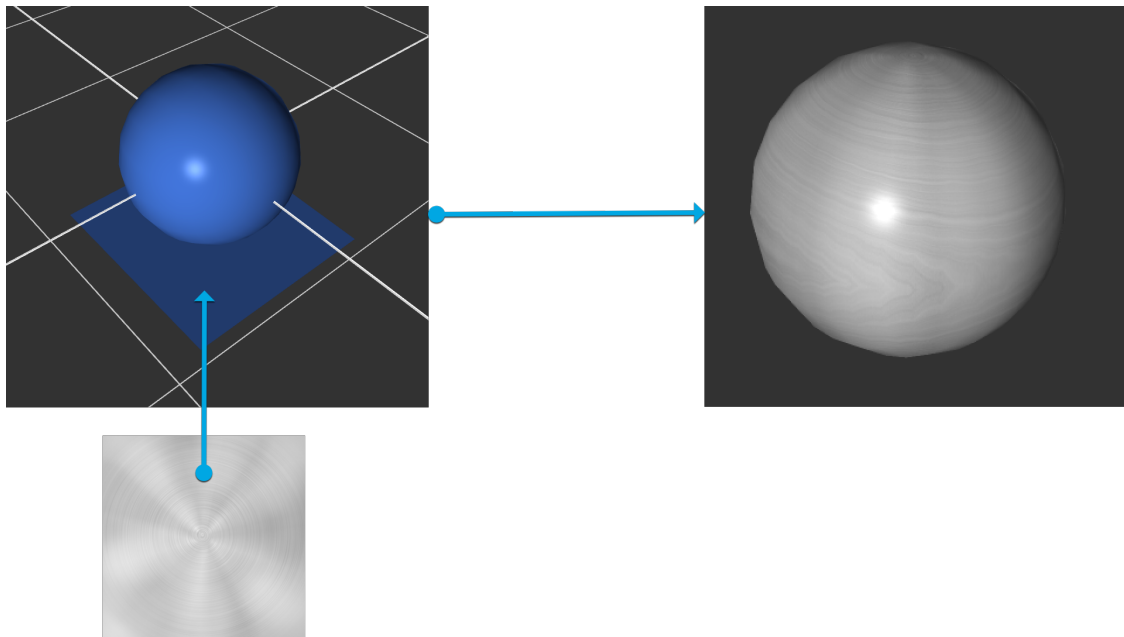
**Table 97:** Properties of Surface material in group **Material properties**.

### 15.5.3 Textured surface material

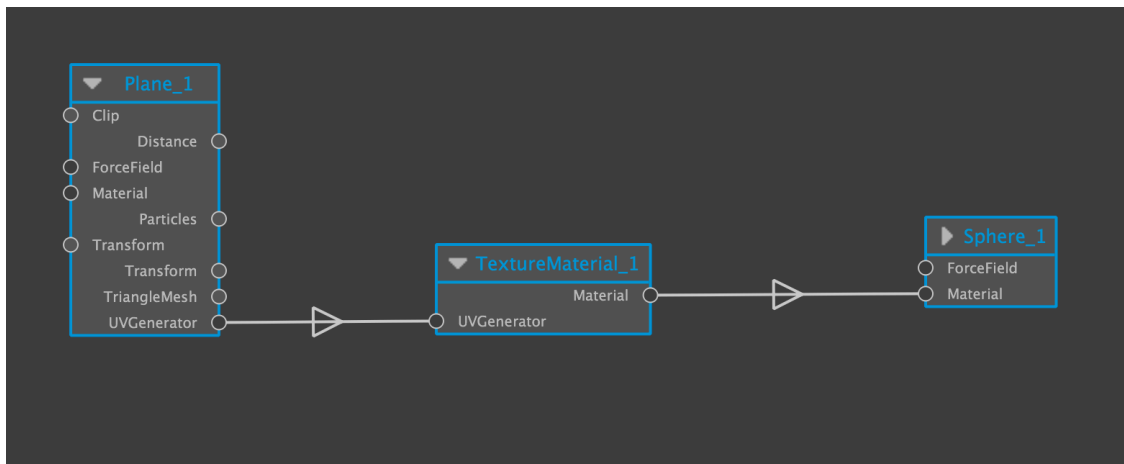
The Textured surface material gives you the option to project a texture onto a given mesh. In order to specify UV coordinates, a plane is used. The concept is illustrated in Figure 41. The plane needs to be connected to the Textured surface material as *UVGenerator* in the connection editor. The Textured surface material itself is connected to the mesh the texture is meant to be applied to as material, as depicted in Figure 42. The texture will only be visible in images generated with Preon renderer. Please note, that the plane is only an auxiliary object and can be set to **invisible** and **inactive** if it should not take part in the simulation. In addition to the properties of the **Surface material**, the **texture material** has the properties listed in Table 98.

Property	What it does
<b>texture file</b>	Path to the PNG texture image file.

**Table 98:** Additional properties of Textured surface material in group **Material properties**.



**Figure 41:** Texture is projected onto sphere using a plane as *UVGenerator*.



**Figure 42:** Connections needed for applying a texture onto a sphere using a Textured surface material.

### 15.5.4 Volumetric material

The Volumetric material is used to render volumetric mediums like water, oil or glass. It implements reflections, refractions and absorption so that deep water may appear less translucent than shallow water. It also includes a specular term similar to the Surface material to model direct reflections from incident light into the camera. In theory, the Volumetric material can be assigned to any object, but it is mostly used for fluids.

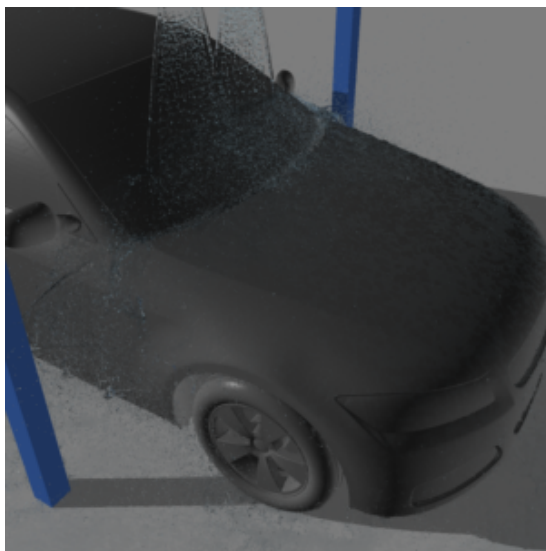
Property	What it does
----------	--------------

<b>absorption coefficient</b>	Sets the absorption coefficient that controls how much light is absorbed for rays travelling through the water volume. Low coefficients mean low absorption and higher coefficients mean high absorption.
<b>specular exponent</b>	Sets the specular exponent which determines the shininess of the light reflection. Typically, values between 50 and 100 are used.
<b>specular factor</b>	Scales the specular term of the Phong illumination system.
<b>index of refraction</b>	Sets the index of refraction for the material.

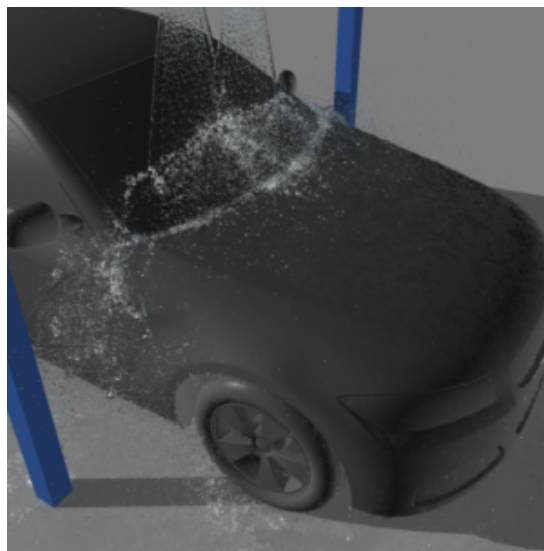
**Table 99:** Properties of Volumetric material in group **Material properties**.

## Spray model

The spray model is an option of the Volumetric material designed to improve the realistic visualization of waves and turbulent water. The spray model tries to detect turbulent regions in which water is mixed with air and renders them as spray as illustrated in Figure 44. Note that the spray model requires a detailed simulation to look convincing. It is not recommended for scenes with low particle counts.



**Figure 43:** Fluid rendered without spray model.



**Figure 44:** Fluid rendered with spray model.

Property	What it does
<b>enable spray</b>	Enables or disables the spray model.
<b>spray color</b>	Sets the spray color.
<b>spray factor</b>	Sets the factor weighting the amount of spray (should be between 0 and 1).
<b>spray velocity min</b>	Sets the minimum fluid velocity for rendering fluid as spray.

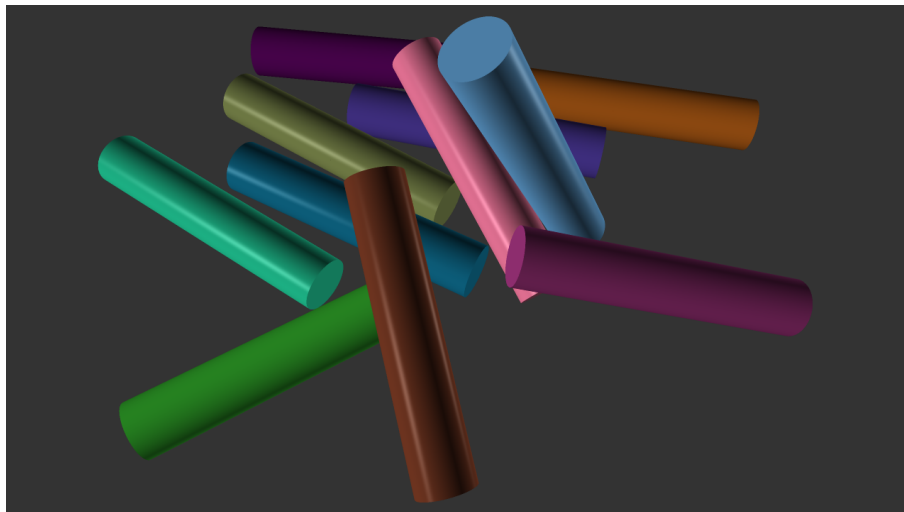
<b>spray velocity max</b>	Sets the velocity above which fluid is fully eligible to be rendered as spray (if mixed air is detected as well).
<b>spray threshold</b>	Sets a threshold value between 0 and 1 that determines when a mixture of fluid and air is considered as spray. Higher values mean more spray.
<b>ignore transp. background</b>	If enabled, the transparency of the background is not considered for reflections and refractions. This means that the fluid will have full opacity in the final rendering, even if a transparent background is used.

**Table 100:** Properties of water material in group **Spray model**.

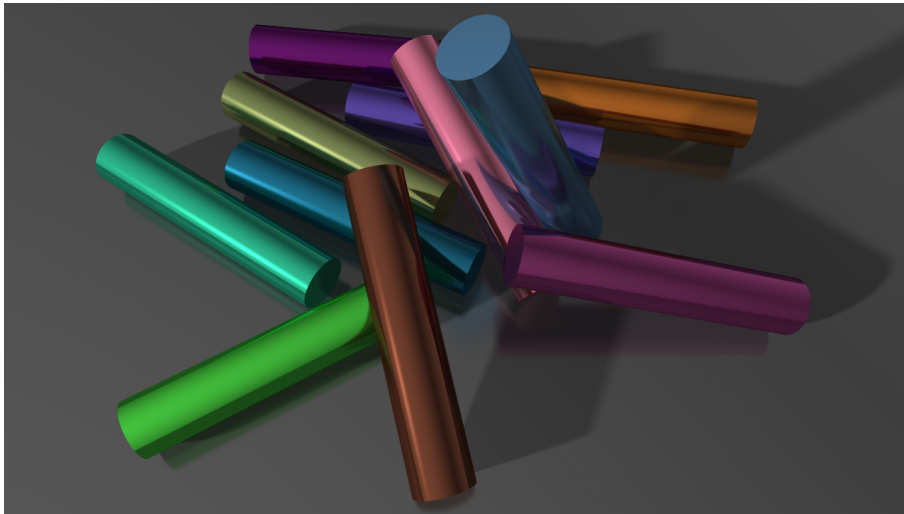
## 15.6 Presets

Preon renderer offers many possibilities to create visualizations from your simulations. To reduce the need for parameter tuning, material presets were introduced. Even if you do not find a perfect preset for your application, it is recommended to pick one that comes close and then adjust parameters from there. To apply a preset to a material, right-click on the material in the scene inspector and choose the preset you want to apply.

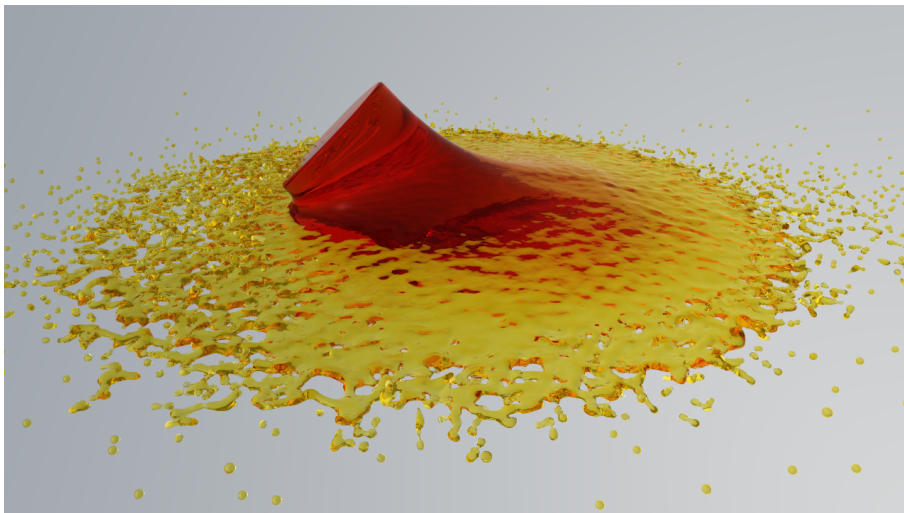
### 15.6.1 Examples



**Figure 45:** Cylinders on a plane rendered with the default material.



**Figure 46:** Here, the *metallic paint* preset was applied to the cylinders. The plane uses the *default preset (quality)*. Furthermore, shadows were enabled in the light source.



