PreonLab 5.2.3

Manual

Get support via:
🌍 http://help.preonlab.eu/
✉ support@fifty2.eu
☎ +49 (0) 761 45 89 23 – 81
Mo-Fr: 9am-5pm (CET)

August, 2022

Contents

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 are 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 7.0 and higher versions.

Typically, all required libraries should be available for Linux installations with a graphical desktop. In some cases you might have to install one or more of the following *xcb* libraries and extensions (they might be named differently on other Linux distributions): *libxcb*, *libxkbcommon-x11*, *xcb-util-wm*, *xcb-util-image*, *xcb-util-keysyms*, *xcb-util-renderutil*. Additionally, PreonLab requires OpenGL libraries. See Section 2.1.4

for more information regarding systems without graphics.

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

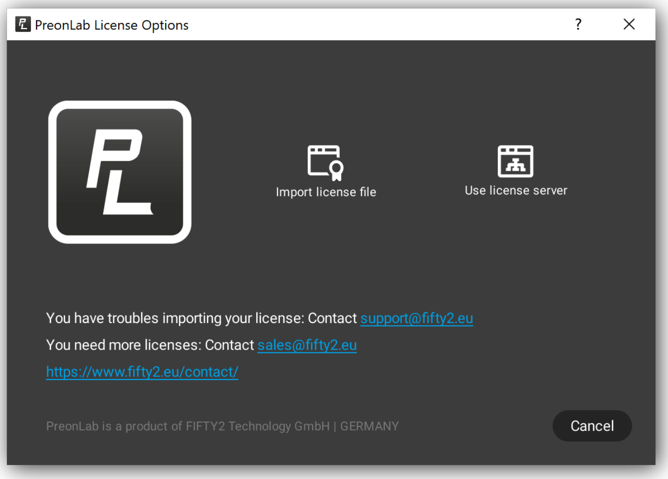
The Python API PreonPy is integrated into PreonLab and PreonCLI, but it is also available as a regular Python package. See Section 19.2.4 for further instructions on how to install the Python package into an existing Python 3 installation. Also see Chapter 18 for information about PreonCLI and Section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

PreonLab and PreonCLI both come with a Python 3.8 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).



You have troubles importing your license: Contact support@fifty2.eu

You need more licenses: Contact sales@fifty2.eu

https://www.fifty2.eu/contact/

PreonLab is a product of FIFTY2 Technology GmbH | GERMANY

Figure 1: Licensing dialog.

2.2 Licensing

When you start PreonLab for the first time, you will be asked for a valid product license in the dialog shown in 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 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:

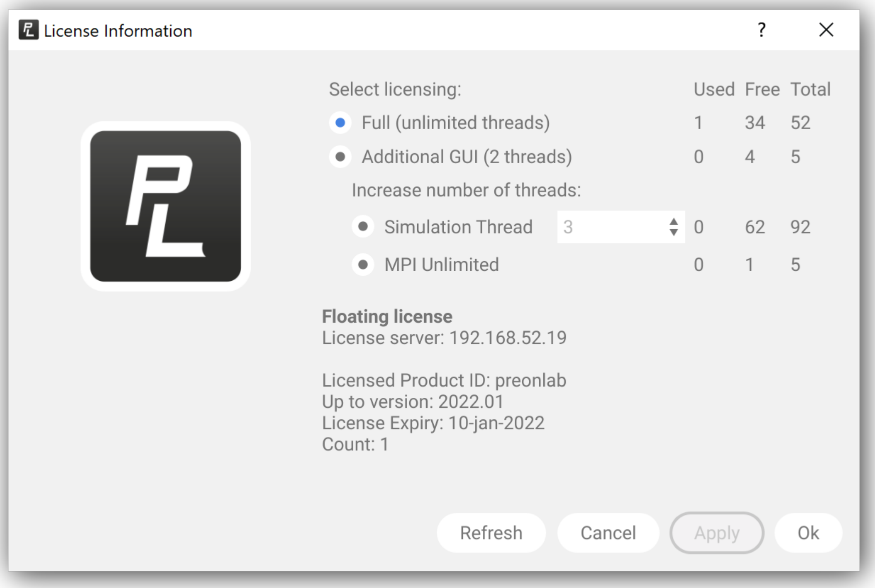
Full license: This license contains all components of PreonLab.

Prepost license: The prepost license is used for scene setup and postprocessing tasks. You can also simulate, but only with two simulation threads. Postprocessing such as rendering runs with all available 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 single machine.

For distributed simulations with PreonNode a *full* license 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. Alternatively, multiple *full* licenses can be used to license each node on its own in a single distributed simulation. Each node can use unlimited threads in this case.

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.



Select licensing:		Used	Free	Total
● Full (unlimited threads)		1	34	52
○ Additional GUI (2 threads)		0	4	5
Increase number of threads:				
○ Simulation Thread	3	0	62	92
○ MPI Unlimited		0	1	5

Floating license
License server: 192.168.52.19

Licensed Product ID: preonlab
Up to version: 2022.01
License Expiry: 10-jan-2022
Count: 1

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 broadcast in your network. Reprise provides an extensive manual for license administrators 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 also place multiple license files in the above-mentioned directory. All files with a `.lic` file ending will be considered. This can be useful if you have a node-locked license, but also want to be able to checkout additional floating licenses. Another typical use case is to point to a primary and a failover license server, such that the failover license server can take over without reconfiguring the clients. As a final example, consider the case where you purchase additional licenses and want to simply add them alongside existing ones.

Autodiscovery of license servers via broadcast uses the first license server that responds. If you are using a failover setup, it is recommended to manually specify both license servers as described above, instead of relying on autodiscovery. Otherwise, PreonLab could first detect the failover server and doesn't recognize the regular one anymore, resulting in no license being available. If both servers are manually specified, both will be checked for licenses and so the regular and the failover case are covered.

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 (it ends with `.lic` and contains lines that start with `LICENSE fifty2 preonlab`) 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.

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 Installation of a failover license server

1. After installing the primary license server according to Section 2.2.2, perform the same step again for the failover license server. (Important: Also put the licenses that were issued for the primary license server to the failover license server.)
2. Edit the failover license file (it ends with `.lic` and contains a line that starts with `LICENSE fifty2 rlm_failover`). Verify the hostname of the primary server in the `_primary_server=` field and change it if it does not fit your setup. Put the hostname of the failover license server on the HOST line (replace ”failover_host” such that it reads `HOST <your host name> <your mac address>`).
3. Also put this failover license file on the failover license server next to the other license files. Afterwards, the server has to be (re-)started. Now, it will wait for the primary server to disappear and will take over to serve licenses in such a situation.

As soon as the primary server disappears, PreonLab has to use the failover license server. On a computer inside your LAN, PreonLab should automatically find the license server. If this does not work, you may have to remove the license file that points to the primary license server and enter the address of the failover license server when prompted. Please read Section 2.2.1 for further advice.

2.2.4 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 four 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**. **RESERVE** can be used to set aside a certain amount of licenses for specified users or machines. 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 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 cluster machines consisting of nodes *compute1*, *compute2* and *compute3* have access to the three *full* licenses, and two of the three licenses are reserved to be used only by it. The remaining one *full* license can be used by the user *john*. Additionally, 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 host_group cluster
RESERVE 2 preonlab host_group cluster
INCLUDE preonlab user john
MAX 1 preonlab_prepost internet 192.168.26.26
```

The following is an excerpt of the corresponding license file.