**Figure 47:** Fluid rendered with Volumetric material and the *motor oil* preset.

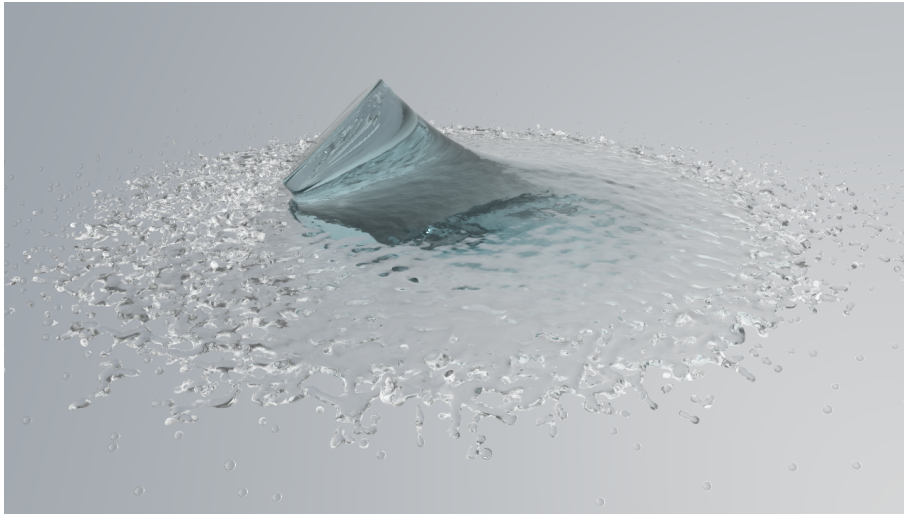


Figure 48: Fluid rendered with Volumetric material and the *water* preset.

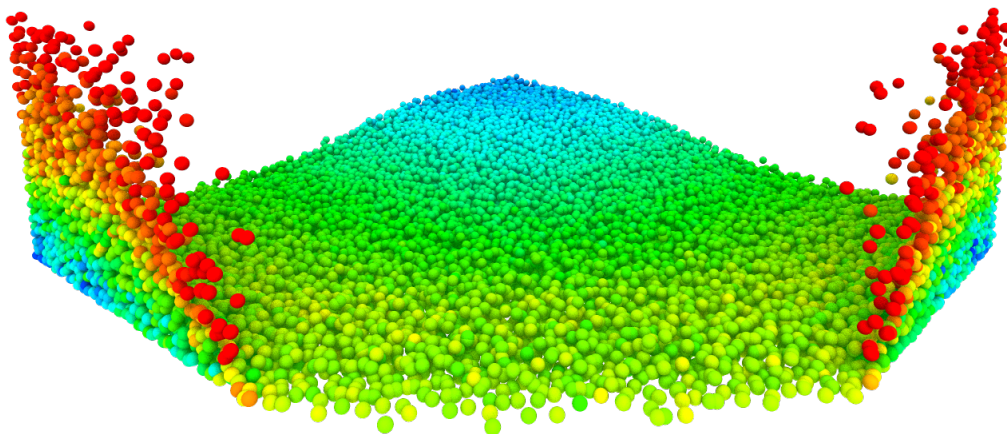


Figure 49: Fluid rendered with the Surface material and the *particles\_quality* preset.



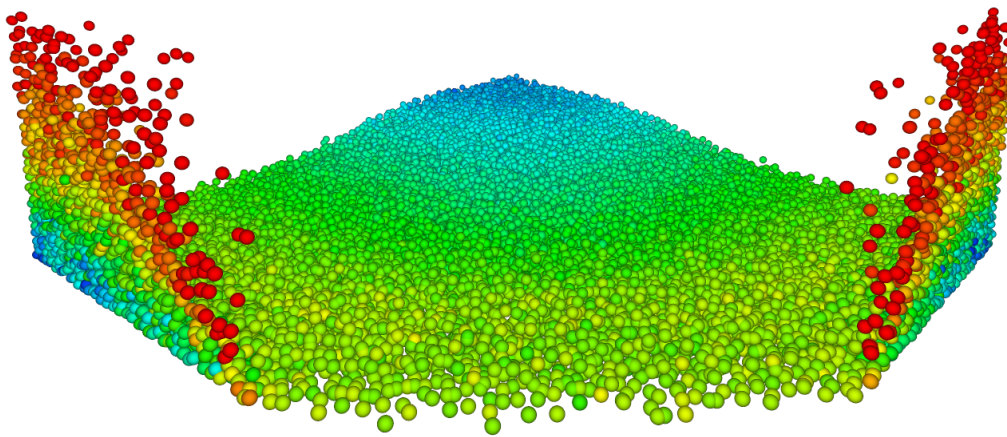


Figure 50: Fluid rendered with the Surface material and the *particles\_fast* preset.



## 16 Sensors

Sensors work identically in simulation mode and in playback mode. This means that sensors do update themselves and save the measured data on disk whenever the play button is pressed and the behavior of the sensor is active.

If the behavior is set to cache, sensors do not update, but load the already measured data from disk.

Every sensor logs a comma-separated value (csv) file on disk. The file is located under *[sceneFolder]/SensorData/[SensorName]/\*.csv*.

You can always reset or clean the sensor data by right-clicking the sensor in the scene inspector and clicking *CleanData*.

### 16.1 Mesh-based sensors

Mesh-based sensors (i.e., force sensor, heat flux sensor, height sensor, wetting sensor and sensor mesh) need to be connected to the solid object using the Input/Output slot *TriangleMesh* in order to define the surface of the sensor. The connection must be directed from the solid object to the sensor. A connection with the fluid solver is automatically added by PreonLab. Note that you can connect a force sensor, a heat flux sensor, a height sensor and a wetting sensor to the same solid object at the same time. In such a case, all sensors collect their data, but only one can be visualized via mesh coloring. Therefore, the sensors have a mesh coloring priority in the order given above. If you want to prioritize any particular sensor, set the **render mode** of all the sensors with higher priority to **invisible**. For example, if a force sensor and a wetting sensor are connected to the same solid, the mesh coloring shows the force sensor data. Turn the force sensor **invisible** to have the mesh colored with wetting sensor data. Further note that mesh-based sensors can not be connected to multiple solid objects at once and that a solid that is connected to a sensor mesh can not be connected to any other sensor at the same time.

### 16.2 Sensor color legend

Many sensors visualize data using a color. These objects have a property group **Coloring** that lets you control the mapping between data and color. The mapping is

defined by specifying a minimum, middle and maximum color for a min, mid and max value respectively.

By default, PreonLab sets these values automatically based on the sensor values. This automatic computation can be misleading, because a change of the data range can result in a very different coloring, even if the actual data mostly stayed the same. To avoid flickering artifacts during playback, it is therefore recommended to use a fixed mapping. To control the mapping manually, you need to disable the automatic flag and adjust the min, mid and / or max values.

### 16.3 Distance sensor

A distance sensor can be used to measure the distance between the local coordinates of two spatial objects in a scene. Up to two spatial objects may be connected to one distance sensor via the input/output slot *Distance*. Logically, a distance is only computed if exactly two spatial objects are connected (otherwise, the measured distance is set to zero). Note that the connection must be directed from the spatial objects to the sensor. No connections are added automatically by PreonLab.

The distance sensor can be helpful to track distances when arranging objects in a scene or running a simulation with moving objects. The easiest way to do this is to just connect two spatial objects to a distance sensor. Then, the distance between their global positions is measured.

Additionally, if you want to measure the distance between two specific spots on objects in the scene, you can use two **Points** (see Section 8.4) and the **Placement tool** (see Section 3.3.5). Add two points to the scene, place them on one or more objects with the placement tool and connect the points' *Distance* outputs to a distance sensor. Optionally, you can also connect the *Transform* output of the objects you placed the points on to the points' *Transform* input to track distances of the desired spots on moving objects. This procedure is significantly simplified by the **Measure tool** (see Section 3.3.6).

### 16.4 Volume sensor

A volume sensor measures the fluid volume inside the specified domain in m<sup>3</sup>. The volume domain can have a box or cylindrical shape that can be arbitrarily scaled and oriented. Furthermore, arbitrary custom meshes can also be used in order to restrict the shape of the volume sensor (See Section 16.4.1). If there is only a single fluid in the scene, volume sensors are automatically connected to this fluid via the *Particles* slot.

---

Property	What it does
shape	Defines the shape, <b>Box</b> or <b>Cylinder</b> , in which the volume is measured.

---

<b>sub particle precision</b>	If off, particles with position inside the volume sensor domain contribute with their full volume to the measured volume. If on, particles partially overlapping the sensor domain do only contribute the overlapping volume.
-------------------------------	---

**Table 101:** Volume sensor properties.

### 16.4.1 Using meshes as volume sensors

When using the box shape, it is also possible to restrict the volume in which fluid is measured with custom meshes. To do so, connect the *TriangleMesh* output slot of one or multiple solids to the corresponding input slot in the sensor. Then right-click on the sensor and select *Regenerate volume*. The properties in the group **Volume settings** specify how the meshes are considered when the volume is generated. This works mostly the same way it does for the volume source.

The sensor volume will only be updated once automatically (on first use, for example after loading a scene). In all other cases, it is required to explicitly trigger the recomputation using the right-click action.

Property	What it does
fill type	See Table 45 for more information.
manual border size	See Table 45 for more information.
border size	See Table 45 for more information.
volume generation frame	Defines the (view) frame in which the volume is generated. The volume will never be regenerated during post-processing or simulation. However, it will be transformed dynamically according to the sensor's position and orientation.

**Table 102:** Volume sensor properties in group **Volume settings**.

## 16.5 Wetting sensor

The wetting sensor measures the current and total amount of wetting of any solid object it is connected with. It measures and visualizes where and how much fluid has been or is currently in touch with a solid object. Furthermore, it can measure and visualize the duration the object stayed in touch with a fluid. See Section 16.1 on how to connect a solid with the sensor and the rules that apply.

Property	What it does
----------	--------------

<b>show wetting</b>	Specifies which type of wetting should be visualized. <b>AccumulatedWetting</b> visualizes all areas that were touched by fluid at some previous point in time while <b>CurrentWetting</b> only marks areas that are currently in contact with fluid. <b>WettingTime</b> visualizes the duration for which the sensor was touched by the fluid.
<b>Coloring (binary)</b>	Specifies the color employed to mark all the areas that were touched by the fluid. This group is enabled for all binary wetting types, i.e., current and accumulated wetting.
<b>Coloring</b>	Provides options to specify minimum, middle and maximum wetting time and their respective coloring (see Section 16.2 for more details) if <b>show wetting</b> is set to <b>WettingTime</b> .

**Table 103:** Properties of wetting sensor in group **Appearance**.

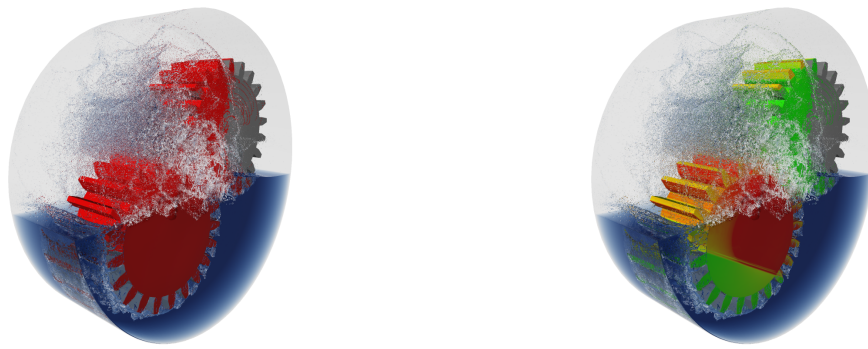
Property	What it does
<b>track wetting time</b>	If enabled, the wetting sensor collects and stores the wetting time of fluid particles on the sensor. Please note that data is not instantly collected, but requires a simulation or post-processing run. This property is automatically enabled whenever <b>show wetting</b> is switched to <b>WettingTime</b> . If you are not interested in the wetting time, disable it again to improve the performance and to reduce disk usage.

**Table 104:** Properties of wetting sensor in group **General**.

The wetting statistics are printed in percentage in the graphics window. Please note that the *Max wetting time* considers all parts of the solid object which were wet for at least the time specified by **Coloring** → **maximum wetting time** (measured from the start of the simulation/post-process run until the current time).

Depending on the type of wetting you have set, wet parts of the solid object are colored. For current and accumulated wetting, you can set one wet color in **Coloring (binary)**. Every wet part of the object in the current frame (**CurrentWetting**) or over time (**AccumulatedWetting**) is marked with this color. For visualizing the wetting time, ternary **Coloring** is employed. Define values for minimum, middle and maximum wetting time and their respective colors in order to visualize wetting times on solid objects.

You can export the wetting sensor data on a per-sample basis by right-clicking the sensor in the scene inspector and choosing "Export sample data to CSV". In the export dialog, select **Start Frame** and **End Frame**, whether you want to store the sample positions in a **Local coordinate system** and the measured quantities you want to export. The exported files are stored in the respective subfolder of the folder 'SensorData' in the scene directory.



**Figure 51:** A gear setup after a simulation time of 0.2s. The gears start rotating after 0.1s. The wetting sensor in the left picture visualizes accumulated wetting whereas the wetting sensor in the right picture visualizes the wetting time. For the latter, the color legend is as follows: Green:  $> 0.0s$ , Yellow:  $0.1s$ , Red:  $\geq 0.2s$ .

### 16.5.1 Known issues

The mesh coloring is sometimes not very accurate. Most notably, colors may bleed through thin surfaces. In these cases, it can help to look at the particle representation of the rigid. To view the particles, select the rigid and enable **show particles** in the **Appearance** property group.

## 16.6 Force sensor

The force sensor measures the forces acting from the fluid onto a solid object and the resulting pressure and shear stress at the solid object. The measured forces and torque measurements are divided into pressure, friction, adhesion and total forces. For each measured force, the net sum of all forces acting on the solid are measured. The unit of the measured forces is N and of the torque N m. See Section 16.1 on how to connect a solid with the sensor and the rules that apply.

Property	What it does
track max pressure	If enabled, each sensor sample will store the maximum force (per $m^2$ ) that acted on this sample at some point in time. If this was disabled during the simulation and is enabled afterwards, it will be necessary to run post-processing again to accumulate the data over time. The data can then be visualized by choosing the appropriate coloring mode and it can also be exported as CSV. Note that enabling this may require a significant amount of disk space.