```
LICENSE fifty2 preonlab 2020.06 20-jun-2020 3 …
LICENSE fifty2 preonlab_prepost 2019.12 20-dec-2019 3 …
```

2.2.5 Troubleshooting

If you are not able to checkout your license, first check if your licensing configuration is correct. You can start PreonCLI as well as PreonNodewith the `--licenseList` argument. It will show you which licenses can be accessed. If you see that licenses exist but are already checked out by someone else, it is possible to look at the web interface of the RLM license server. By default, it can be accessed by `http://server:5054/home.asp` where `server` is the name of the host the RLM license server runs on.

If you don't see all licenses listed that you are expecting, you can look at how the RLM licensing client inside of PreonLab, PreonCLI or PreonNode tries to find licenses using the `RLM_DIAGNOSTICS` environment variable. Please also checkout `https://www.reprisesoftware.com/RLM_Troubleshooting_Tips.pdf` for a guide on how to investigate typical licensing related issues and `https://www.reprisesoftware.com/RLM_License_Administration.pdf` for more detailed information on how how RLM works and how it can be configured.

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 Microsoft's 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 Known limitations

Serving RLM in a container might not work reliably due to an issue in our licensing library. In any case you have to remove, or rather not copy, the vendor binary called *fifty2* and only use the set-file *fifty2.set* in order to let RLM recognize the hardware identification. Still it might crash shortly afterwards.

2.3.2 Setting the number of threads

You can specify the number of used threads for PreonCLI, PreonNode and the PreonPy Python package, 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. `OMP_NUM_THREADS` is not considered at all for PreonLab.

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.

2.4 Logging

Information about the status and (possible) problems of the PREON® applications is written to the message window for PreonLab and for the non-graphical variants to the console output. In addition to that, it is written to the logfile (split up depending on the severity into **Info**, **Warning** and **Error**). You will find the location of the logfile in the about dialog or via PreonPy with a call to `preonpy.get_logfile_path()`.

The default base path for all applications but PreonNode is `C:/Users/<USER>/AppData/Local/PreonLab/Logs` (Windows) or `/.config/PreonLab/Logs` (Linux). As PreonNode runs on different nodes, its logfiles are stored in the scene directory (`<scene dir>/MPILogs`).

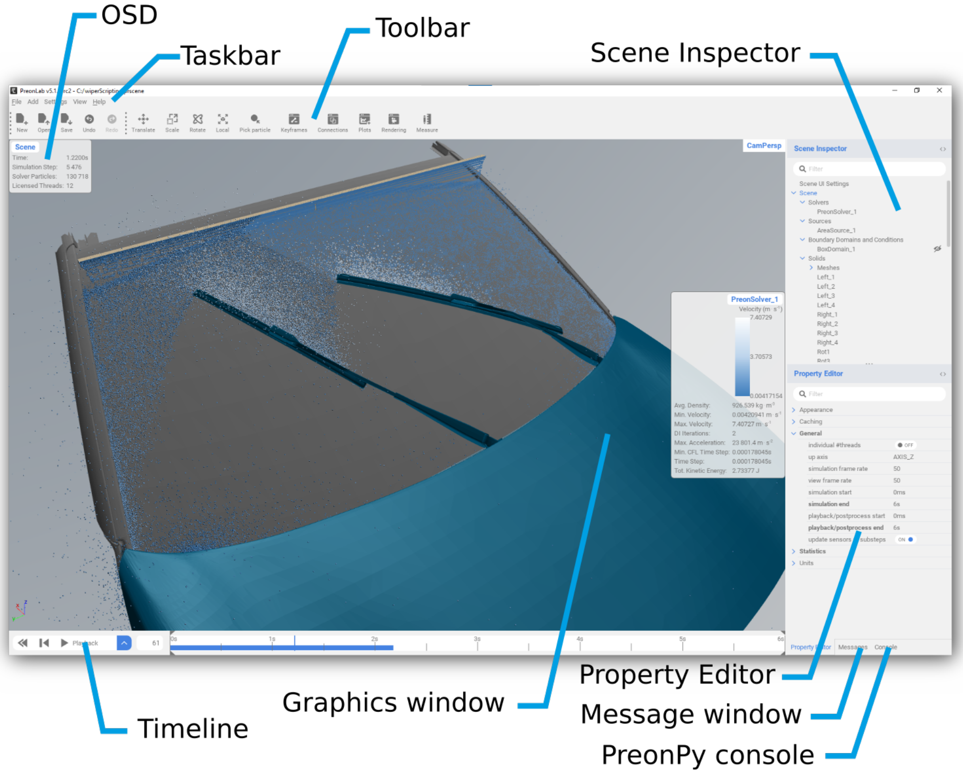
It is possible to set the default logging directory with the `--logDir` start parameter of PreonLab, PreonNode PreonCLI. In PreonPy, call `preonpy.set_log_dir("/path/to/logdir")`.

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.



Scene
Time: 1.2200s
Simulation Step: 5 476
Solver Particles: 130 718
Licensed Threads: 12

PreonSolver_1
Velocity (m · s^{-1})
7.40729
3.70573
0.00417154
Avg. Density: 926.539 kg · m^{-3}
Min. Velocity: 0.00420941 m · s^{-1}
Max. Velocity: 7.40727 m · s^{-1}
DI Iterations: 2
Max. Acceleration: 23 801.4 m · s^{-2}
Min. CFL Time Step: 0.000178045s
Time Step: 0.000178045s
Tot. Kinetic Energy: 2.73377 J

individual #threads: OFF
up axis: AXIS_Z
simulation frame rate: 50
view frame rate: 50
simulation start: 0ms
simulation end: 6s
playback/postprocess start: 0ms
playback/postprocess end: 6s
update sensors / substeps: ON

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 3.13).

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 + drag 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

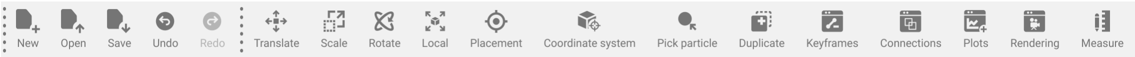


Figure 4: PreonLab toolbar. The toolbar is context sensitive and shows different options depending on the selected objects.

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's 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 3.13).

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 Section 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. 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 as well as links to the online manual and PreonPy documentation.

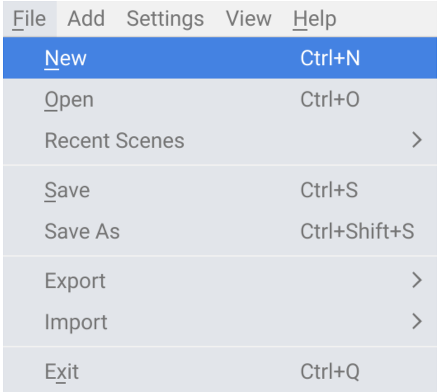


Figure 5: Taskbar.

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).

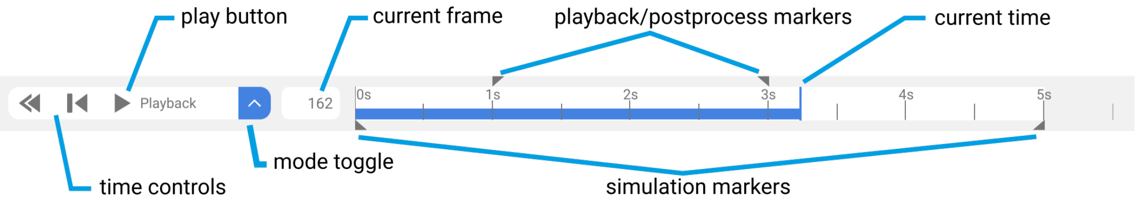


Figure 6: Timeline with controls on the left and an overview of several time-related functions.

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:4m:5s:6ms*. 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.

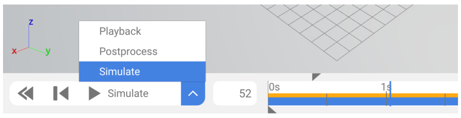


Figure 7: Clicking on blue button with white up arrow next to the play button opens a menu in which you can choose the mode: Playback, Postprocess or Simulate.

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.

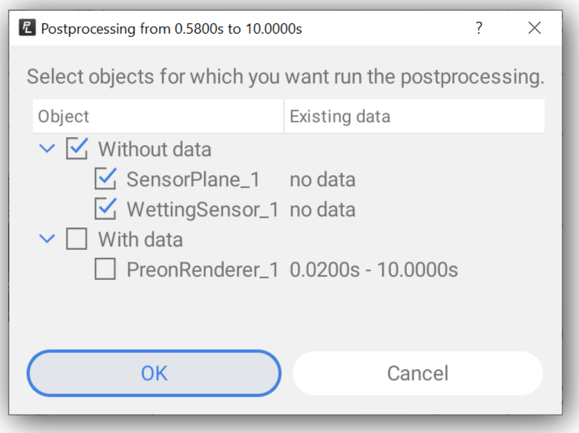
Postprocess: This mode allows you to gather data for postprocessors like sensors on already existing simulation data. When you start a postprocessing run, a dialog is shown that allows you to select which postprocessors should update their data in this postprocess run. An example for this dialog is shown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



Postprocessing from 0.5800s to 10.0000s

Select objects for which you want run the postprocessing.

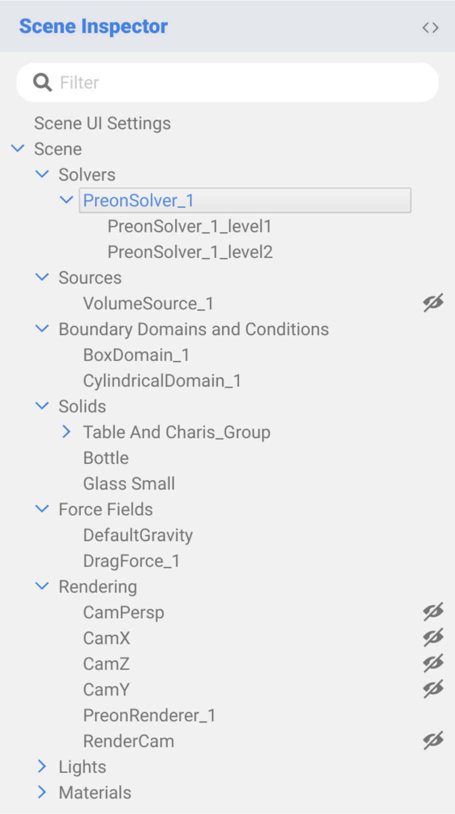
Object	Existing data
☑ Without data	
☑ SensorPlane_1	no data
☑ WettingSensor_1	no data
□ With data	
□ PreonRenderer_1	0.0200s - 10.0000s

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

tions 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's name shows the behavior of the object: active (white), inactive (red), cached (yellow). Note that regardless of its behavior, an object will be highlighted in blue when it is selected. The user can select any object by left-clicking on it. By double-clicking an object or pressing F2 when it is selected, the user enters to the name-editing mode and can rename the corresponding object. In the name-editing mode, navigation between the objects of a category is possible via the Tab key. Pressing the eye icon beside the name of an object toggles its visibility. 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.



Scene Inspector

Filter

Scene UI Settings
Scene
Solvers
PreonSolver_1
PreonSolver_1_level1
PreonSolver_1_level2
Sources
VolumeSource_1
Boundary Domains and Conditions
BoxDomain_1
CylindricalDomain_1
Solids
Table And Charis_Group
Bottle
Glass Small
Force Fields
DefaultGravity
DragForce_1
Rendering
CamPersp
CamX
CamZ
CamY
PreonRenderer_1
RenderCam
Lights
Materials