**Table 105:** Properties in group **General** of Force sensor.

Property	What it does
----------	--------------

<b>coloring mode</b>	When choosing <b>Current</b> , the sensor will be colored according to currently measured forces. In this case, the properties below determine which force components will be displayed. When choosing <b>Accumulated_Max</b> , the sensor will colorize each sensor sample according to the maximum force (per m <sup>2</sup> ) that acted on this sample at some point in time. Accumulated data will only be available if <b>track max pressure</b> was enabled during the last sensor update run.
<b>physical value</b>	Defines according to which physical value the geometry is colored. Possible values are <b>ShearStress</b> or <b>Pressure</b> . Note that if <b>coloring mode</b> is set to <b>Accumulated_Max</b> , the physical value will always be <b>Pressure</b> even if <b>ShearStress</b> is chosen. This is a known issue and will be fixed in the future.

**Table 106:** Properties in group **Appearance** of **Force sensor**. These properties determine what is displayed on the mesh.

The min, max and avg pressure is measured in Pa and it only considers the perpendicular force acting locally on the surface. The min, max and avg shear stress is also measured in Pa and only considers forces acting tangential to the surface.

In order to use force sensors as a post-processor, i.e., on already simulated data, the pressure values of the solver have to be cached. See Section 9.1.12 on how to do this. Make sure that you do not transform the solid geometries after simulation. Currently, setting the behavior to cache does not work for the force sensor. Keep it always active.

You can export the force and pressure on a per-sample basis by right-clicking the sensor and choosing *Export sample data to CSV*. In the export dialog, select **Start Frame** and **End Frame**, whether you want to store the sample positions in a **Local coordinate system** and the measured quantities you want to export. The exported files are stored in the respective subfolder of the folder *SensorData* in the scene directory.

## 16.7 Particle tracker

The particle tracker measures the physical quantities of a single fluid particle with the user-specified **Settings**→**particle ID**. The trajectory of the particle is rendered as a single pathline which is color coded according to the particle velocity. Blue is used for velocity zero, red for values equal and above the user-specified maximum range **Appearance**→**max velocity** and green for the medium value.

## 16.8 Heat flux sensor

The heat flux sensor measures heat that is exchanged between a solid and fluid particles that are in contact with this solid. More specifically, it measures the heat flux as well as the heat transfer coefficient, i.e. their minimum, maximum and average.

The unit of the measured heat flux is  $\text{W/m}^2$  and is  $\text{W}/(\text{m}^2 \text{K})$  for the heat transfer coefficient. See Section 16.1 on how to connect a solid with the sensor and the rules that apply.

Property	What it does
ignore dry areas	If enabled, heat flux and heat transfer coefficient statistics ignore the surface areas where no fluid is in contact with the solid.

**Table 107:** Properties in group **General** of the **Heat flux sensor**.

Property	What it does
coloring mode	Specifies whether the sensor should be colored according to the measured heat flux or heat transfer coefficient.

**Table 108:** Properties in group **Appearance** of the **Heat flux sensor**. These properties determine what is displayed on the mesh.

The heat transfer coefficient, HTC, is computed with:

$$HTC = \frac{q}{T_S - T_F}, \quad (3)$$

with  $q$  being the heat flux.

Two types of heat transfer coefficient can be computed by the sensor: local or global. The local HTC computes the temperature difference between the temperature  $T_S$  of a solid particle and the average temperature  $T_F$  of its immediate fluid particle neighbors. In contrast, the global HTC employs the average temperature  $T_F$  that is measured by a **Sensor plane** (preferably outside the thermal boundary layer). Therefore, you have to connect the *MeasuredScalarValue* output slot of a **Sensor plane** to the *SensorPlaneTemperature* input slot of a **Heat flux sensor**.

You can export the heat flux and heat transfer coefficient on a per-sample basis by right-clicking the sensor and choosing "Export sample data to CSV". In the export dialog, select **Start Frame** and **End Frame**, whether you want to store the sample positions in a **Local coordinate system** and the measured quantities you want to export. The exported files are stored in the respective subfolder of the folder 'SensorData' in the scene directory.

The heat flux sensor supports MPI, however, with no substep updates, i.e., it will only measure values at full frames.



## 16.9 Pathlines

A pathline visualizes the trajectory traveled by a single particle over time. It also visualizes the velocity magnitude of the particle over time. The **Pathlines** object allows to select a set of particles and draw pathlines for them. In order to select the particles for which pathlines should be drawn, it is necessary to specify a point in time and a region in space. The space can be specified by moving, scaling and rotating the Pathlines box. The point in time is specified by the **capture frame** property. To start pathline generation, right-click on the object in the scene inspector and click on *Generate pathlines*.

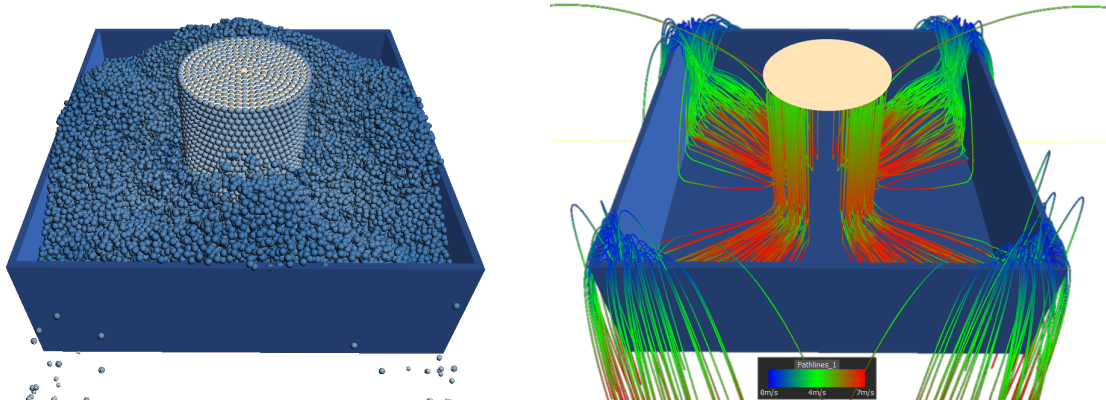


Figure 52: Pathlines for fluid particles flowing over the box.

If a pathline object has a transform parent, pathlines are automatically generated relative to this parent. Consider a moving train in which some water is spilled. In global space, the pathlines of the water would include the movement of the train, which probably would not tell you much about the water flow. In contrast, generating pathlines relative to the train would remove the movement of the train and would result in much more meaningful results as demonstrated in Figure 53.

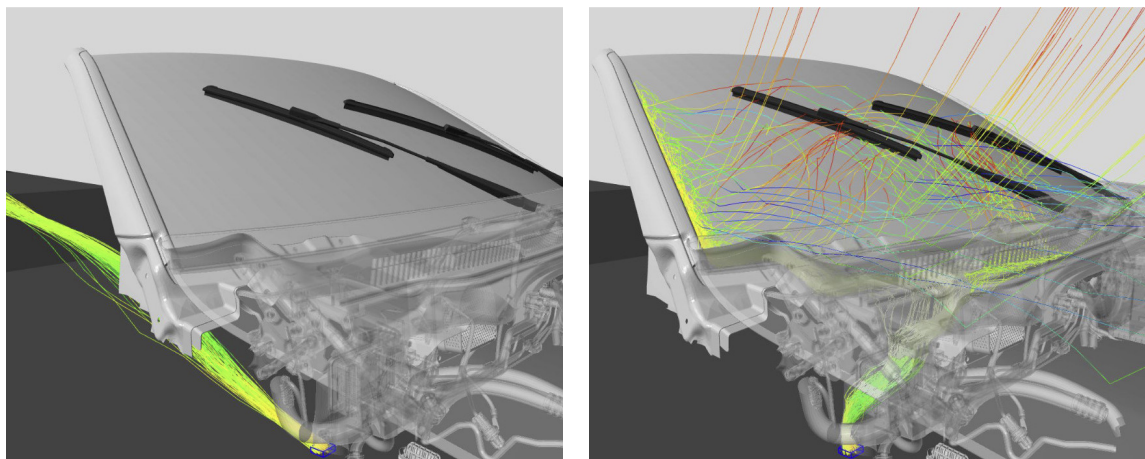


Figure 53: Global pathlines (left) versus pathlines relative to a moving object (right).



### 16.9.1 Parameters explained

Property	What it does
<b>capture frame</b>	Specifies the frame in which particles to be tracked are identified.
<b>thinning</b>	Specifies whether a grid should be used to reduce the number of tracked particles, resulting in less pathlines. Disabled by default.
<b>thinning factor</b>	Only relevant if Thinning is enabled. Specifies the ratio between ignored and tracked particles in one dimension.
<b>vertex limit</b>	When tracking many particles, pathlines may consume a considerable amount of GPU memory. This property limits the total amount of line vertices in order to prevent allocation of too much GPU memory. If this limit is reached, a message will be printed to the log.
<b>playback</b>	Enables or disables playback of pathlines over time.
<b>restrict lifetime</b>	Enables or disables the ability to restrict the lifetime of pathline segments. Thus, pathlines disappear again after a specified time during the playback. Prerequisite: <b>playback</b> has to be enabled.
<b>lifetime</b>	Specifies for how long a pathline segment is shown during the playback of pathlines over time. Prerequisite: <b>restrict lifetime</b> has to be enabled.

Table 109: Properties in group **Pathlines**.

Property	What it does
<b>line width</b>	Specifies the line width of the pathlines. For the OpenGL visualization, the line width is given directly as pixels. In Preon Renderer, the line width also depends on the distance from the camera and on the particle spacing. You may have to adjust the line width parameter when using Preon Renderer to achieve results similar to OpenGL.
<b>show box</b>	Specifies if the box visualizing the capture area should be visible.

Table 110: Properties in group **Appearance**.

### 16.9.2 CSV export

After generating pathlines, you can export them to a CSV file via right-click action. Please note that this functionality is only intended for users who wish to process the data manually using a script. The CSV file will contain four columns for each tracked

particle. Three of the columns are for the position and one (the W component) is the velocity magnitude. The CSV files will have one row for each frame covered by the pathlines (this range is determined when you enter the start and end frame into the pathline generation dialog). Some particles will have no position values in some rows, which means that they either do not exist yet at this point in time or that they were deleted.

## 16.10 Sensor plane

The sensor plane measures and displays different quantities of one or multiple fluids on a two-dimensional plane. A sensor plane always measures all possible properties and the user can decide which value is displayed by changing the value of **property**.

If there is only a single fluid in the scene, sensor planes are automatically connected to this fluid via the *Particles* slot.

Note that in order to get accurate measurements, the sensor plane needs to measure particles that are in a radius of two times the particle spacing from each cell of the sensor plane. Accordingly, if a sensor plane is placed too closely to a domain where particles are deleted (see Chapter 11), it may get inaccurate measurements.

Furthermore, in postprocessing, the results measured by a sensor plane can deviate from the results measured during simulation. This is due to the fact, that during postprocessing only selected particle states at each frame are available for the sensor plane to perform its measurements. If the fluid flow varies a lot in the time between two consecutive frames, there are not enough information to correctly reconstruct the flow in order for the sensor plane to be able to measure as accurate as during simulation.

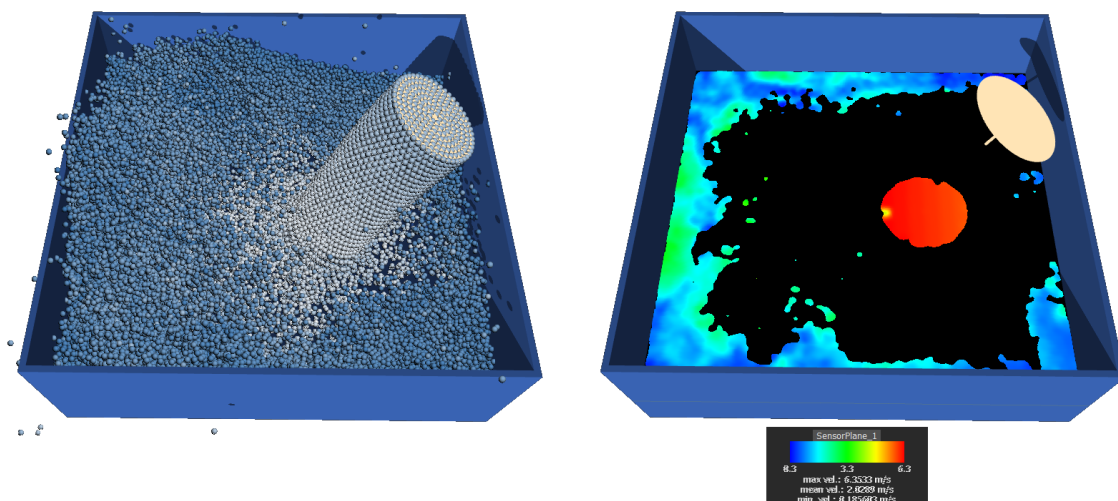


Figure 54: Sensor plane showing velocity magnitude for fluid flowing into a box.

Property	What it does
property	Sensor planes always measure all possible properties, e.g., <b>FlowRate</b> , <b>VelocityMagnitude</b> , <b>Density</b> , or <b>Pressure</b> . The user can set via <b>property</b> which value should be displayed on the plane. The sensor plane is by default connected to the fluid solver.
cell size	The cell size defines the resolution of the sensor grid. Higher resolutions give finer results, but take more time to compute. Per default the spacing of the fluid solver is used, larger values are not recommended and can not be set.
save image	For animations, the sensor values can be saved onto disk as an image by setting save image property to true. The image files will be located under <i>[scene-Folder]/SensorData/[SensorName]/[PropertyName].png</i> .
active cells	By default, the whole sensor plane measures data. This property can be used to mask out only specific portions of the sensor plane (referred to as cells). Setting this property to <i>image</i> allows to mask out active cells using a bitmap image. Setting it to <b>Triangle mesh</b> allows you to select active cells by projecting a triangle mesh on the sensor plane. To specify the mesh, connect the <i>TriangleMesh</i> slot of a solid object to the sensor plane using the connection editor.
relative flow rate	If on, the flow rate is measured relative to the sensor surface, taking its velocity into account.
flow rate mode	Controls how the sensor surface orientation is factored into the flow rate computation. A side of the plane is active if it has an arrow on it. Additionally, positive and negative arrows show which sign is applied for the incoming flow. <b>TwoSided_Diff</b> will sum up signed flow rates leading to a positive and negative arrow. A <b>TwoSided_Sum</b> will measure unsigned flow rates leading to two positive arrows. The two one-sided modes will only measure flow rates going into one direction which is represented by a single positive arrow.