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.
left mouse button click + drag mouse + release left button	Select all objects between clicked object and object on which the mouse button was released.
CTRL + a	Select all objects in scene inspector.

Table 2: Controls for scene inspector.

At the top of the scene inspector you can find a search field (see Figure 9) where you can enter (partial) object names to quickly filter the list to only keep the objects which have the search string as (part of) their names. When searching for an object, note that the names of all groups it is contained in are considered. 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 on the right of the search field to clear the search string.

If you want to select all children of a specific object category, e.g., 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* which does exactly that.

Furthermore, the context menu provides actions to *Show/hide* objects in the graphics window, as well. 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.

3.6.1 Grouping of objects

It might be practical to group objects, especially for scenes with a great number of them. To do that, first select the objects to group, these objects need to belong to the same main category. For example **Solids** could be grouped with each other. On the other hand, a **Solid** object and a **Camera** object can not be grouped. Secondly right click on one of the selected objects, then choose *Grouping* from the context menu. The options in *Grouping* are described below.

Create new group: Clicking on *Create new group* creates a new group object. If some of the selected objects were already in a group previously, they are moved to the new group and removed from the existing one.

The advantage of grouping objects like this is the better overview in the scene inspector. The groups are also used to reduce the number of displayed objects in the connection editor (see Section 5.1.4). If the number of total objects in the scene is large, then grouping objects drastically improves the usability and the performance of the connection editor. Note that grouping does not influence behavior like translating, rotating and so on. Hence, if you translate or rotate a member of a group the other members of the group are not changed. If you would like to move several objects at once, use a transform group instead. Transform groups are explained in Section 8.3.

Remove from group: To remove objects from a group, select them, right-click on them and click on *Grouping*, then on *Remove from group*.

Move to group: In order to move objects to a group, you need to select a group as well as the objects that you would like to move there. Right-click on your selection and click on *Grouping*, then on *Move to group 'Name_of_group'*. For each group that you selected, there is one such entry. If you did not select a group, the entry *Move to group* in the context menu is grayed out.

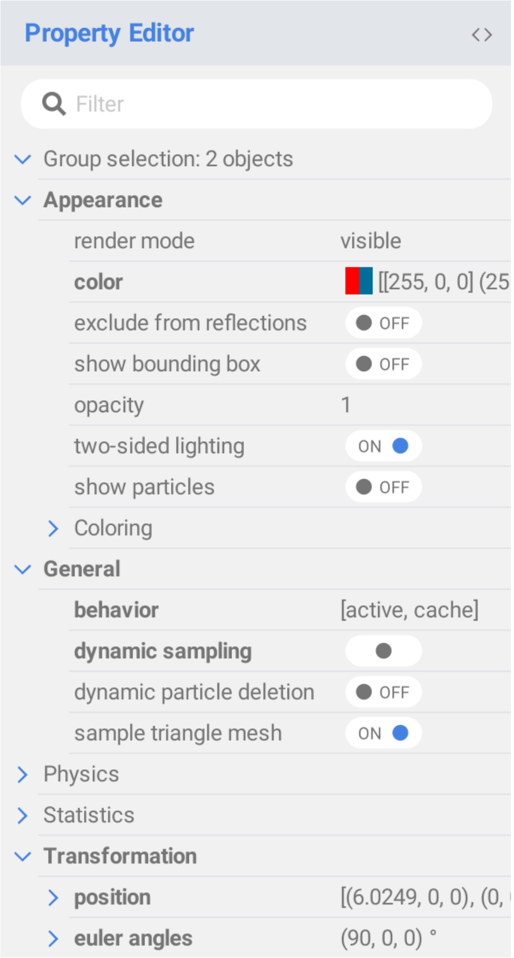
Select all children: When you right-click on a group, there is a menu item *Select all children* which shows the group name in brackets. Clicking on it selects all children, i.e., members of that group. If you hold CTRL pressed while right-clicking to open the context menu of a group, then click *Select all children*, all children of the group are added to your selection.

When a group becomes empty, by deleting all its members, or by moving all its members out, it is removed from the category and shown at the bottom of the scene inspector.

3.7 Property Editor

The property editor displays the properties of the selected object. If multiple objects are selected, the union of properties is shown. In this case, for a common property with differing values, all values are shown in array format (except for the names of the objects) as can be seen in Figure 10. The properties are grouped semantically and groups can be expanded or collapsed. Multi-dimensional properties like positions are also represented as expandable groups. The user needs to expand the group and edit each dimension individually.

At the top of the property editor, you can find a search field (see Figure 10) where you can enter (partial) property names, values or units to quickly filter the list to only keep the properties which have the search string as (part of) their names, values or units. The search is case-insensitive. Click the X icon on the right of the search field to clear the search string.



Property Editor

Filter

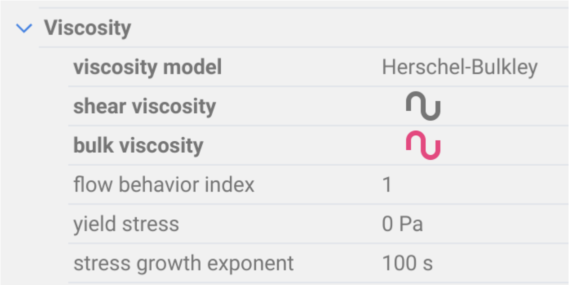
⌄ Group selection: 2 objects	
⌄ **Appearance**	
render mode	visible
color	[[255, 0, 0] (25
exclude from reflections	● OFF
show bounding box	● OFF
opacity	1
two-sided lighting	ON ●
show particles	● OFF
› Coloring	
⌄ **General**	
behavior	[active, cache]
dynamic sampling	●
dynamic particle deletion	● OFF
sample triangle mesh	ON ●
› Physics	
› Statistics	
⌄ **Transformation**	
› **position**	[(6.0249, 0, 0), (0,
› **euler angles**	(90, 0, 0) °

Figure 10: Property Editor.

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.

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. Note that in this case, the user can use the right-click action *Set key* in order to define that property value as a key. There are also properties for which key values can be set based on particle values, e.g., temperature-based viscosity. These properties are displayed as shown in Figure 11. A red color indicates that no keys are set for the property yet while the default color indicating that at least one key is set for it. Clicking the icon opens the keyframe editor for the respective property.



Viscosity	
viscosity model	Herschel-Bulkley
shear viscosity	∿
bulk viscosity	∿
flow behavior index	1
yield stress	0 Pa
stress growth exponent	100 s

Figure 11: Properties whose key values are not based on time are indicated in the property editor as seen in this figure.

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 is invisible. Clicking on *Messages* makes it visible. If the message window is not visible, a number with a colored background is added next to the tab title *Messages* for each type of aforementioned new (unread) messages (if any). (See Figure 12.) Moreover, each color corresponds to one of the three types of messages; red indicates errors, yellow indicates warnings, and the default gray indicates messages that are just informative. See 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.

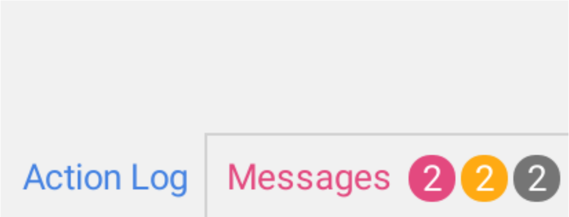
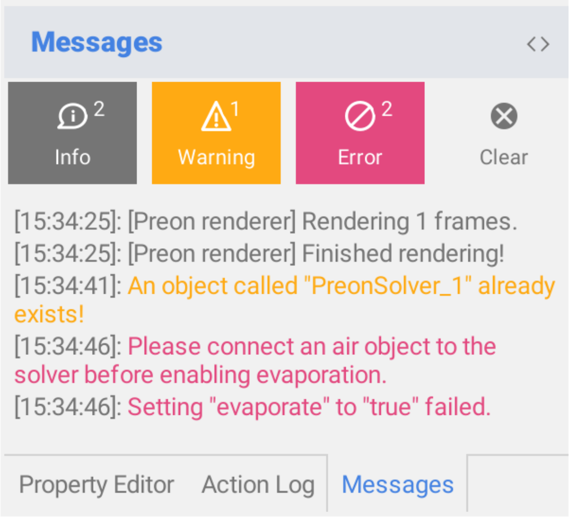


Figure 12: Message count indicates number of new (unread) messages of each type.

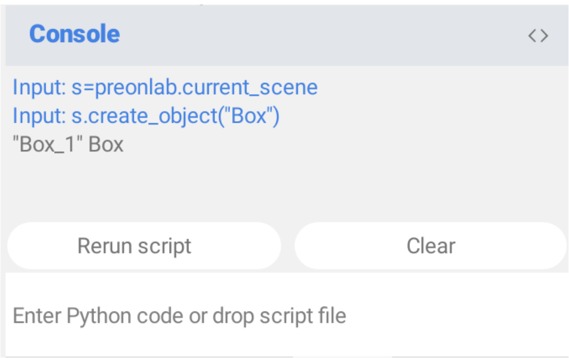


```
[15:34:25]: [Preon renderer] Rendering 1 frames.
[15:34:25]: [Preon renderer] Finished rendering!
[15:34:41]: An object called "PreonSolver_1" already exists!
[15:34:46]: Please connect an air object to the solver before enabling evaporation.
[15:34:46]: Setting "evaporate" to "true" failed.
```

Figure 13: Message window with one message of each type.

3.9 PreonLab 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.

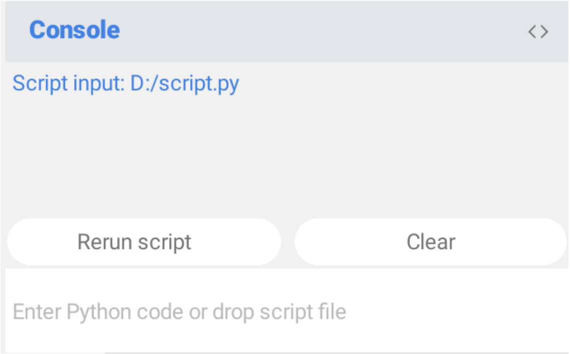


```
Input: s=preonlab.current_scene
Input: s.create_object("Box")
"Box_1" Box
```

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

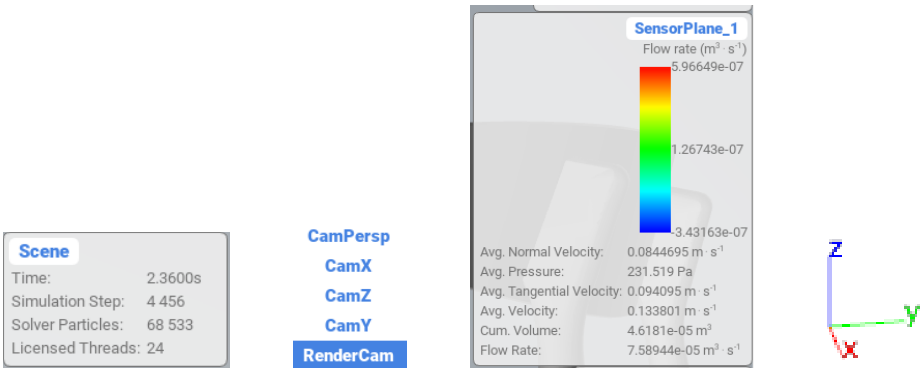
3.10 OSD (On-Screen-Display)

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



Script input: D:/script.py

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



Time: 2.3600s
Simulation Step: 4 456
Solver Particles: 68 533
Licensed Threads: 24

Avg. Normal Velocity: 0.0844695 $m \cdot s^{-1}$
Avg. Pressure: 231.519 Pa
Avg. Tangential Velocity: 0.094095 $m \cdot s^{-1}$
Avg. Velocity: 0.133801 $m \cdot s^{-1}$
Cum. Volume: 4.6181e-05 m^3
Flow Rate: 7.58944e-05 $m^3 \cdot s^{-1}$

Figure 16: OSD elements for **Scene**, camera menu, a sensor plane and the global coordinate axis.

for each object in the property editor **Statistics**→**OSD Settings**.

OSD elements are collapsed and expanded by left-clicking on the object name in the OSD element.

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 16.

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.

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

Table 3: List of keyboard shortcuts.

3.12 Units

PreonLab uses per default SI units to represent physical quantities. Depending on the scale of your simulation setup, it could be beneficial to change the units for certain quantities (e.g. length in mm). Table 4 presents you with the list of supported quantities and their corresponding unit options, where the first unit is the SI unit, and the subsequent ones are the possible alternatives we offer.

Quantity	Units
acceleration	m^2/s
angle	°; rad
angular velocity	°/s; rad/s
area	m^2; mm^2
concentration	%
damper rate	N s/m
data	B; kB; MB; GB; TB
density	kg/m^3; g/cm^3
dynamic viscosity	Pa s
energy	J; mJ
force	N; kN
frequency	1/s
heat flux	W/m^2; mW/m^2; kW/m^2
heat transfer coefficient	$W/(m^2\,K)$
isobaric mass heat capacity	J/(kg K); J/(g K)
length	m; mm; cm; km
mass	g; kg; t
moment of inertia	$kg\,m^2$
momentum	kg m/s
precipitation rate	mm/min
pressure	Pa; kPa; bar; MPa; GPa
surface tension	N/m
temperature	K; °C
thermal conductivity	W/(m K)
time	s; ms; min; h
torque	N m
velocity	m/s; mm/s; km/h
volume	m^3; mm^3; ml; l
volume flow rate	m^3/s; mm^3/s; ml/min; l/min

Table 4: List of PreonLab quantities and their possible units.

Units can be changed from different objects within PreonLab. How you can change them and how these changes related to each other is described in the corresponding sections, namely in the user preferences section (see Section 3.13), in the **Scene** object section (see Section 8.1) and in the plot dialog section (see Section 7.1.3).

Note that in the rest of the manual, we use the SI units when describing physical quantities.

3.13 User Preferences

The *User Preferences* can be found under the menu *Settings* in the task bar (see Figure 3). Here, you can switch between dark and light color mode to change the background colors of the graphical user interface. You can also define some general settings like the *maximum number of threads* used by PreonLab, enable new features which are marked as experimental. In the *User Preferences*, you can also preset the colors used for images exported from the plot dialog as described in (see Table 14). Furthermore, you can specify several preferences that are used when creating a new scene. 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 can still be modified per scene later.

Preference	What it does
dark mode	If enabled, the PreonLab user interface will use dark colors as main background colors. If disabled, the main background colors will be light, instead. Figure 17 shows a comparison of the two modes.
rendered particles target	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.
default scene directory	The directory where new scenes are created.
default up axis	The up axis (either z-axis or y-axis) used for new scenes.
show grid by default	Defines whether the orientation grid should be drawn by default in newly created scenes.
default background color	New scenes will be created with this color as the first background color.
default background color 2	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.
Units	In this section, you can change the default units. For each new scene, the units you have set in this section will be assigned to the corresponding quantities. Already existing scenes are not affected. Note that PreonLabs default convention is restored when you right-click the property section and select **Set all subproperties to their default values**.
default video export directory	For new scenes, videos will be exported to this directory by default.
maximum number of threads	The number of CPU threads used by PreonLab will never exceed this maximum number. This correlates with *individual #threads* and *#threads for this scene* in the scene properties, i.e., a lower number of threads may be specified per scene.
experimental features	If on, experimental features of PreonLab are exposed. See Section 3.15 for more details.

save scenes before simulation or post-processing	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.
control key	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 5: The user preferences to define.

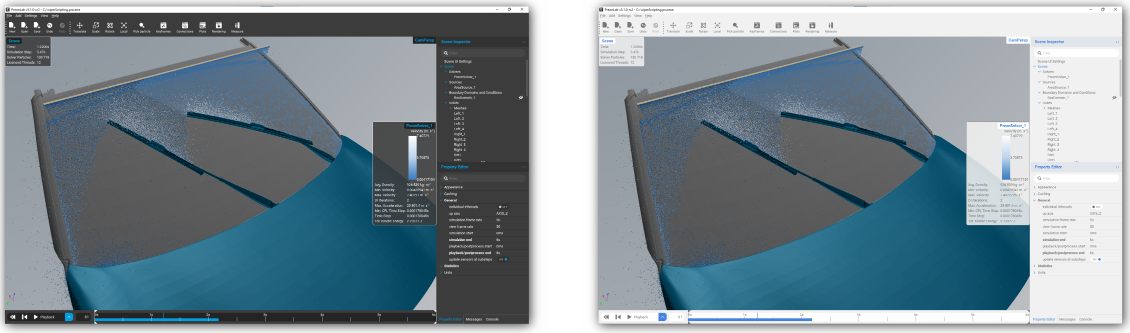


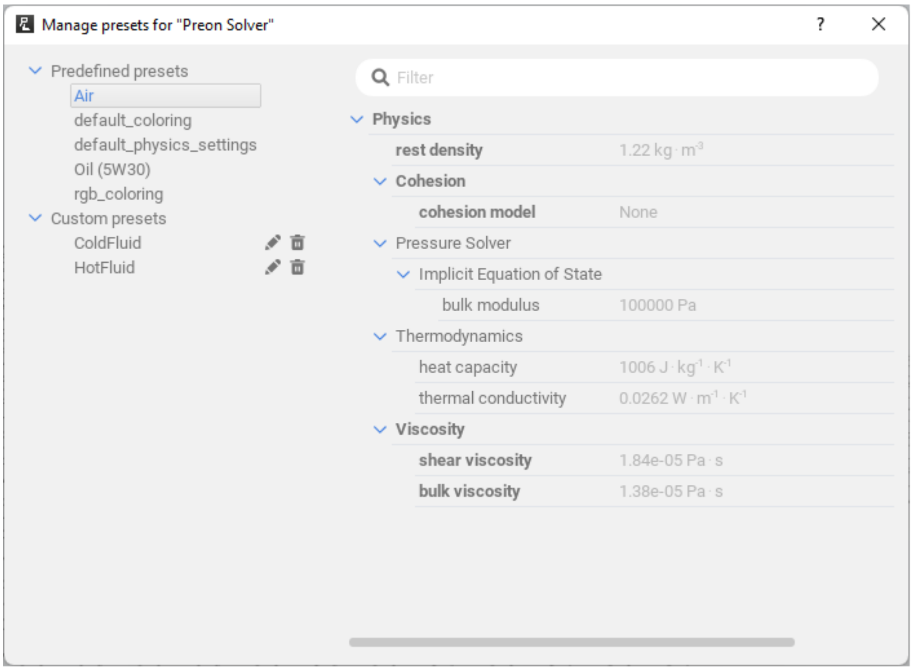
Figure 17: Comparison of the dark and light mode of PreonLab.

3.14 Presets

For some object types, predefined presets are available that contain common property configurations for the object. Examples include camera configurations, different fluid types, particle coloring schemes or material settings for rendering. To apply a preset, right-click on an object in the scene-inspector, in *Presets* select the preset you want to apply. Right-clicking in the graphics window will display the same menu for the currently selected object. Click on *Manage* in order to get a list of all available presets for the selected object types.

You can create your own presets for any object via *Create* in the context-menu group *Presets*. This will show a dialog listing all the properties for this object. Specify a preset name and select those properties that you want to be stored with the given value in your custom preset. Note that all the values are taken from the current property settings of the selected object. In order to change the value, you have to use the property editor as the *Create preset* dialog doesn't support the change of property values by design. If you have confirmed your selection with OK, your preset should be listed in the *Manage* dialog and you can apply it to any other object or selection of objects which have this type.

All presets are stored in the folder `C:/Users/<USER>/AppData/Local/PreonLab/Presets/` (Windows) or `~/.config/PreonLab/Presets` (Linux). In order to share custom presets with your colleagues, just copy the respective presets to the the other computer. Please note that you have to adhere to the folder structure.



Manage presets for "Preon Solver"

Predefined presets
- Air
- default_coloring
- default_physics_settings
- Oil (5W30)
- rgb_coloring

Custom presets
- ColdFluid
- HotFluid

Physics
- rest density: 1.22 kg · m^{-3}
- Cohesion
  - cohesion model: None
- Pressure Solver
  - Implicit Equation of State
    - bulk modulus: 100000 Pa
- Thermodynamics
  - heat capacity: 1006 J · kg^{-1} · K^{-1}
  - thermal conductivity: 0.0262 W · m^{-1} · K^{-1}
- Viscosity
  - shear viscosity: 1.84e-05 Pa · s
  - bulk viscosity: 1.38e-05 Pa · s

Figure 18: The *manage presets* dialog lists all the presets for the selected object(s). If you click on the pen icon, you can rename the preset. Clicking on the trash icon will delete the preset.

3.15 Experimental features

PreonLab includes experimental features that are only available if they are activated in the user preferences (c.f. Section 3.13). 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.

3.16 Start parameters

PreonLab can be started with optional start parameters. The available parameters are listed in Table 6.

Start parameter	What it does
--scene <filename>	Directly loads the given <*filename*> as scene. If <*filename*> is not a valid scene file, an empty untitled scene will be opened.
--noGeometryShaders	Prevents PreonLab from using geometry shaders. This may help on some Linux-based systems if various artifacts appear in the graphics view. See Section 3.17.1 for more information.
--skipglxtest	On Linux, PreonLab executes an OpenGL version test on startup using *glxinfo* to make sure the graphical system is matching the requirements. When providing this start parameter, the test is skipped.