Table 111: Properties in group **Settings**.

Property	What it does
density threshold	This value is used to decide if sensor data is shown at a specific cell of the sensor plane. Cells where the interpolated density value of the fluid is above the given threshold visualize the interpolated property.
color	Defines the color for cells that currently have no fluid data in proximity and, thus, measure no data for this instance in time. You can make those cells transparent by setting the <b>Alpha</b> value to zero.
display arrows	Toggles the <b>flow rate mode</b> arrows <b>on</b> or <b>off</b> . The property group <b>Arrow colors</b> will be hidden if arrows are toggled <b>off</b> .

Table 112: Properties in group **Appearance**.

### 16.10.1 Context actions

#### CSV export

You can export the sensor plane data on a per-sample basis by right-clicking the sensor in the scene inspector and choosing *Export sample data to CSV*. In the export dialog, select **Start Frame** and **End Frame**, whether you want to store the sample positions in a **Local coordinate system** and the measured quantities you want to export. The exported files are stored in the respective subfolder of the folder 'SensorData' in the scene directory.

#### Save cell status in image

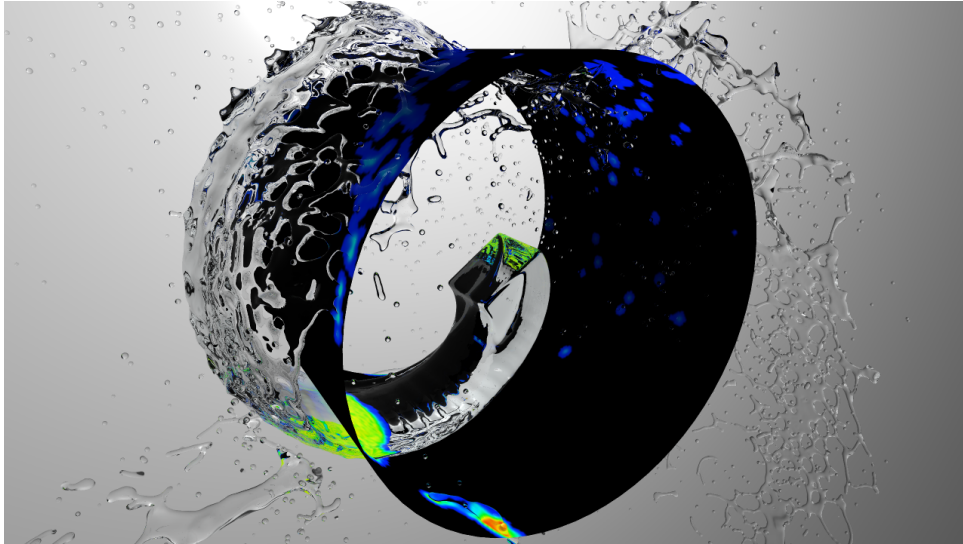
Each cell of a sensor plane can either be active and measure values or be inactive and not measure values. You can change the active cells using the **active cells** property described in Table 111. The *Save cell status in image* action lets you save the current status of each cell in an image. Active cells are visualized with black pixels while inactive cells are visualized with white pixels.

## 16.11 Sensor mesh

The sensor mesh is a generalization of the sensor plane. It measures and displays quantities of fluids on a surface that is defined by a solid object in the scene. Figure 55 shows an example of using this feature in PreonLab. See Section 16.1 on how to connect a solid with the sensor and the rules that apply. Establishing this connection will set the **behavior** of the solid to **inactive** by default to prevent the physical interaction with other solids and fluids during the simulation.

The sensor mesh shares the properties **property**, **relative flow rate** and **flow rate mode** with the sensor plane. You can read about these properties in Table 111.

Note that setting **flow rate mode** to anything other than **TwoSided\_Sum** will only give meaningful results if the underlying mesh consists of triangles that are oriented consistently. If one triangle faces into one direction and a triangle next to it to the other, the sensor mesh will show misleading flow rates. This is why in contrast to the sensor plane, **flow rate mode** is set to **TwoSided\_Sum** by default. To see whether the triangles of your mesh are oriented consistently, you can turn off **two-sided lighting** for



**Figure 55:** Cylindrical sensor mesh measuring flow rate. Image rendered with Preon renderer.

the solid. If the lighting is still smooth over the whole surface without visible seams, the triangles are probably oriented consistently.

## 16.12 Projection fields

Projection fields visualize different fluid properties projected on a plane. Thereby, only fluid above the plane is considered.

### 16.12.1 Velocity projection field

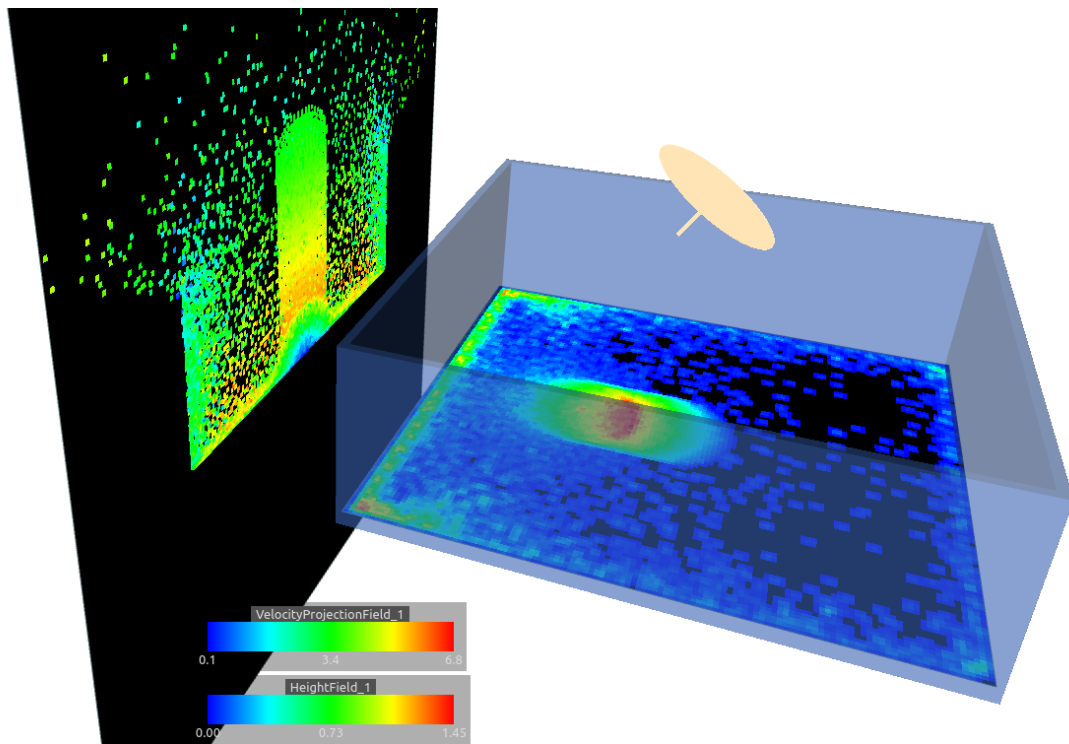
The velocity projection field visualizes average velocity magnitudes projected on a plane.

### 16.12.2 Height field

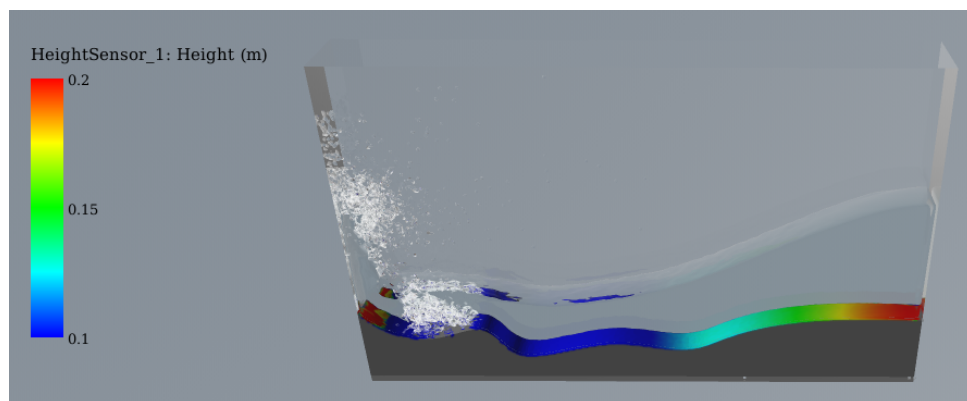
The height field visualizes fluid height on a plane.

## 16.13 Height Sensor

The height sensor is a generalization of the height field with one important difference; It only measures material directly in contact with the sensor. Figure 57 and Figure 58 demonstrate how the height sensor can be used with Preon Solver and Snow Solver, respectively. Figure 59 illustrates how splashed particles are not considered by the



**Figure 56:** Invisible fluid is pouring from a circle source into a box. A height field is placed at the bottom of the box and a velocity projection field is placed left to the box.



**Figure 57:** Height sensor used with arbitrarily shaped solid mesh.

height sensor. Furthermore, the height sensor is defined by a solid object in the scene and its surface. See Section 16.1 on how to connect a solid with the sensor and the rules that apply.

Property	What it does
sampling rate	Specifies the rate with which the fluid surface is sampled as a factor of the fluid spacing. A lower sampling rate trades accuracy for performance.
height threshold	Height multiplied with fluid spacing between fluid and sensor that still counts the fluid as on top. You may need to tweak this if <b>direction pointing</b> is set to <b>Custom</b> and measuring near parallel to the surface.

<b>min culling neighbor count</b>	Sets the neighbor count threshold above particles inside the fluid volume may be selected for efficient culling. In normal cases it should never be necessary to change this, but you may try to increase it when experiencing areas that should have a height but are missing.
<b>direction pointing</b>	<b>OneSided</b> measures height in normal direction and <b>OneSided_Flipped</b> in inverted normal direction. <b>TwoSided</b> measures both sides and outputs the maximum. <b>Custom</b> measures in a user-defined direction.
<b>direction</b>	Only shown for <b>direction pointing</b> set to <b>Custom</b> . Sets the direction in which to measure.
<b>local transform</b>	Only shown for <b>direction pointing</b> set to <b>Custom</b> . When enabled, <b>direction</b> is in the local coordinate system of the mesh and will rotate with the object. Set to <b>off</b> the <b>direction</b> is global and fixed.

**Table 113:** Properties in group **Height estimation**.

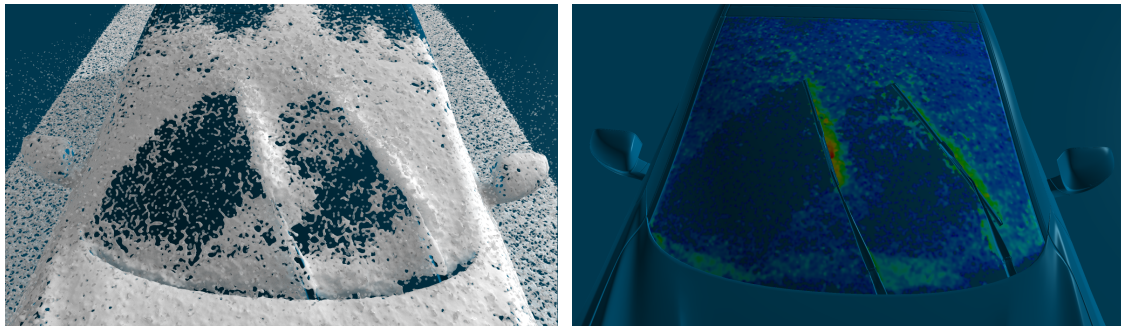
Note that setting **direction pointing** to **OneSided** or **OneSided\_Flipped** will only give meaningful results if the underlying mesh consists of triangles that are oriented consistently. If one triangle faces into one direction and a triangle next to it to the other, the height sensor will show misleading height. This is why **direction pointing** is set to **TwoSided** by default. To see whether the triangles of your mesh are oriented consistently, you can turn off **two-sided lighting** for the solid. If the lighting is still smooth over the whole surface without visible seams, the triangles are probably oriented consistently.

You can export the height on a per-sample basis by right-clicking the sensor and choosing *Export sample data to CSV*. In the export dialog, select **Start Frame** and **End Frame**, whether you want to store the sample positions in a **Local coordinate system**. The exported files are stored in the respective subfolder of the folder *SensorData* in the scene directory.

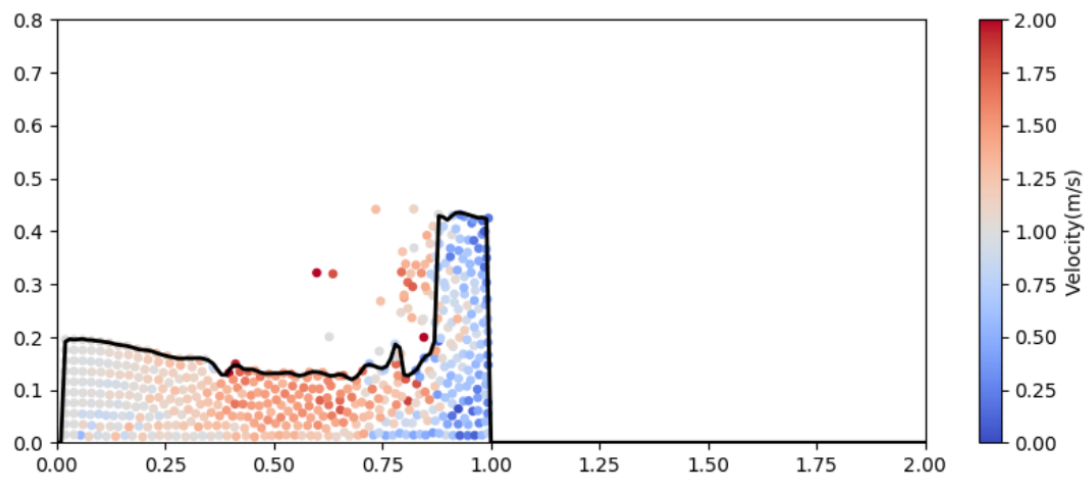
## 16.14 Air flow visualizer

You can visualize the velocity of an air flow field (see Section 12.3) using an **air flow visualizer** which can be added to your scene via *Add*→*Sensor*→*Air Flow Visualizer*. The air flow visualizer works like a sensor plane for air flow fields. As the air flow field only stores velocity data, the air flow visualizer can only measure and visualize velocities. The air flow visualizer is automatically connected with the first inserted air flow field.





**Figure 58:** A car has been snowed in and the wipers just got active clearing the front window. A height sensor is connected to the front window and measures the heights on top of the front window.

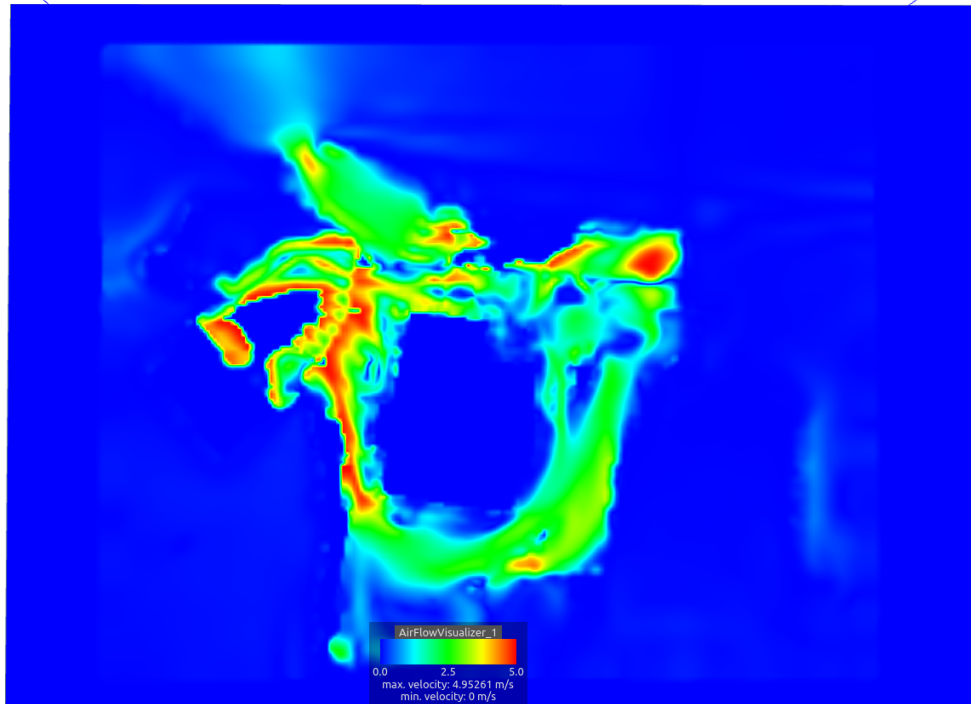


**Figure 59:** Fluid is released from a dam on left side and hits a wall at 1m on the right side. Results are obtained in PreonLab and visualized in matplotlib with a PreonPy buffer set. The black line represents the heights obtained by the height sensor. Only fluid in direct contact with the surface is measured. Splashed particles are ignored.

Property	What it does
cell size	Sets the cell size of the air flow visualizer. When using the grid-based velocity storage for the connected air flow, this will be automatically set to one tenth of the air flow cell size (but can still be changed manually). When using the gridless velocity storage, the cell size must be chosen manually.
save image	If enabled, the sensor image will be written to disk when post-processing or simulating. The image files will be located in <code>[sceneFolder]/SensorData/[SensorName]/[PropertyName].png</code>

**Table 114:** Properties in group **Settings**.





**Figure 60:** The air flow visualizer displays air flow velocity magnitudes on a two-dimensional plane.

### 16.15 Python particle access

In some situations, the PreonLab's built-in sensors, might not exactly offer what you need. In such cases it might be an option to directly access the particles and the attached values like velocities from Python. See section Section 19.3 and the `preonpy` API documentation linked from there for further information.

## 17 Import & Export

### 17.1 Scene loading and saving

Scene loading and saving can be initiated via the toolbar or the *File* menu in the taskbar. Scene loading can be also directly triggered when starting PreonLab with the `--scene` parameter.

PreonLab requires and enforces the scene directory to be located in the same folder as the scene file. The directory must have the same name as the scene file (minus the extension).

The *Save as* dialog gives you full control on how the existing data is handled (see Figure 61). The central part of the dialog contains a list of objects with sub-entries for the different data categories. Selecting an entry means that the corresponding files are moved/copied to the target directory.

Data is partitioned into five categories, whereas *External Resources* is a special case of the *Resources* category:

**Resources:** Represents data that is in general needed to simulate or post-process. Please note that only the files inside of the scene directory are counted (see the next category).

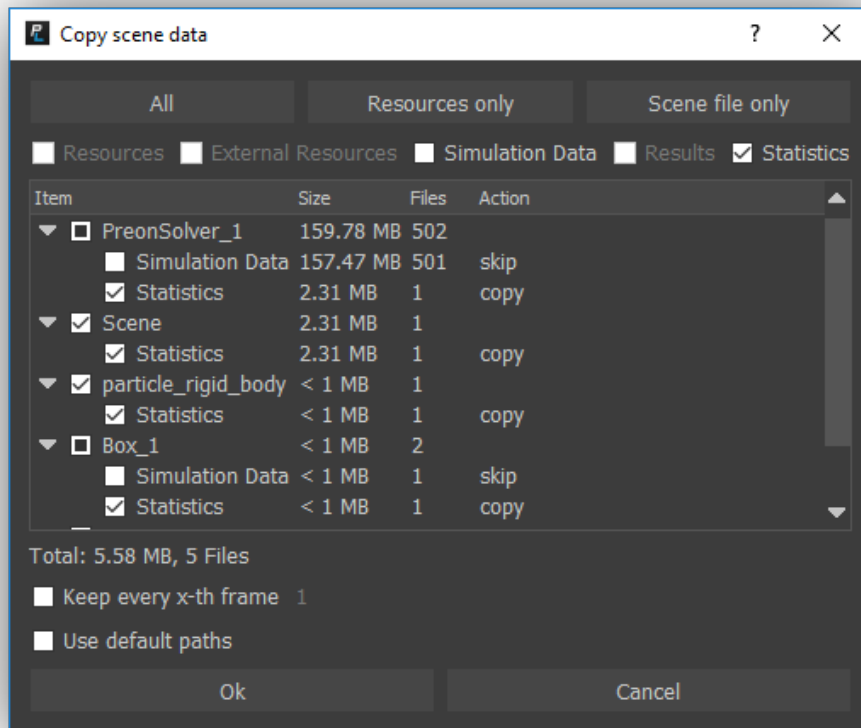
**External Resources:** The same type of data as *Resources*, but located outside of the scene directory. Selecting these entries will copy the respective data to the scene directory. Deselecting these entries will keep the scene intact, because the path to external resources will be still valid after saving. However, if you want to port the scene to another computer and the scene depends on resources like mesh files that are referenced using absolute file paths, make sure that not only the *Resources*, but also all *External Resources* items are selected. This ensures that all dependent file resources are copied in the scene directory and are referenced using relative paths.

**Simulation Data:** This is mainly the particle data and some related information such as the sampling. It is needed for post-processing or for resuming a simulation.

**Results:** Encloses sensor data, renderings and other data resulting from post-processing (and also simulation if post-processing objects are active during simulation).

**Statistics:** Statistic data collected during simulation or post-processing.

If you save to another directory, the existing data is kept at it's original position. There



**Figure 61:** The save as dialog shows an overview of the amount of data used by the scene and let's the user specify how to handle it.

is one exception: Deselecting entries in untitled scenes actually removes them if they are stored inside the untitled scene directory.

It is also possible to remove certain type of data from the current scene directory by invoking the *Save as* dialog and by specifying exactly the same path and name of the current scene. In this case, not selecting an entry corresponds to deleting the data (except for *External Resources* of course).

The *Keep every x-th frame* option can be used to thin out data and is explained in Section 17.1.1.

If you want to ensure that resources are stored in a consistent fashion you can check *Use default paths*. Then PreonLab copies resources into the default directories like *Geometries* and *Airflows* inside of the scene directory. This could especially make it easier to implement automated workflows which copy scenes to other locations e.g. on clusters.

### 17.1.1 Archiving and reducing disk space consumption

In order to reduce the amount of space used by a scene, you have certain options. As a first step you can open the save as dialog to get an overview of the amount of data certain objects use. Consider re-evaluating whether you need to keep everything. Maybe you can leave out the statistics or all the data of certain objects.

It is also possible to reduce the number of frames that are stored. If a scene was simulated with a *simulation framerate* of 200 frames per second, a divisor of 10 would lead to keeping the frame 0, 10, 20, 30 and so on. The data of all frames in between would be deleted. In the example above, the simulation framerate would be set to 20 and the remaining files would be renamed to 0, 1, 2, 3... accordingly. You can enable this feature by checking the *Keep every x-th frame* box in the *Save as* dialog and entering the divisor. Note that this only works on data that corresponds to the simulation framerate, not the view framerate.

As mentioned in Section 17.1, it is possible to apply the above mentioned options to the current scene directory. In this case data is irretrievably removed. If another directory is chosen, then only the selected subset is copied over to the new directory and the existing scene directory still contains everything.

### 17.1.2 Known issues

Currently, PreonLab can not handle two specific types of resource files.

The first one is the **background image** in the **Scene UI Settings**. Either keep it outside of the scene directory, referenced using an absolute path, or make sure that file is copied manually to the target location after saving the scene to another directory.

For Transient airflow resources, the frame-related *.prairflow* files, located in the same directory as the *.case* file which is referenced by the **air flow path** property, are not copied by PreonLab. Please copy them manually.

## 17.2 Import meshes

Solid objects can be imported either via *File→Import→Import Mesh* or per drag-and-drop. It is also possible to import multiple files at once. You can choose if you want to import the files as mesh resource, which allows to create separate mesh objects for each submesh contained in the file. This can be selected for all files or for each file individually.

You can also create a Transform group connected to all rigids (from all files). This can be handy, if you have an object separated into multiple files, but the partitioning does not play a role for the simulation.

You can also specify the unit in which the mesh was modeled in for all files. This value is then set in the created resource objects or as scale factor in the mesh objects, if no

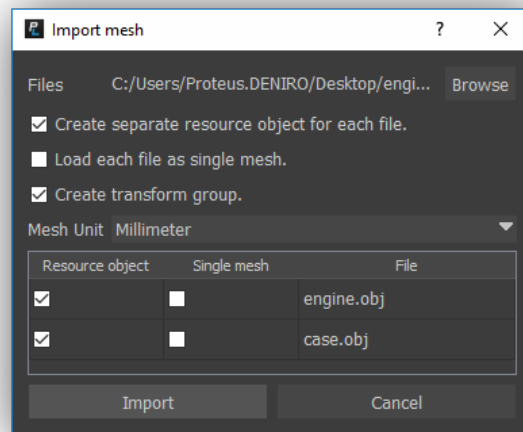


Figure 62: Mesh import dialog.

resource object should be created

### 17.3 Import animation data

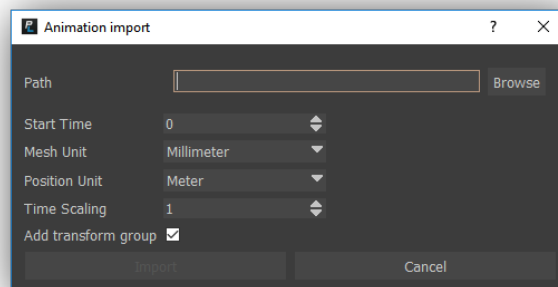


Figure 63: Animation import dialog.

PreonLab can import multiple meshes together with their animation. This data must be provided in a specific folder structure and format. The import can be started via *File→Import→Import Animation* which opens a dialog as shown in Figure 63. In the path option of the dialog, the top folder of the data must be provided. The *Start Time* field allows you to offset the animation, e.g., when your data starts at 0 s and you insert 1 into the dialog, the animation will start at 1 s in PreonLab. The *Mesh Unit* and *Position Unit* fields allow you to specify the units for the mesh files and the positions in the animation files. This is required for PreonLab to convert the imported

meshes and positions to meter. The *Time Scaling* field allows to change the speed of the animation. A value of 0.5 would result in an imported animation that is twice as fast as the original data. The *Add Transform group* checkbox allows you to decide if a transform group connected to all imported meshes should be added.

### 17.3.1 Data format

The top folder provided in the dialog must contain two subfolders: *meshes* and *animation*. For each geometry file (which can have any format readable by PreonLab) in the *meshes* folder, an object is created when importing. The *animation* subfolder contains the animation data for these meshes, which are imported into PreonLab as keyframes. If no animation data is found, the mesh is imported anyways but will not have any keyframes. The animation data per object can be provided in two different ways:

1. PreonLab keyframe format: A file named *meshName.csv* using the PreonLab keyframe format (see Section 6.1.2).
2. Custom format: For each mesh file in *meshes* folder, seven *txt*-files must exist which provide:
  - **position** (for x, y and z). The unit can be changed using the *Position Unit* field. The position files must be named *x\_meshName\_Position.txt*, *y\_meshName\_Position.txt*, *z\_meshName\_Position.txt*.
  - **rotation axis** (for x, y and z) given as direction vector. These files must be named *x\_meshName\_Orientation.txt*, *y\_meshName\_Orientation.txt*, *z\_meshName\_Orientation.txt*.
  - **rotation angle** around the rotation axis given in degree. This file must be named *angle\_meshName\_Orientation.txt*.

Each of these files must contain two columns separated by a space with a row for each time sample point. The first column contains the time and the second column the value as exemplified below. The sample time stamps must be equal across all imported files.

```
24.00 1.52
24.01 2.50
24.02 3.62
```

## 17.4 Import VDAFS data

PreonLab allows to import data from a *\*.vda*-file using the **VDAFSResource**-object. However, PreonLab currently only supports loading transformation data from *\*.vda*-file. A **VDAFSResource**-object can either be created by dragging and dropping a *\*.vda*-file into PreonLab or by choosing *Add→Resource→VDAFSResource*.