--logDir	Sets a log directory other than the default.

Table 6: Possible start parameters for PreonLab.

3.17 Known issues and workarounds

3.17.1 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.

3.17.2 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.

3.17.3 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.

3.17.4 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).

The 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.

Enabling higher versions of OpenGL with RD

Warning: This solution only works with users that have administrative rights on the RD host computer.

First, access the group policy editor by typing "*gpedit.msc* " in the start menu on the host computer. Then navigate to the following section:

- *Computer Configuration*→*Administrative Templates*→*Windows Components*→*Remote Desktop Services*→*Remote Desktop Session Host*→*Remote Session Environment*.

Note: Double click on *Remote Session Environment* to access options to enable.

Enable the following options:

- *Use hardware graphics adapters for all Remote Desktop Services sessions*
- *Prioritize H.264/AVC 444 graphics mode for Remote Desktop Connections*
- *Configure H.264/AVC hardware encoding for Remote Desktop Connections*

Restart the host computer after enabling these options. PreonLab will now work on the host computer over RD.

4 Common properties

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

4.1 General

Property	Unit/Type	What it does
behavior	-	Controls the behavior of the object during simulation and playback. **active** means that the object is part of the simulation and will save simulation data to disk when simulating. **inactive** is the opposite, the object will be completely ignored during simulation and playback. **cache** 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 7: 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	Unit/Type	What it does
render mode	-	Changes the way the object is displayed. **visible** is the default setting and will enable smooth per-pixel lighting. **wireframe** is only available for meshes and displays the wireframe rendering of the mesh triangles. **invisible** 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.
show bounding box	On/Off	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.
opacity	-	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 **render mode** accordingly for better performance.
exclude from reflections	On/Off	If enabled, this object will not be mirrored in reflective surfaces when using Preon renderer. Mainly intended for sensors.

Table 8: Common properties in group **Appearance**.

4.3 Transformation

Property	Unit/Type	What it does
position control mode	-	Determines how the position for the object is specified. The default is **position**, which lets you specify the position directly. It is also possible to control the position using its derivatives by choosing **velocity** or **acceleration**.
position	m	The local position of the object in *x*, *y* and *z* coordinates. Only available if **position control mode** is set to **position**. 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)
global position	m	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.
velocity	m/s	The local velocity of the object. Only available if **position control mode** is set to **velocity**.
start position	m	The local position of the object at time 0. Only available if **position control mode** is set to **velocity** or **acceleration**.
acceleration	m/s^2	The local acceleration of the object. Only available if **position control mode** is set to **acceleration**.

start velocity	m/s	The local velocity of the object at time 0. Only available if **position control mode** is set to **acceleration**.
orientation control mode	-	Determines how the orientation for the object is specified. You can either enter the rotation as Euler angles (choose **eulerAngles**) or as a rotation around an axis (choose **revolution**). You can also specify rotations per second around an axis by choosing **revolutions_PerSecond**.
revolution axis	-	The local axis around which the object should rotate. Only available if **orientation control mode** is set to **revolution** or **revolutions_PerSecond**.
revolution	-	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 **orientation control mode** is set to **revolution**.
revolutions per second	1/s	Sets the revolutions per second around the chosen axis. Only available if **orientation control mode** is set to **revolutions_PerSecond**.
revolution start	-	Sets the state of the revolution around the chosen axis at time 0. Only available if **orientation control mode** is set to **revolutions_PerSecond**.
euler angles	°	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 **orientation control mode** is set to **eulerAngles**.
scale	-	The scale of the object as a three-dimensional vector.

Table 9: Properties in group **Transformation**.

Property	Unit/Type	What it does
adjust own transform	On/Off	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.
adjust child transform	On/Off	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.

inheritance mode	-	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 10: Properties in group **Transformation**→**Transform connections**.

Regarding the transform connection properties described in Table 10, please also read Section 5.2.1 on page 38 for additional information.

4.4 Statistics

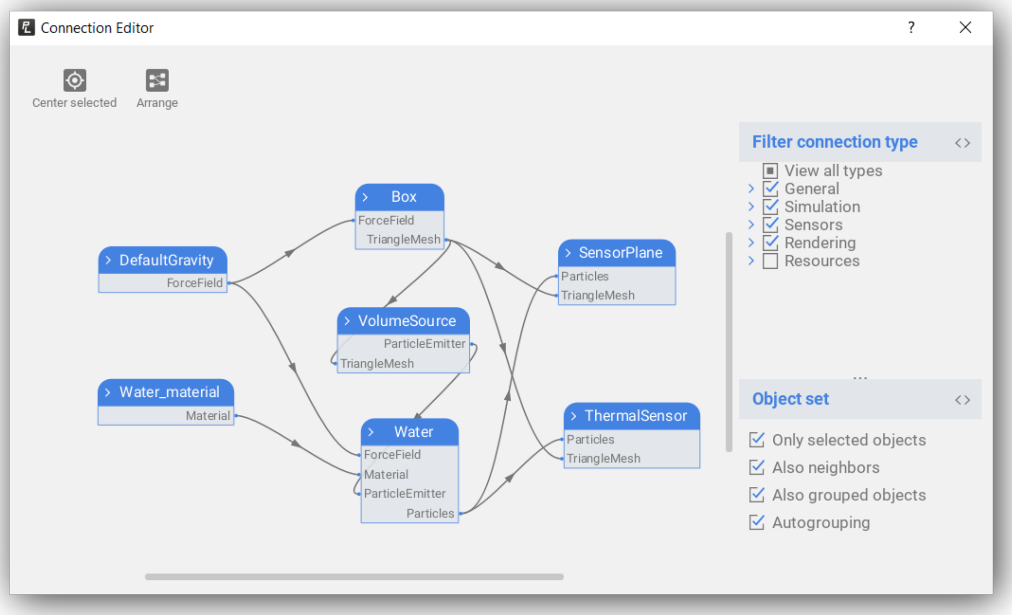
Objects in PreonLab can gather various statistics during simulation or postprocessing. Plots for these statistics can be viewed using the Plot dialog. Additionally, objects may display statistics for the current time in the on-screen-display (OSD). The following properties control whether and how statistics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

Property	Unit/Type	What it does
track statistics	On/Off	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	On/Off	Specifies whether statistics (except for performance timings) are displayed in the OSD.
show timings	On/Off	Specifies whether performance timings are displayed in the OSD.
show osd in video	On/Off	Specifies whether the statistics for this objects are also included in OpenGL frames written to disk.
show color legend	On/Off	Specifies when to display the color legend. By default, the property is set to **Dynamically**, for which the color legend is only shown when data to display is available. Alternatively, **Never** will hide the color legend while **Always** 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	On/Off	Specifies whether statistics are written to disk whenever a new frame is reached during simulation or playback.
track memory	On/Off	If enabled, memory consumption for this object is tracked as a statistic.

store statistics per substep	On/Off	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 **off** by default for solids and sources and **on** for all other objects.
live CSV export	On/Off	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 11: Common properties in group **Statistics** and subgroups **OSD Settings** and **Statistics Recording**.

5 Connections



Filter connection type

- ■ View all types
- ☑ General
- ☑ Simulation
- ☑ Sensors
- ☑ Rendering
- □ Resources

Object set

- ☑ Only selected objects
- ☑ Also neighbors
- ☑ Also grouped objects
- ☑ Autogrouping

Figure 19: Connection editor.

A connection establishes a relation between two objects. Connections are always directed, starting from one object and ending in another object. The boxes symbolizing the scene objects are called object nodes. 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. Similarly, if there is only one solver present, 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 define 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 connections or delete existing ones. Therefore, there are a number of options to simplify the currently displayed graph as described in the following subsections.

5.1.1 Showing object sets

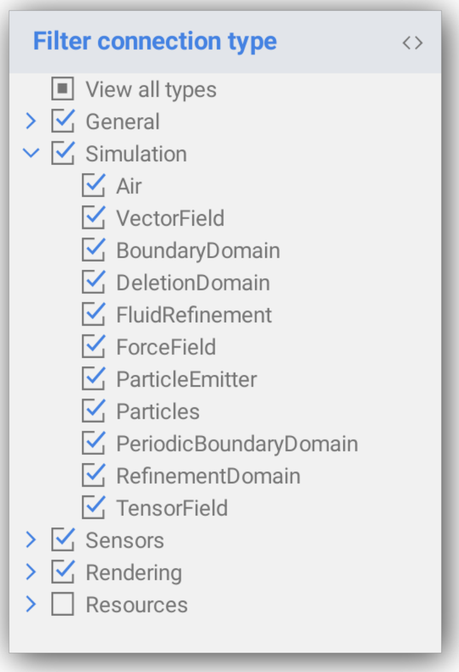
Using the checkbox *Only selected objects* on the right of the connection editor under *Object set*, you can limit the graph to only show objects that you currently have selected in the scene using the Scene Inspector or the graphics window. With the additional checkbox *Also neighbors*, objects that are currently connected to at least one of your selected objects will also be shown. If the *Also grouped objects* option is enabled (which is the default), and a group object is selected in the Scene Inspector, all objects in the group are shown regardless of their individual selection. Disabling this option will hide objects in the group which are not individually selected. Moreover, if *Autogrouping* is enabled (as it is by default), when multiple objects in the same group are selected, they will be shown in a node together. (This only takes effect when you change your selection.)

5.1.2 Arranging objects