The created object loads the transformation data from the file as keyframes for itself. Other objects can then use this imported data by making a *Transform* connection from the **VDAFSResource**-object to them. See Section 5.2 for more information regarding *Transform* connections.

The imported transformation of the **VDAFSResource**-object can be viewed in the keyframe editor (see Section 6.1). Please note, that you can change the keyframes in the editor, however once you change a property of the **VDAFSResource**-object or reload the scene, all transformation keyframes will be replaced again by the ones loaded from the \*.vda-file.

Property	What it does
file	Specifies the path to the loaded *.vda-file.
sample time interval	This allows to define the time interval between subsequent transformation samples loaded from the *.vda-file. This is necessary as the file only provides transformation samples without notion of a time for each sample.
time offset	Specifies the time for the first loaded sample.
units	Allows to uniformly scale the loaded positional data with the origin as scaling center.
num. repetitions	Specifies how often the loaded transformation samples should be repeated. This repeat a movement multiple times which is only appears once inside the *.vda-file.

Table 115: Properties for **VDAFSResource**.

## 17.5 Import Alembic file

An Alembic file can be imported by either choosing *File→Import→Import Alembic File* or by dragging and dropping the Alembic file into PreonLab. A dialog opens that allows to choose which objects contained in the given Alembic file should be imported. For each selected object, an **Alembic Mesh** object is created. More information on the properties of an **Alembic Mesh** can be found in Section 13.6.

### 17.5.1 Alembic files with HDF5 data format

Beginning with version 3.2, PreonLab only supports the new Ogawa data format for Alembic files. The old HDF5 format is no longer supported. This means that only Alembic files using the Ogawa format can be imported by PreonLab. Furthermore, Alembic files exported out of PreonLab are always written with the Ogawa data format.

A command-line converter tool is provided on the download page for PreonLab v3.2.<sup>1</sup>

<sup>1</sup>[PreonLab Downloads](#)

The converter tool can be used to convert the data format of an Alembic file from HDF5 to Ogawa or vice versa.

## 17.6 Import Airflow

See Section 12.3 for information on how to import airflow fields.

## 17.7 Import statistic data

PreonLab allows to import statistic data via *Add→Resource→Statistic resource*. Imported statistic data is handled as statistic data created by PreonLab. Accordingly, it can be visualized in the plot dialog and compared to other statistics. As an example, you can compare exported statistic from previous simulation runs with current results, or compare current results with externally generated data, e.g., from a spreadsheet. As soon as a **Statistic resource** object is created, you are asked to provide an input file in the CSV file format, whereat semicolons are used as separators. For instance, a file with information about current and accumulated wetting percentages at two timestamps may look like this:

```
Time;CurrentWetting;AccumulatedWetting
0;0;0
2.0;10;30
```

You can import new statistic data into existing **Statistic resource** objects by right-clicking on the object in the scene inspector and selecting *Import*.

## 17.8 Import Point Cloud Resource

A **Point cloud resource** stores three-dimensional sample points with physical quantities like temperature or velocity. This point cloud can be used by other objects, for instance in order to map pre-defined temperatures on solid objects. A **Point cloud resource** is usually created using an import dialog (see below), but it is also possible to insert one directly via *Add→Resource→Point cloud resource*. In the latter case, the resource can be initialized via one of the right-click actions in the scene inspector that will open the respective import dialog.

### 17.8.1 Importing temperature samples from CSV

To import a CSV file with temperatures, choose *Import→Import Temperature Field*. An import dialog opens and asks for all the necessary information, see Table 116.



Property	What it does
Path	Provide a valid path to a CSV file.
Separator	Common separators are automatically detected and shown here. If your file employs a separator which cannot be automatically detected, you have to enter it manually.
X-Position Name	Choose the respective column name from the drop-down list to specify the sample point's x-position.
Y-Position Name	Choose the respective column name from the drop-down list to specify the sample point's y-position.
Z-Position Name	Choose the respective column name from the drop-down list to specify the sample point's z-position.
Temperature Name	Choose the respective column name from the drop-down list to specify the sample point's temperature.
Distance Unit	Provide the unit in which the sample point positions are specified. Default is <b>Millimeter</b> .
Temperature Unit	Provide the unit in which the temperatures are specified. Default is <b>Celsius</b> .

**Table 116:** Import dialog input fields for importing temperatures.

After the **Point cloud resource** has been added to the scene, it can be used to map the stored temperatures on solid objects. To do so, create a connection between the resource and the receivers of the mapping, i.e. the solid objects via the *Temperature-Samples* slot. Finally, the mapping has to be triggered via right-clicking on the solid object in the scene inspector and selecting *Apply connected temperatures*. Note that this action becomes available only after:

- a) a point cloud resource is connected
- b) thermodynamics are enabled
- c) the scene contains at least one Preon solver

## Best practices

In the following, we recall important steps to consider in order achieve a proper mapping:

- The mapping can only be executed on solid objects of type **Mesh**, i.e. imported meshes. The built-in basic shapes do not support the mapping.
- Set **coloring** to **TemperatureBased** and **enable mesh coloring** in property group **Appearance**→**Coloring** before applying the connected temperatures to the selected solid object to immediately see the mapping results.
- Toggle **Appearance**→**show particles** of the respective solid object to verify that the particle spacing can exhibit all the details of the distribution.

- After importing a point cloud, the OSD of the **Point cloud resource** shows the number of imported sample points and the minimum and maximum temperature in degree Celsius. If the numbers are not reflecting what you expect, check the correctness of the provided CSV columns as well as the units for the **Point cloud resource** in the property editor.
- After you have applied the connected temperatures of a **Point cloud resource** to a solid object, the bounding box of the **Point cloud resource** becomes visible and indicates the distribution of the samples within the scene. The size of the bounding box might indicate an incorrectly set distance unit.
- In case the point cloud samples do not cover the mesh of the solid object completely, the default temperature (to be set as a thermodynamics property of the solid object itself) is set in the respective particles.

## 17.8.2 Importing velocity samples from CSV

To import a CSV file with velocities, choose *Import*→*Import Velocity Field*. An import dialog opens and asks for all the necessary information, see Table 117.

Property	What it does
Path	Provide a valid path to a CSV file.
Separator	Common separators are automatically detected and shown here. If your file employs a separator which cannot be automatically detected, you have to enter it manually.
X-Position Name	Choose the respective column name from the drop-down list to specify the sample point's x-position.
Y-Position Name	Choose the respective column name from the drop-down list to specify the sample point's y-position.
Z-Position Name	Choose the respective column name from the drop-down list to specify the sample point's z-position.
X-Velocity Name	Choose the respective column name from the drop-down list to specify the sample point's x-velocity.
Y-Velocity Name	Choose the respective column name from the drop-down list to specify the sample point's y-velocity.
Z-Velocity Name	Choose the respective column name from the drop-down list to specify the sample point's z-velocity.
Distance Unit	Provide the unit in which the sample point positions are specified. Default is <b>Millimeter</b> .
Velocity distance Unit	Specifies the distance unit in which the velocities are defined. Default is <b>Meter</b> . The time is always in seconds.

Table 117: Import dialog input fields for importing velocities.

The imported velocity field can be used to define the initial velocities of particles emit-

ted by a **Volume Source**, see Section 10.2.4 for more information.

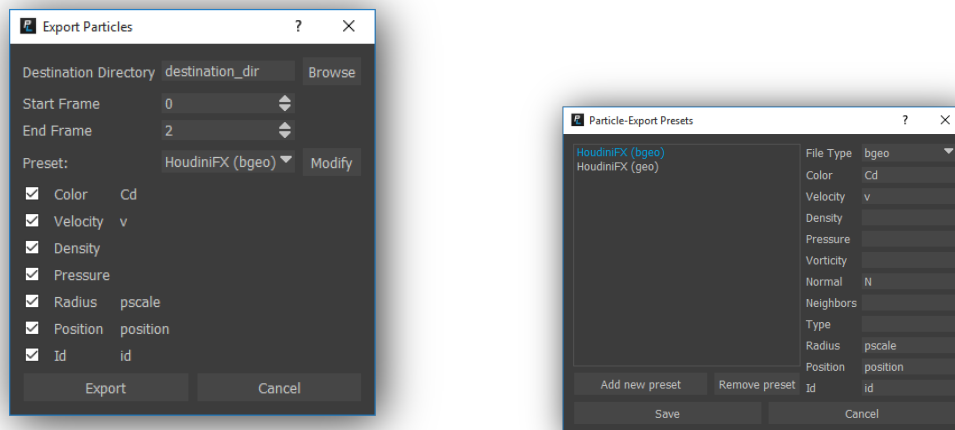
## 17.9 Export Alembic file

Objects in the scene can be exported as an Alembic file. Choose *File*→*Export*→*Export Alembic File* to open an export dialog. In this dialog, you can choose the path of the created Alembic file and the objects and the frames that should be exported. Only solid objects are supported to export. Furthermore, you can choose if the triangle orientation of the exported meshes should be flipped. This is imported depending on the program you intend to use to import the exported Alembic file.

Note, that beginning with version 3.2, Alembic files exported from PreonLab always use the Ogawa data format. See Section 17.5.1 for more information.

## 17.10 Export particle data

Particle data of a specific solver can be exported to other formats by right-clicking on the solver in the scene inspector and selecting *Export Particles*. The dialog that opens (see Figure 64) allows to choose the destination directory for the exported particle data, the start and end frame and the format to export. PreonLab comes with some format presets. Additional presets can be configured by clicking on *Modify* next to the preset which opens an additional dialog as shown in Figure 64.



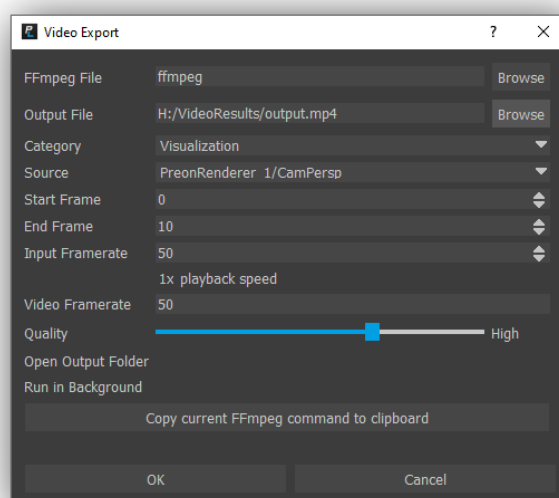
**Figure 64:** The left dialog allows to export particle data. The right one is used to change export format settings.

## 17.11 Export to EnSight

PreonLab can export particle data to the EnSight Gold format. At the moment only the particle positions, IDs, velocities and densities can be exported. Note that the IDs are not stored in separate files, but together with the positions. You can choose whether a particle's data should be exported as part of a node or an element. The presented options depend on the caching settings of the selected fluid object which can be changed in the *Property Editor* in the group **Serialization** (see Section 9.1.12). Note that changing the caching settings after the simulation won't change the simulation data which is already stored on disk.

## 17.12 Export video

The image sequence produced by the Preon renderer (see Section 15.4) can be assembled into a video using the *Export video* dialog. The assembly itself is performed by the third party tool *FFmpeg*<sup>2</sup> which has to be downloaded and extracted onto the computer running PreonLab as a prerequisite<sup>3,4</sup>. Please note that we require a version greater or equal to 2.6.8. for matching command line arguments.



**Figure 65:** Export dialog for video assembly with FFmpeg.

If you use the dialog for the first time, you have to give the path to the *FFmpeg* executable (see Figure 65). Next, the output directory, filename and file extension have to be set via the file browse dialog or by editing a previously used *Output File* manually.

<sup>2</sup><https://www.ffmpeg.org/>

<sup>3</sup>To install on a Windows OS, download the latest *version* choosing the respective *architecture* and *static linking* from here: <https://ffmpeg.zeranoe.com/builds/>

<sup>4</sup>To install on a Linux OS, download the latest *release* choosing the respective *architecture* from here: <https://www.johnvansickle.com/ffmpeg/>

The *Category* can be either a visualization (e.g., a renderer) or sensor data. Depending on the category, you can choose a *Source*, e.g., a specific sensor or camera. PreonLab automatically detects generated images in the scene folder and fills the drop-down lists of these two fields accordingly. *Start Frame* and *End Frame* are read from the image sequence files located within the respective *Source* directory. *Input Framerate* defines the number of rendered images employed by *FFmpeg* in one second of video. As default, the *view frame rate* is set. *Playback Speed* indicates a realtime playback of 1x in this case, since equally many images are used per video second as have been rendered during one simulation second. If you increase the *Input Framerate* above the *view frame rate*, you will get an accelerated playback, while decreasing it will result in a slow-motion video. *Video Framerate* defines the number of frames per second in the final video. For a smooth output, this should be equal (recommended) or lower than the *Input Framerate*. For example, rendering with a *view frame rate* of 50 produces this number of images. Setting the *Input Framerate* to 25 results in a *Playback Speed* of 1/2x. If the *Video Framerate* is 50, this would mean that one input image is used twice in the output video. Consequently, for a smooth 50 frames-per-second output and a *Playback Speed* of 1/2x you should set the *view frame rate* to 100 before rendering to get a 1:1 ratio.

The quality of the video can be adjusted using the *Quality* slider. The four settings available are *Low*, *Default*, *High* and *Lossless*.<sup>5</sup> For lower settings, the file sizes of the generated videos will be smaller, but more compression artifacts will emerge. The highest-quality setting *Lossless* will result in an output video free of compression artifacts, but the file size may grow drastically.<sup>6</sup> You can define whether PreonLab should *Open the Output Folder* in a file explorer to be able to inspect the result immediately. You can also choose whether you want to monitor the progress of the video generation actively, or whether you just want to *Run in Background* and continue to work with PreonLab while the video is being produced. Using the *Copy current FFmpeg command to clipboard* button, you can copy the command that PreonLab uses to produce the video. This command can be executed in a terminal and help you if you want to automate the video creation using a script.

### 17.12.1 Best practices

Please note that if you keyframe the *view frame rate*, you have to export the rendering results piecewise. This is due to the interplay of *view frame rate*, *Playback Speed* and *Video Framerate* described above. If keyframes are detected for the *view frame rate* property, *Start Frame* and *End Frame* are set to the keyframed time interval which includes the currently selected time. For exporting another portion of the simulation with a different *view frame rate*, select one of the frames within the respective interval from the timeline.