To arrange the object nodes in the connection editor, they can just be dragged using the mouse. The respective positions of each object node in the connection editor is saved in the scene. In addition to manually arranging the objects in the graph, the connection editor can automatically arrange the currently visible objects by clicking on the *Arrange* button. Note that this only works as long as only a moderate number of objects are visible. You can also center the current view on your selected objects by clicking on the *Center selected* button.

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 right side of the connection editor. The slot types are categorized in five groups as shown in Figure 20.



Filter connection type ‹ ›

- ▣ View all types
- › ☑ General
- ⌄ ☑ Simulation
  - ☑ Air
  - ☑ VectorField
  - ☑ BoundaryDomain
  - ☑ DeletionDomain
  - ☑ FluidRefinement
  - ☑ ForceField
  - ☑ ParticleEmitter
  - ☑ Particles
  - ☑ PeriodicBoundaryDomain
  - ☑ RefinementDomain
  - ☑ TensorField
- › ☑ Sensors
- › ☑ Rendering
- › □ Resources

Figure 20: Filtering connection types in connection editor.

5.1.4 Grouping objects

To gain a better overview and perform multiple connection tasks on a set of objects, in addition to the group objects created in the scene inspector, you can temporarily group object nodes in the connection editor by using its grouping feature. To do so, select your object nodes of interest by left-clicking and drawing a rectangle around them. All contained object nodes will then be merged into a group and a number will appear in the group node indicating the number of grouped objects as shown in Figure 21. Note that this will only affect the object nodes in the connection editor and will not create group objects in the scene inspector. In general, in order to view the grouped objects individually, you can click on the icon on the top right corner of the group node (see Figure 21). This will permanently vanish the group if it was created in the connection editor. For groups created in the scene inspector, however, you can bring the objects back into the group node by clicking on the icon placed on the top right corner of any of the object nodes belonging to the group (see Figure 21). 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. Figure 21 demonstrates how this works through an example where 6 objects having a fixed set of connections are grouped in different ways. Moving a group node translates the position of the contained object nodes in the connection editor, too.

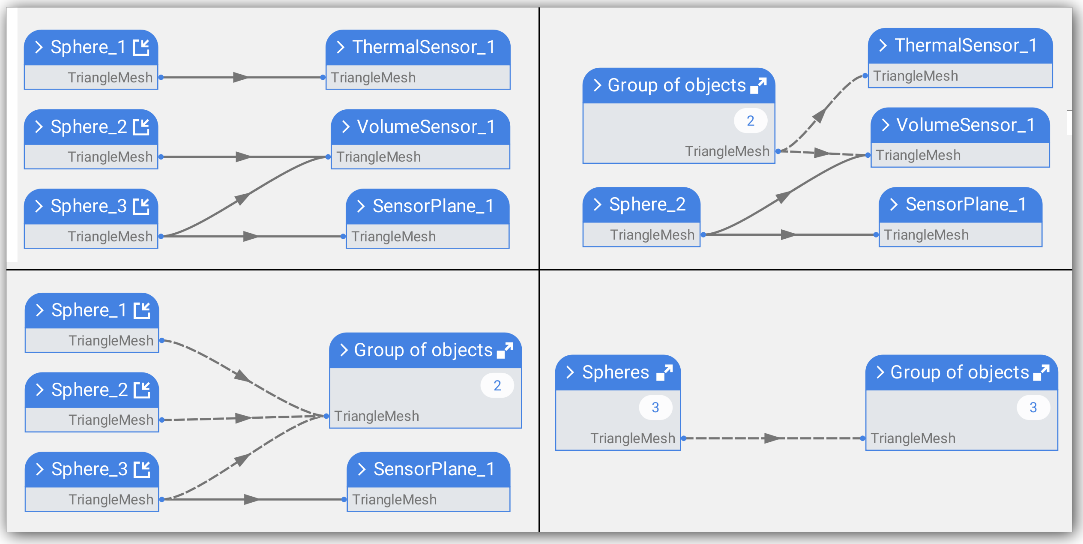


Figure 21: Object grouping in connection editor.

5.2 Transform connections

The *Transform* connection allows you to establish transformation hierarchies, i.e., to combine the spatial transformation of one object with that 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 a *Transform* connection from object A to object B, we could say that object B derives its transformation from object A and that object A is the transform 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 for that object. 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 object A located at position $(10, 0, 0)$ and 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 10 on page 33 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, where 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.

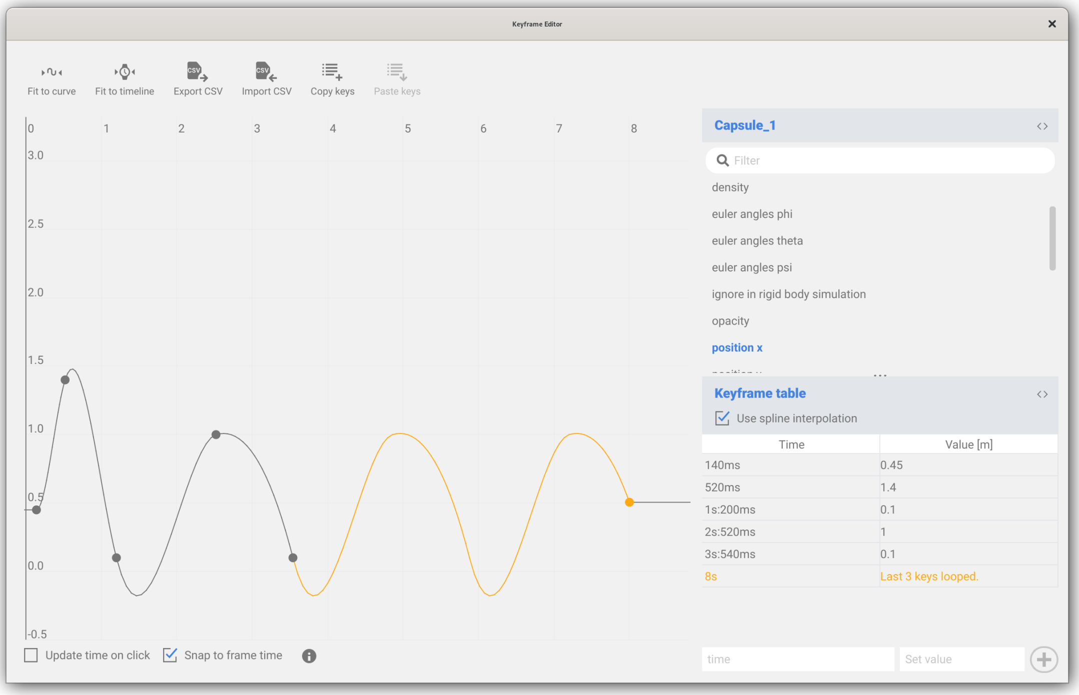
For some fluid properties, keyframes can also be used to map temperature to other properties, see Section 9.1.12 for more details.

6.1 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 an ordered list. Selecting a property shows all keyframes for this property in a table 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. When hovering over the plotting area, a tooltip showing the time and value at the specific position is displayed.

A new key can be created by double clicking in the plotting area. Keys can be moved by pressing and holding the SHIFT key and dragging them using the mouse. Left-clicking a keyframe point selects it, and selected points can be 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 checkbox *Use spline interpolation* which is located between the keyframe table and its caption. After disabling spline interpolation, you can select each key in the keyframe table and modify the interpolation used between this key and the subsequent key, by choosing a different curve type in the third column of the keyframe table. Please note that for properties which can have only a fixed set of possible values, e.g. On and Off, the curve type is not editable. The list of different curve types for each key is available once double clicked in the respective field in the curve column. It is also possible to select multiple cells from the table by



☐ Update time on click ☑ Snap to frame time

☑ Use spline interpolation

Time	Value [m]
140ms	0.45
520ms	1.4
1s:200ms	0.1
2s:520ms	1
3s:540ms	0.1
8s	Last 3 keys looped.

Figure 22: Keyframe editor: The column on the upper right lists all properties that can be keyframed for the selected object. The name of the selected object is shown in the top right corner. In the lower right side the table of keyframes is shown. On the left-hand side the graph determined by the keyframe sequence is visualized. The toolbar at the top contains options for resetting the zoom-level for the curve as well as tools to import/export or copy/-paste keyframes.

pressing and holding the CTRL key while selecting each required cell. The curve type for the multiple selection can be changed by changing the curve type of one of the cell while holding the CTRL key pressed. Similarly the values of a multiple selection can be changed by updating the value of one of the cell and holding the CTRL key pressed while pressing the ENTER button. Please note that multi-editing of values across columns is possible but is rarely sensible.

A new key can be defined by defining time and value in the corresponding fields, in the bottom of the keyframe editor just below the table, and clicking the plus-shaped *Add* button. The newly added key will be now seen in the table and also on the graph.

Below the plotting area, 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. A locked property has its table entries grayed out preventing any editing of the exising keyframe values. A locked property is also marked by an additional comment at the bottom of the table that the sequence is locked. Only keyed properties are lockable. The keyed properties are emphasized with bold letters. By right-clicking on a property, a context menu will give you the following options: a) *Lock selected properties* or *Unlock selected properties*, 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. It is also possible to unlock a property by clicking on the *Unlock* button at the bottom of the table. For a locked property, its keyframes can still be inspected via their curve intervals.

Deleting keyframed points for several properties can be achieved as follows: Select the keyframed properties in the list. Then right-click on one of the selected properties. The last option in the context menu *Delete all keyframes from selected and unlocked properties* deletes all keyframe points from all selected properties which are unlocked.

6.1.1 Keyframe looping

It is possible to repeat a set of keyframed points until any required time point. For this, select a set of consecutive keyframe points from the table and right-click. From the drop-down menu select *Add loop to repeat selected keyframes*. In the appearing dialog, enter the time until the keyframe sequence loop shall be repeated. The keyframes defining the loop are marked yellow and the part of the curve depicting the loop is plotted in yellow, too. Confirm Figure 22. You can also change the end time of the loop keyframe by dragging it in the plot area. While doing so, the number of keyframes defining the loop remains the same but its depiction (i.e., the yellow part of the curve) is updated dynamically based on the change in the end time. The looped keyframe is marked invalid and is displayed in red, if the loop could not be created because the set number of keyframes are not available. When the original keyframe points which determine the looped part are changed, the looped part changes accordingly.

6.1.2 Move keyframes

An existing set of keyframes can be moved to any time point by defining the required time offset. For this, select those required keyframes and right-click and select the *Move selected keyframes* option. Enter the time offset, by which the selected keyframes shall be moved, in the *Time offset* field. By selecting the option *Duplicate and move the selected keyframes* after right-clicking on the selected keyframes, a copy of the old keyframes at their respective initial time points can be preserved while moving the keyframes to the required time offset.

6.1.3 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.

6.1.4 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
```