Furthermore, the images exported from sensors might have a very small resolution

---

<sup>5</sup>These settings are realized using the CRF (Constant Rate Factor) parameter of FFmpeg. The default value is 23, while the low-quality mode uses a CRF of 31, and the high quality mode a CRF of 15. The lossless mode sets the CRF parameter to 0.

<sup>6</sup>For playing videos generated with the *Lossless* mode, a media player implementing the H.264 "High 4:4:4" encoding profile is required. Currently, the freely-available VLC is recommended to play such files. The default media player shipped with Microsoft Windows is known not to support this profile.

due to the chosen particle spacing and sample size of the sensor. This might lead to several video players not playing back the video.

## 18 PreonCLI

PreonCLI is the command line version of PreonLab. PreonCLI can be started and run from the command line without requiring a graphical user interface. This allows to run Python scripts or simulate a scene from the command line.

### 18.1 Simulating a scene

You can simulate a scene with PreonCLI by just providing the scene file (which needs to have a *.prscene* file ending) as a start parameter. The scene will then be simulated from the start to the end time defined in the scene properties. You can also optionally provide two additional parameters, the start and end for the simulation, either in simulation times or frame numbers. If you provide these additional parameters, the scene will only be simulated in the given range.

Example: `PreonCLI example.prscene`

or using simulation times: `PreonCLI example.prscene 0s 2s:100ms`

or using frames: `PreonCLI example.prscene --frames 0 40`

### 18.2 Environment variables

PreonCLI considers the same environment variables as variables as PreonNode, PreonPy and PreonLab. This includes OpenMP related variables like `OMP_NUM_THREADS` but also licensing related variables like `fifty2_LICENSE`. Checkout the chapters on OpenMP (section 2.3) and licensing (section 2.2.1) for further information.

### 18.3 Running a Python script

You can provide a single Python script file as a start parameter to PreonCLI. The file needs to end with *.py* to be detected. In the script file you can use any of the PreonPy commands which are detailed in Chapter 19.

Example: `PreonCLI example.py`

## 18.4 Optional start parameters

You can use some optional parameters when starting PreonCLI. These are listed in the table below.

Start Parameter	What it does
<code>--help</code>	Displays help information. These are also displayed when you start PreonCLI without any parameters.
<code>--version</code>	Shows the version of PreonCLI.
<code>--no-progress</code>	Disables the progress bar shown by default during simulation.
<code>--licenseMain, --lMain</code>	The main license that should be used. Possible values are <i>full</i> and <i>prepost</i> . If not specified, PreonCLI will try to checkout a suitable license.
<code>--licenseBoost, --lBoost</code>	The license extension that should be used in addition to the main license. Only meaningful if the main license has a limited number of simulation threads. Possible values are <i>mpi_unlimited</i> , <i>sim_threads_max</i> or a number that specifies the number of additional licensed simulation threads that should be checked out. <i>sim_threads_max</i> will attempt to checkout the number of threads that are required to achieve the best performance on the machine.
<code>--licenseModel, --lModel</code>	The license model that should be used for the main license. Possible values are <i>node-locked</i> or <i>n</i> for node-locked licenses and <i>floating</i> or <i>f</i> for floating licenses. If omitted, both models can be checked out.
<code>--licenseList, --lList</code>	This command prints the free and total amount of all available licenses. This includes node-locked licenses and floating licenses. It also shows the amount of included threads for main licenses (see <code>--licenseMain</code> ).
<code>--postprocess</code>	Postprocesses a given scene instead of simulating it, e.g., for rendering. Note that this always postprocesses all active sensors/renderers, i.e., previous sensor data from the simulation run may be overwritten.

**Table 118:** List of optional start parameters for PreonCLI.



## 19 Python API

The Preon solver and most of the features found in PreonLab are accessible in the Python API called PreonPy. PreonPy lets you load and save scenes, change properties, run simulations and fetch statistics. PreonPy was first introduced in PreonLab 2.0 and replaces the deprecated Python integration in PreonLab 1.x.

PreonPy is integrated into PreonLab and PreonCLI. If you want to integrate PreonLab with other Python libraries, you have to install PreonPy as a Python package into a Python installation. Otherwise you can use the Python interpreter, that comes with PreonLab and PreonCLI.

### 19.1 Supported Python version

PreonPy currently supports Python 3.5, 3.6, 3.7 and 3.8. These versions are officially supported by the Python developers. We plan to continue supporting the Python versions that are available at the time we release a particular PreonLab version.

PreonLab and PreonCLI currently ships with a Python 3.5 distribution. Note, that we might update it in future versions, not later than fall 2020 when 3.5 hits end-of-life. Note, that minor Python updates are overall backwards compatible.

### 19.2 Installation as Python package

In order to use PreonPy as a Python package, a 64-bit Python installation is required and the *pip* module has to be installed. Note, that Python 3.5 is recommended since this version is also integrated in PreonLab and PreonCLI. Using a more recent version of Python brings more features, that on the other hand are not compatible with PreonLab and PreonCLI.

#### 19.2.1 Installing Python from package manager

The easiest way to install Python is using the package manager. Please install the Python interpreter as well as *pip*. Linux distributions usually offer them as different packages. For RHEL these packages are called *python3* and *python3-pip*. You can install them via:

- `yum install -y python3 python3-pip`

Note that RHEL 6 does not include Python 3 in the main repositories, so the safest way to install it is to compile it yourself. Please refer to Section 19.2.2 for this. Also note that on some Linux distributions you may have to enable the extra packages (EPEL) in order to install *python3* and/or *python3-pip*.

- `yum install -y epel-release`

## 19.2.2 Installing Python from source

Python can usually be built very easily on many Linux distributions. Especially on CentOS/RHEL 6, the safest way to get a Python 3.5 installation is to build it from source and install it alongside the default system version. This can also make sense for e.g. CentOS/RHEL 7 systems if no other repositories should be added. The following instructions show how to compile Python 3.5 from source and install it to a custom location. This way, the impact on the system is minimized.

First install the prerequisites. This requires administrative rights, but only installs packages from the main repository.

- `yum groupinstall -y 'development tools'`
- `yum install -y zlib-devel bzip2-devel openssl-devel ncurses-devel sqlite-devel readline-devel tk-devel gdbm-devel db4-devel libpcap-devel xz-devel expat-devel`

From now on, no administrative rights are necessary anymore. First, download and unpack the Python source code, e.g., from:

- `wget https://www.python.org/ftp/python/3.5.7/Python-3.5.7.tgz`
- `tar xzf Python-3.5.7.tgz`

Then compile and install it. Replace the `/home/username/Python35` to a location of your choice. Note, that if you want to install it to `/opt` or `/usr/local` in order to let other users of your system use it easily, you have to run *make altinstall* with administrative rights.

- `cd Python-3.5.7`
- `./configure --prefix=/home/username/Python35`
- `make`
- `make altinstall`

### 19.2.3 Installing Python on Windows

On Windows, simply download the Python 3.5 distribution from <https://www.python.org/>. Make sure to download the 64 bit version, since the 32 bit version is presented as the default.

You also need to install the *Microsoft Visual C++ Redistributable 2015 64 bit*. This will also be installed with PreonLab, so you only have to care for it, if you solely want to install *preonpy*. The installer can be downloaded from the Microsoft homepage.

### 19.2.4 Installing PreonPy

*pip* is used to install the PreonPy package into a Python environment. Starting with PreonLab 3.1, the preferred way of installing is by using the FIFTY2 package repository. It is still possible to manually download and install the *.whl* files. The benefit from using the repository is that it is easier to update to a newer PreonPy version and that the right Python version and operating system is automatically selected.

- `pip install --extra-index-url=https://python.fifty2.eu preonpy`
- `pip install preonpy-....whl`

You can select a specific version by changing *preonpy* for example to *preonpy==3.1*. If no version is given, the newest available version is installed. Passing *--upgrade* updates the currently installed package if a newer version is available.

The `--extra-index-url=https://python.fifty2.eu` part of each command can be omitted if you specify `extra-index-url = https://python.fifty2.eu` in the *[global]* section of the pip config file (see [https://pip.pypa.io/en/stable/user\\_guide/#config-file](https://pip.pypa.io/en/stable/user_guide/#config-file) for more information).

You have to use *pip* of the Python installation you want to use. For example, if you followed the instructions from Section 19.2.2 on a Linux machine, use:

- `/home/username/Python35/bin/pip install ...`

On Windows, you can refer to your *pip* executable in one of the following two manners:

- `C:\Python35\python.exe -m pip install ...`
- `py -3 -m pip install ...`

Note, that the *six* and *progressbar2* modules are also installed with this command. The *progressbar2* module is optional, but installing it enables better feedback for long running tasks. You can uninstall it with `pip uninstall progressbar2` if you want to.

Also note, that the file name of the *.whl* file has to follow a naming scheme. Changing the filename will preclude the installation.

### 19.2.5 Installing multiple versions

The standard way of installing multiple versions of a certain package in Python is to use virtual environments (*venvs*). If you need to have multiple versions of PreonPy installed we recommend this technique. The supported Python versions come with this tool under the name *venv*. Note, that under some Linux distributions you may have to install it as a separate package.

The general procedure is the following. First, you create a *venv* at a location of your choice:

- `python -m venv path/to/location`

(Note that *python* refers to the Python version you want to use and should be replaced with e.g. *python3*, *path/to/python* or *py -3* as appropriate.)

Then you have to activate the *venv*. This step depends on the platform and the shell you're using. On Windows you have to execute "*path\to\location\Scripts\activate*". For most Linux systems you can either use "*source path/to/location/bin/activate*" or "*path/to/location/bin/activate*".

Inside such a virtual environment, you can install packages like PreonPy with *pip* but without affecting the system's Python installation. If you start Python from here, you can use PreonPy and all packages you have installed in this environment. You can return to your normal system environment by executing "*deactivate*".

## 19.3 Usage

After PreonPy is installed into your Python environment, you can import it with the following line:

```
import preonpy
```

The same holds if you use PreonPy with PreonCLI. (Note that the PreonPy module is already imported in PreonLab, so importing it is not necessary, but also does no harm.)

If no license file is installed, this import statement raises an *ImportError*. Install the license either manually or with PreonLab (see Section 2.2).

The PreonPy API is explained in detail with a few short examples at:

<http://fifty2.eu/PreonLab/preonpy-4.2/>

### 19.3.1 Error Handling

In case of unexpected behavior, please always check the Preon log for warnings and errors. The log is printed in the Python console and is stored in the Preon log file. The location of the log file is printed in the help dialog of PreonLab or can be obtained in a Python script with the following statement:

```
print(preonpy.get_logfile_path())
```

## 20 Distributed computing using MPI

Using PreonNode, a simulation can be computed on multiple machines which increases the performance in comparison to simulating it on a single machine. For the communication between the respective nodes in the cluster, PreonNode uses MPI. In the following we describe how to install and use PreonNode for distributed simulation. PreonNode runs on Linux-based and Windows-based machines. In the following, further instructions are given to setup and use PreonNode on Linux-based systems. If you want to use PreonNode on a Windows-based system and have questions regarding setup or use, please contact FIFTY2 or AVL support.

### 20.1 Installation

#### 20.1.1 MPI

PreonNode requires an existing and functioning MPI installation. If you already have one installed you can skip this section and continue with installing PreonNode. If not, we recommend using MPICH, which is freely available. The following is the typical installation procedure, where *<mpiuser>* is the user on every machine in the cluster:

- `wget http://www.mpich.org/static/downloads/3.2/mpich-3.2.tar.gz`
- `tar -xzf mpich-3.2.tar.gz`
- `mkdir mpich-3.2-build`
- `cd mpich-3.2-build`
- `../mpich-3.2/configure --prefix=/home/<mpiuser>/mpich-3.2-install --disable-fortran --disable-cxx`
- `make`
- `make install`

After the installation on every machine you should be able to test the installation by running an example program that calculates  $\pi$  on the cluster.

- `export PATH=/home/<mpiuser>/mpich-3.2-install/bin/:$PATH`

- `mpiexec --hosts 192.168.1.1,192.168.1.2 mpich-3.2-build/examples/cpi`

(The IP addresses have to be replaced by the ones of the machines in you cluster.)

If this does not work, typical error sources are SSH and firewall. SSH has to be configured so that it has to be possible to connect from every node to every other node without a password (using public key authentication). If SSH works, there might be a problem with the firewall blocking the MPI communication. It should either be turned off or configured in a way so that the nodes can reach each other. It is also possible to tell MPI which ports to use.

Please contact FIFTY2 support if you can't get MPI working in your environment. Note that we may not be able to fully support every aspect of your infrastructure.

### 20.1.2 PreonNode

The application that runs on every node is called PreonNode. It can be downloaded from our download page as a zip file. After unzipping it, the file *preonmpiadapter.sh* has to be executed, to adapt PreonNode to your MPI installation. For this, the MPI installation path has to be added to the *PATH* environment variable if it is not already the case.

- `export PATH=/home/<mpiuser>/mpich-3.2-install/bin/:$PATH`
- `unzip PreonNode_Linux_v4_2_4.zip`
- `cd PreonNode_Linux_v4_2_4`
- `./preonmpiadapter.sh`

Note that in order to load scenes and read/write simulation data a common network share has to be available on all machines under the same path.

## 20.2 Usage

PreonNode accepts similar parameters as PreonCLI, with the difference, that Python is not supported.

Example: `PreonNode example.prscene`

or using simulation times: `PreonNode example.prscene 0s 2s:100ms`

or using frames: `PreonNode example.prscene --frames 0 40`

With MPI, you start it indirectly by calling `mpiexec`

Example: `mpiexec <mpiexec arguments> <application> <application arguments>`

Together this results in the following example script.

- `export PATH=/home/<mpiuser>/mpich-3.2-install/bin/:$PATH`
- `cd PreonNode_Linux_v4_2_4`
- `mpiexec --hosts 192.168.1.1,192.168.1.2 ./PreonNode /share/example.prscene 0s 2s:100ms`

This will load the example scene and simulate it starting from 0s to 2s and 100ms simulation time. Thereby, the simulation is performed on the two given hosts. Please note that while only two instances of PreonNode are launched, this does not mean that the simulation is only performed with two CPU cores. Both PreonNode instances will use local threading (using OpenMP) to fully utilize their respective host machine.