When importing or exporting keyframes as CSV files, the values are interpreted as having the units currently selected for the quantity in the Scene object. By default, the spline interpolation is turned off for imported keyframes. 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.

6.1.5 Units

The keyframe editor uses the units from the **Scene** object and these can only be changed via the **Scene** object itself. For more information, please see section Section 8.1. Note that you need to take the units into account when importing keyframes.

The values you would like to import from a CSV file need to match the unit you have specified.

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, do not 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. However, loop keyframes and the corresponding loop sequence are not exported. 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

By 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 is capable of plotting statistics over time.

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 statistics 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. In the plot dialog, you will then see all of the statistics grouped by object per default. Alternatively, by choosing the option *Group by statistic instead of object*, statistics are first grouped by name and afterwards by object.

If you hover over a plot displayed in the plot dialog using the mouse, a tool tip will pop up showing the time and value for the hovered position of the plot. Additionally, the average of the plotted data for the current time range will be plotted in another tool tip at the bottom-right corner of the plot.

Activating the option *Update time on click* enables you to jump to the corresponding frame when clicking on the plot.

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

Table 12: Controls for navigating in the plot dialog.

7.1.1 Select statistics

In the plot dialog you can plot a statistic by left-clicking it from the list on the right side. To select multiple statistics, you have the options described in the table Table 13. The variants can be combined. To deselect properties, simply reverse your actions. For instance, if you press CTRL and click on a selected statistic, it will be deselected.

Control	What it does
left mouse button	Clears the previous selection + Selects the new statistic
left mouse button + Drag from A to B	Clears the previous selection + Selects the statistics from A to B
CTRL + left mouse button	Keeps previously selected statistics + adds the clicked statistic.
CTRL + left mouse button + Drag from A to B	Keeps previously selected statistics + adds all statistics from A to B.
SHIFT + left mouse button	Selects all statistics between the current (last selected) statistic and the clicked one.

Table 13: Controls for selecting the statistics in the plot dialog.

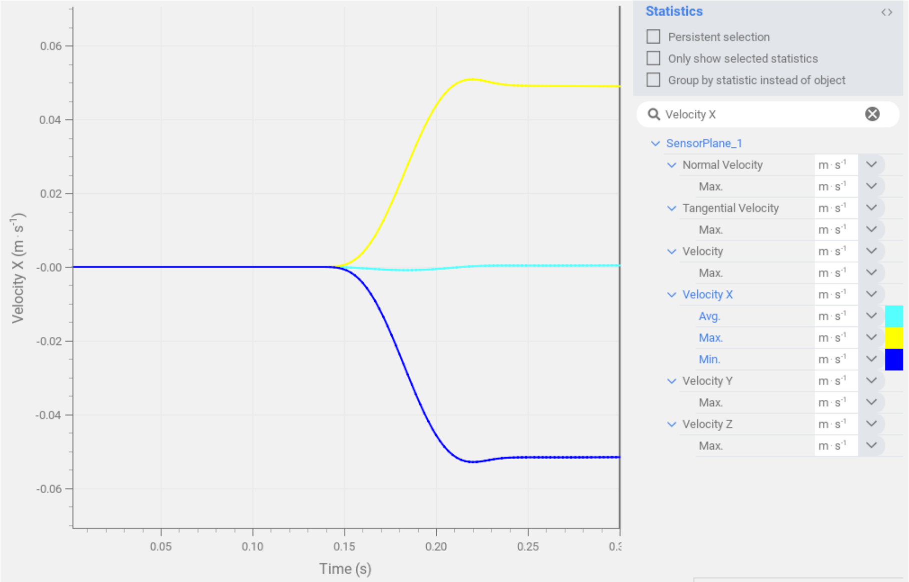
By default, if you select a statistic with the **left mouse button**, then the previous selection is cleared and the new item is selected. To avoid this, you can toggle the option *Persistent selection*. When you activate this option, the previously selected statistics will only be deselected when you click on them a second time. If you toggle *Only show selected statistics*, only the selected statistics will be shown in the list. This option provides the color of each plotted statistic and can be used as a legend.

7.1.2 Filter

On top of the statistics list, you can find a filter text box for the plot dialog. When typing a string there, only the statistics containing that string will be displayed. Moreover, notice that a space between two strings will be interpreted as a logical AND, as demonstrated in Figure 23. Filtering does not have an effect on the selected statistics. Moreover, filtering the statistics list also considers names of the ancestor nodes in the tree. You can combine the filter with the *Only show selected statistics* option to only filter your selected statistics. The filter is not case-sensitive.

7.1.3 Units

Besides the names of some physical quantities, you will find a drop-down list to specify the units of the recorded statistics. The default units when opening the plot dialog



Statistics

□ Persistent selection

□ Only show selected statistics

□ Group by statistic instead of object

Velocity X

- SensorPlane_1
  - Normal Velocity m·s^{-1}
    - Max. m·s^{-1}
  - Tangential Velocity m·s^{-1}
    - Max. m·s^{-1}
  - Velocity m·s^{-1}
    - Max. m·s^{-1}
  - Velocity X m·s^{-1}
    - Avg. m·s^{-1}
    - Max. m·s^{-1}
    - Min. m·s^{-1}
  - Velocity Y m·s^{-1}
    - Max. m·s^{-1}
  - Velocity Z m·s^{-1}
    - Max. m·s^{-1}

Figure 23: Applied filter in the plot dialog.

for the first time are inherited from the **Scene** object. How to change them is explained in more detail in section Section 8.1.

When selecting one of the units, the corresponding curve in the plot dialog will be updated accordingly. Note that when you change the unit for a specific quantity in the plot dialog, this will also adjust the unit in the OSD statistic. Changing the units of statistics is persistent between openings of the same scene. When the selected unit for a statistic diverges from the scene's default unit for the respective type of quantity, the unit is serialized to the scene file. An overview of all available units can be found in section Section 3.12

7.1.4 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 statistics list. This will open a color dialog which lets you choose a color.

7.1.5 Plot settings

The format, the range and the sampling parameters of the plot can be adjusted in the *Plot* tab that you can find on the bottom-right of the plot dialog. When you modify a setting you can reset it to its default value by the respective right-click action, just like for the property editor.

Format

In the **Format** section, you can control whether or not **axis labels** are shown. For the y-axis (value) the label will be automatically displayed as long as the selected units represent the same type of value. With **tick font size** you control the font size of the numbers displayed per axis while with **label font size** you can adjust the font size of the label. These settings are also applied on exported plot images per *Copy image* or *Save image*. Note that you can change the default sizes in the section **Plot** of the *User Preferences*.

Range

The range of the x-axis (time) and y-axis (value) can be adjusted in the respective group of the plot settings tab. The *auto update* toggle switches automatic updating of plot ranges on and off during the simulation/playback.

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 results in noisy plots oversampled in certain regions, which can cause performance problems when processing the data.

In the *Sampling* group, you can find settings for filtering the raw simulation data to produce more meaningful plots. The toggle *fixed step* lets you use a fixed sampling rate for plot sample points which might improve the responsiveness of the dialog when showing statistics holding many sample points. The toggle *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.

7.1.6 Import/Export

You can export statistics either as a *CSV* file or as a native binary *stats* file using the *Export* button found in the toolbar. Note that only the selected statistics are exported. In order to export all statistics regardless of whether they have been selected or not, use the *Export all* button. Exported statistics in *CSV* or *stats* format can be imported via drag'n'drop into the PreonLab GUI. If you have specified additional post-processing steps in the *Sampling* settings, they will be applied on the exported data, as well. The exported data will have the units you already selected in the plot dialog. For instance, if you plot two velocities with the first one in m/s and the second one in km/h, the corresponding exported values would be in m/s for the first statistic and km/h for the second one.

The *Save image* button saves the displayed plots with title and legend as a *PNG* image to a file, the *Copy image* button (shortcut **CTRL + c**) copies that *PNG* image to the clipboard such that it can easily be pasted to a presentation software like PowerPoint. Note that you can change the background and text colors for the image export by changing the corresponding settings in the *User Preferences* as listed in Table 14 (see Section 3.13 for more details about the *User Preferences*).

Preference	What it does
tick font size	Specifies the font size of the numbers on the x-axis and y-axis.
label font size	Specifies the font size of the axes labels.
use theme colors	If enabled, the exported/copied image will have the same background and text color as the current PreonLab theme.
background color	Defines the background color of the exported image. Note that by setting the alpha channel to 0, you can get a transparent background[1]. Prerequisite: **use theme colors** has to be disabled.
text color	Defines the text color of the exported image. Prerequisite: **use theme colors** has to be disabled.

Table 14: The configurable settings in *User Preferences* for the plot dialog export.

The *Reload from disk* button allows you to import data from the hard drive for the current scene. This can be useful if you are running the same scene with another instance, on another computer or even on a cluster. Consequently, you can update your statistics without having to reload your scene. Note that during a simulation PreonLab writes data to the disk only every 100 seconds and only when it reaches a full frame. Therefore, be aware that the plot will not be updated, if new data has not yet been written on the disk.

7.1.7 Specific changes of statistic names with PreonLab 5.0

For PreonLab 5.0, we have introduced a new naming convention for statistics. For example minimum, maximum, average and summed values are from version PreonLab 5.0 onwards abbreviated as **Min.**, **Max.**, **Avg.** and **Tot.**. Values that are accumulated over time are abbreviated as **Cum.**. As many statistic names have changed, we have added a convenience function that automatically maps the names of statistics written by PreonLab versions 4.x to the new names. This step is silently performed everytime you open the corresponding scene with PreonLab 5.0 or PreonNode 5.0. This mapping is also supported by PreonPy. For imported statistics via the *Import→ Import Statistics* menu, no mapping is applied.

Please note that in general, if you load a scene of an older PreonLab version and start a new simulation sequence or post-processing, the existing statistic files will be overwritten. This updates the version of the statistic file. Thus, the newly generated statistics can no longer be interpreted by a former version of PreonLab.

[1] On windows to get a transparent background, *Save image* should be used, not *Copy 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.
Caching→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.
General→individual #threads	If enabled, this allows you to specify an individual number of threads to be used for the related scene in *#threads for this scene*.
General→#threads for this scene	The number of CPU threads used by PreonLab if *individual #threads* is enabled. Note that any number greater than the *maximum number of threads* specified in *User Preferences* will not affect the actual number of threads used by PreonLab.
General→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.
General→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 the negative y-direction. Other changes are not required.