## 20.3 Optional start parameters

You can use some optional parameters when starting PreonNode. These are listed in the table below.

Start Parameter	What it does
<code>--help</code>	Displays help information. These are also displayed when you start PreonNode without any parameters.
<code>--version</code>	Shows the version of PreonNode.
<code>--licenseMain, --lMain</code>	The main license that should be used. Possible values are <i>full</i> and <i>prepost</i> . If not specified, PreonNode will try to checkout a suitable license.
<code>--licenseBoost, --lBoost</code>	The license extension that should be used in addition to the main license. Possible values are <i>mpi_unlimited</i> , <i>sim_threads_max</i> or a number that specifies the number of additional licensed simulation threads that should be checked out. <i>sim_threads_max</i> will try to checkout as many threads as are required to fully utilize all CPU cores on all machines. If not enough thread licenses are available, a warning is printed and some nodes will run with a limited number of threads.
<code>--licenseModel, --lModel</code>	The license model that should be used for the main license. Possible values are <i>node-locked</i> or <i>n</i> for node-locked licenses and <i>floating</i> or <i>f</i> for floating licenses. If omitted, both models can be checked out.
<code>--licenseNodeIndividual, --lNode</code>	If this flag is passed, each node checks out a license on its own. Use it for example, if you have a separate full license for each node.
<code>--licenseList, --lList</code>	This command prints the free and total amount of all available licenses. This includes node-locked licenses and floating licenses. It also shows the amount of included threads for main licenses (see <code>--licenseMain</code> ).



<code>--timeout</code>	Timeout in seconds after which a worker node waiting on a message from other workers will terminate, aborting the entire simulation. The default is 900 (15 minutes). It might be necessary to increase this for a scene that takes very long to load.
<code>--busyWaiting</code>	Enables active waiting which increases CPU load during wait times. This can improve performance on some machines, but it is highly system dependent. It is even possible that performance degrades. Never use this on machines that also run other tasks beside PreonNode and only use this if you observe a benefit.
<code>--noPerformanceChecks</code>	Disables performance checks executed at startup that detect typical bad configurations like running PreonNode multiple times on the same machine, low network speed or using not enough threads.
<code>--allowSceneSpecificThreads</code>	Unless this option is specified, the <b>individual threads</b> property in the scene settings is ignored.
<code>--testThreads</code>	Allows to launch PreonNode without a given scene file and instead just prints the total number of simulation threads that are available. Useful to test the MPI configuration before starting a simulation.
<code>--debugLogging</code>	Enables more log messages intended for debugging.
<code>--particleBasedLoadBalancing</code>	Enables load-balancing based on the number of fluid particles. This often results in inferior performance and is intended for debugging technical problems only.
<code>--postprocess</code>	Postprocesses a given scene instead of simulating it, e.g., for rendering. This only works if PreonNode is used on a single MPI node and aborts otherwise. Note that this always postprocesses all active sensors/renderers, i.e., previous sensor data from the simulation run may be overwritten.

**Table 119:** List of optional start parameters for PreonNode.

## 20.4 Environment variables

PreonNode considers the same environment variables as PreonCLI, PreonPy and PreonLab. This includes OpenMP related variables like `OMP_NUM_THREADS` but also licensing related variables like `fifty2_LICENSE`. Check out the chapters on threading (section 2.3) and licensing (section 2.2.1) for further information. Please note that on some MPI implementations, environment variables are not always forwarded to the PreonNode instances on the host machines. Please check the documentation of your MPI implementation for further information regarding the handling of environment variables.

## 20.5 Performance guide

Given a sufficient number of worker nodes, MPI simulations can run an order of magnitude faster than simulations on a single machine. However, not every case can be accelerated using MPI and for every case there is a limit of worker nodes above which the performance will not scale. There are two reasons for this: First of all, distributed computation via MPI introduces an overhead compared to simulations that run on a single machine because of the additional synchronization effort. Secondly, there are some tasks like frame saving that can not be distributed efficiently and therefore can't scale with multiple machines. Here are some guidelines that should be considered when setting up a MPI simulation:

### 20.5.1 Do not create one process per core

Only create one process for each machine / node. Each process will utilize all available CPU cores on the machine automatically (or as many cores as specified in the scene). Spawning multiple processes per node can seriously harm the performance. By default, PreonNode will refuse to start if it detects that multiple processes are running on the same machine. Pass the argument `--noPerformanceChecks` if you want to allow it anyway for some reason.

### 20.5.2 Particle workload requirements

Each thread should manage at least 10 000 fluid particles (this is in fact not even MPI specific). Let's assume that you have a cluster in which each machine has 24 threads and you want to simulate a scene with one million fluid particles. In this case, using more than 4 machines probably won't give you a significant speedup because 5 machines have 120 threads, which results in less than 10000 fluid particles per thread.

### 20.5.3 Reduce post-processing effort

MPI can't achieve a speedup if the majority of the time is spent in post-processing instead of simulation, as only the simulation is parallelized with MPI. To see the impact of post-processing on the performance, open a previously simulated scene with PreonLab, select the scene in the scene inspector and click on **Plots** in the toolbar. Compare the statistics *Total Update (s)* and *Total Update Sim (s)*. *Total Update (s)* is the overall computation time including post-processing, while *Total Update Sim (s)* is the time required by the simulation. If the timings differ significantly, post-processing may be a bottleneck. In this case, consider reducing the simulation and view frame rate. It can also help to deactivate sensors during the MPI simulation (you can re-enable them later when you post-process the results on your local workstation).

## 20.5.4 Network requirements

If your network has a transfer speed of only 1 Gbit/s, each node should manage at least half a million fluid particles so that synchronization costs are amortized. If your network has less than 1 Gbit/s, don't bother with MPI. Before each run, PreonNode will run a simple network transfer speed benchmark and report the results in the application output and in the log file. If the transfer speed is unexpectedly low (like 100 MB/s in an InfiniBand cluster), it is possible that the wrong network interface is used. We have collected some information on this topic in chapter 20.6.

## 20.5.5 Multi-socket systems

On multi-socket systems, it can be important to specify OpenMP thread affinity environment variables to get the best performance. See section 2.3.2 for more details.

## 20.5.6 Heterogeneous clusters

It is possible to use machines with different hardware configurations for a single simulation. PreonNode will automatically adjust the workload according to the capabilities of the machine. However, please note that additional tasks like post-processing are always performed by the first worker only, and therefore this machine should be as powerful as possible. If you pass a list of hosts to `mpiexec` / `mpirun`, make sure that the first host is the most powerful machine.

## 20.5.7 Maximum number of nodes

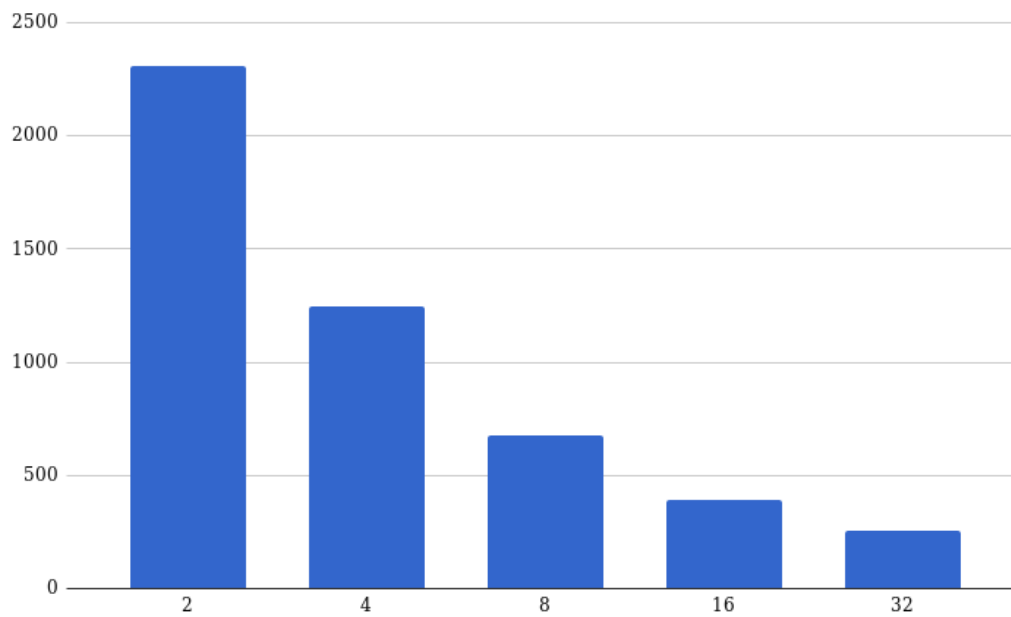
We have tested MPI simulations with up to 32 machines / nodes. In theory, more nodes are possible, but please be aware that you are in uncharted territory if you use more than that.

## 20.5.8 Scaling

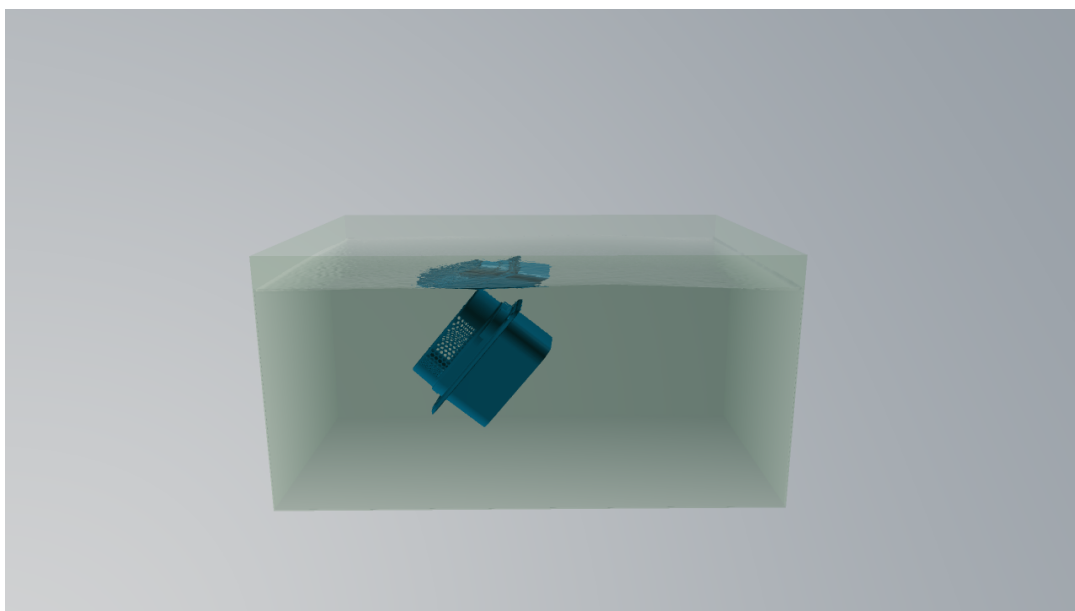
As mentioned, it is impossible to predict the performance scaling for every kind of scene. As a reference point, Figure 66 shows the scaling behavior for a scene with 19 million particles in which a moving and rotating geometry dives into relatively deep water. The scene is illustrated in Figure 67. Feel free to download the scene<sup>1</sup> in order to run the benchmark on your cluster.

---

<sup>1</sup>Download MPI scaling benchmark *mpi\_dive\_portable.prscene* from <https://transfer.fifty2.eu/index.php/s/bhHEW07DSgTvt8f>



**Figure 66:** Total update times in seconds for a specified frame range of the dip coating benchmark. The tests were conducted on the High-Performance Computing Center Suttgart for 2, 4, 8, 16 and 32 nodes. Thereby, each node has 16 CPU cores.



**Figure 67:** Example frame of the dip coating scene used for the MPI scaling benchmark.

## 20.6 Best practices for typical environments

### 20.6.1 Running PreonNode with Open MPI

When using the Open MPI implementation, a few things must be considered in order to start PreonNode correctly.

- By default, most Open MPI environments are configured to launch one CPU thread per process. As explained earlier, this is not the right strategy for PreonNode. Instead, one process per machine should be spawned. Pass `--bind-to board` to the `mpirun` command to ensure that the PreonNode instance has access to all cores of the machine.
- Make sure that the environment variable `OMPI_MCA_mpi_leave_pinned` is not set to 1 by setting it to 0 explicitly. For instance, this can be achieved with `export OMPI_MCA_mpi_leave_pinned=0`
- When using InfiniBand, make sure to load the required MCA plugins. Read the Open MPI documentation for more information.

### 20.6.2 Running PreonNode on IBM Platform LSF

When using a queuing system to run MPI jobs, it can be challenging to launch PreonNode correctly because most systems are optimized for pure MPI applications and not for hybrid MPI/OpenMP applications like PreonNode. For IBM Platform LSF, the following command can be used to schedule PreonNode jobs:

```
bsub -n 4 -x -R "span[ptile=1]" "path_to_mpiexec path_to_scene_file"
```

- `-n 4` means that 4 instances of PreonNode will be launched. As discussed earlier, this does not mean that the computation is only performed with 4 CPU cores because every PreonNode instance uses local threading.
- `-x` ensures that if PreonNode is running on a machine, no other task will be launched on this machine by the queuing system. This is necessary because each PreonNode instance will use more than one CPU core and consequently use much more resources than the queuing system expects. Without the `-x`, the queuing might launch other jobs on the CPU cores which it thinks are free, interfering with the PreonNode job.
- `-R "span[ptile=1]"` ensures that only one instance of PreonNode is launched per machine, otherwise the system may launch all 4 instances on the same machine.

### 20.6.3 Running PreonNode with MPICH

When using InfiniBand, it may be necessary to specify the correct network interface using the `-iface` option to get good transfer speeds (this is however also the case for other applications, and not just PreonNode).

## 20.7 Known limitations

PreonNode does not support all features for MPI simulations yet. Known limitations include:

- Only a subset of sensors supports post-processing at simulation substeps. This includes the sensor plane, the sensor mesh and the force sensor. Other sensors will only perform post-processing at whole view frames. Also, the force sensor can only track maximum per-particle pressures at whole frames. For optimal performance, we recommend to disable post-processing of simulation substeps entirely in the scene settings.
- Rigid body simulations and two-way coupling with dynamic solid objects are not supported when using the Bullet solver.