General→simulation frame rate	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.
General→view frame rate	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.
General→simulation start	Denotes the start time of the simulation, specified in the time format $<\#months> mo :<\#days> d :<\#hours> h :<\#minutes> m :<\#seconds> s :<\#milliseconds> ms$. For example, write 1m:3s:500ms 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.
General→simulation end	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.
General→ playback/postprocess start	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.
General→ playback/postprocess end	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.
General→update sensors at substeps	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 (see Chapter 20).
Units	When you create a new scene, the units you have chosen in the *User Preferences* are taken as default. If you want to have different units for a specific quantity, you can easily adjust this by selecting the desired unit for the quantity in this property section of the **Scene** object. Note that in every object that contains the quantity the chosen unit will be changed accordingly. The conversion will be done automatically. The selected units in this section will be also used in the keyframe editor, the OSD statistics and as default in the plot dialog. Furthermore, the unit will be saved to the scene file and will be taken into account when the scene is reopened. An overview of all available units can be found in section Section 3.12.

Table 15: Properties for scene object.

The **Scene** object allows defining the default units to be used for the statistics and physical quantities of all objects in the scene. Moreover, the default units for new scenes can be set in the *User Preferences*. Furthermore, the **Scene** object 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
show granular timings	If enabled, granular timings are printed in OSD. See Table 17 for more details.
show solid particles	If enabled, the number of total and active solid particles is printed. See Chapter 13 for more details on solid particles.

Table 16: Additional statistic properties for **Scene** in **Statistics**→**OSD Settings**.

Timing	What it measures
Cum. Update	The 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.
Cum. Update Simulation	The 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 the neighbor search (collider).
Cum. Update Physics	The 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 the neighbor search (collider).
Collider: Update	The computation time to update neighbor lists for the current simulation step.
Solvers: Update	The computation time to update the physical quantities of all particles like forces, position and velocity for the current simulation step.
GUI: Update	Time required for updating the GUI in each simulation step or frame.
Load	Time required to load the last set of simulation data from disk.
Postprocess	Time required for updating sources and sensors, as well as saving statistics and updating objects with respect to keyframes.
Save	Time required to save the last set of simulation data to disk.

Table 17: Explanation of timings tracked and recorded by the scene.

8.2 Scene UI Settings

Property	What it does
Appearance→show axes	Draws the orientation axes as an overlay in the graphics window.
Appearance→show grid	Renders a grid at the origin of the scene. Each square has a side length of 1 m.
Appearance→show osd	If switched on, statistics and timings are printed as overlay text in the graphics window. For recording you might want to turn this off.
General→gl auto sleep	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.
Recording→save frames	For recording animations you can save the content of the graphics window (including all overlay text). The frames are recorded to the subfolder *Visualization/OpenGL/[CameraName]* of your scene data.

Table 18: 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 ingoing *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 9.

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 Section 10.2). Also, a point can be useful together with the placement tool and a distance sensor to measure distances in the scene (see Section 16.3).

9 Solvers

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
continuous particle size	m	If enabled, the particles for this solver will be in the range defined by **min. particles size** and **max. particles size**. Detailed information on this feature is given in Section 9.1.10.
particle size	m	The particle size controls the resolution of the fluid. The volume of a fluid particle is (particle size)^3.

dimension	-	The default is **ThreeDimensional** which means that PreonLab simulates in 3D. If this property is set to **TwoDimensional** or **OneDimensional**, particle movements are restricted to two dimensions or one dimension. This property is synchronized for all solvers in a scene.
individual frame rate	On/Off	If enabled, a simulation **frame rate** can be set which is different to the **simulation frame rate** provided in the **Scene** object. Consequently, particle data of this solver are written to disk less or more often than those of other solvers.
frame rate	-	The individual simulation frame rate of this object. Only visible, if **individual frame rate** is enabled.
rest density	kg/m^3	The density of the fluid.

Table 19: 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
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. *This property is experimental and only visible if experimental mode is activated.*
solver type	-	*Please note that this feature is still highly work-in-progress and not intended for production use.*Allows to select the pressure solver formulation. **Density invariant** is the default solver. **DI/PS** 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 **DI/PS**, the solver properties of the position shift solver can be set separately as shown in Table 22. *This property is experimental and only visible if experimental mode is activated.*

implicit equation of state	-	If enabled, you can specify a bulk modulus which adjusts the target density according to the pressure, establishing a linearized equation of state between pressure and density. This helps to stabilize the phase interface for multiphase simulations with high density ratios.
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.

Table 20: 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. *This property is experimental and only visible if experimental mode is activated.*
sampling correction method	-	Determines how the solver deals with artificial density errors, usually introduced by refinement steps or initialization. **Continuous** will correct these errors over multiple time steps and is always used when using **continuous particle size**. **Instant** will correct the error immediately and is only suitable for uniform simulations.
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 **DensityErrorAvg**, the pressure solver does iteratively solve for the pressure field until the average density error is below the user-defined value (see **density error**). If set to **DensityErrorCombined**, the maximum density error of single particles is also taken into account with respect to the properties **high density threshold** and **max high density particles**.
density error	%	The tolerated volume compression given in percentage.
max high density particles	%	If the stopping criterion is set to **DensityErrorCombined** 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 **density error**.

high density threshold	%	Particles with a higher density deviation than the given threshold are classified as high density particles.
adaptive RJ omega	On/Off	If enabled, the relaxed Jacobi omega parameter will be adjusted dynamically to improve convergence. *This property is experimental and only visible if experimental mode is activated.*
improved RJ	On/Off	If enabled, an improved variant of relaxed Jacobi method is used to solve the equations for pressure. In general, this improved relaxed Jacobi method converges with less iterations which saves computation time. If you expect few iterations each time step, disabling this property might improve computation time.
prioritize fine particles	On/Off	If enabled, the stopping criterion will give more weight to finer particles when computing the average density error. This property is only relevant when using **continuous particle size**.

Table 21: Solver properties in **Physics**→**Pressure Solver**→**Density Invariant Solver**.

DI iterations and PS iterations for solver type **Density invariant**

When simulating using a Preon solver with **solver type** set to **Density invariant**, the OSD shows *DI iterations*. *DI iterations* refers to the number of iterations that are needed to solve the system of equations that determines the pressure.

Even the usual Preon solver algorithm with **solver type** set to **Density invariant** can perform *PS iterations*. *PS iterations* are performed to improve particle positions resulting from initialization or refinement. Thus the number *PS iterations* is not related to the number of iterations of the pressure solver. Depending on whether the property **sampling correction method** is set to **Continuous** or **Instant**, PS iterations are performed at different time steps.

- When using Continuous correction, PS iterations are executed *in every time step*.
- When using Instant correction, PS iterations are always executed *when a volume source with fill type ”quality” emits particles*. This time-point of emission typically is the start of the simulation.

When **Continuous** correction is used, the statistic PS iterations shows how many position correction steps were done in a last *substep* of the iterations of that time-step. However, for **Continuous** correction there typically is only one such substep such that the statistic PS iterations gives a good impression of the needed computation time needed for position correction.

For **Instant** correction, the PS iterations statistic is potentially misleading, since there are more often several substeps, and the number displayed for PS iterations is only affected by the last substep. However, it is also less relevant for overall performance because PS iterations are only executed after volume source emissions.

Further Preon solver settings

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 **minimum pressure**. *This property is experimental and only visible if experimental mode is activated.*
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. *This property is experimental and only visible if experimental mode is activated.*

Table 22: 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
bulk modulus	Pa	Defines the bulk modulus of the fluid. This allows to relate the tolerated local compressibility with pressure. Lower bulk modulus result in higher compressibility. The physical value for air is in the range of 100 to 150kPa.
adjust interface bulk modulus	On/Off	If turned on, smoothes the bulk modulus and prevents stuck deletion.

Table 23: Additional solver properties in **Physics**→**Pressure Solver**→**Implicit equation of state**. Properties that are only visible if **implicit equation of state** is enabled.

9.1.3 Surface tension

Property	Unit/Type	What it does
cohesion model	-	The following models to compute cohesive forces are implemented : (i) **PotentialForce**, (ii) the experimental **CSS** (continuous surface stress) which can be used for exactly two phases and the two legacy models (iii) **PairwiseForce** and **PreonCohesion**. The models are explained in more detail below.
performance mode	-	Only available for the model **PotentialForce**. It allows to choose between **Speed** and **Quality** 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 **adhesion** value are used to compute the interaction force. This leads to symmetric forces as long as both fluids use the same **cohesion model**.

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

Computational background of cohesion and surface tension

Cohesion forces and surface tension are two descriptions of the same macroscopic phenomena. However, in modeling these forces PreonLab takes two separate approaches:

- **PotentialForce** (as well as the legacy models **PairwiseForce** and **PREON® cohesion**) model the cohesion forces between particles and can be used for free surface flows and flows in closed domains. If several Preon solvers occur, values are set for each solver separately

- **CSS**, which can be used only for exactly two fluids and which instead controls the resulting surface tension directly.

For **PotentialForce** and its legacy variants, cohesive forces are computed as interparticle 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.[1] The model computes the phenomenon of surface tension by computing pairwise attraction forces between particles based on their distance. Thereby, the

[1] 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.

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 nonzero only near fluid surfaces. The advantage of the pairwise-force model over the continuous surface stress 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).

For **CSS**, a discretization of the curvature around the interface between different fluids, i.e., particles from different Preon solvers is used to compute the normal stress at the interface and accelerations resulting from it. The reason it is not recommended or practical for single phase flows is that it requires particles at both sides of the interface to work correctly. The implementation of **CSS** in PreonLab is inspired by the computational solution by Krimi, Rezoug, Khelladi, *et al.*[2], but differs in some details from the method proposed there. Additionally, you can directly set **contact angles** on the triple line between the two fluids and a solid.

Practical recommendations for cohesion and surface tension

PotentialForce: The potential force model is an advanced version of the pairwise-force model which can reproduce microfluidic behavior of certain types 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.

CSS: This is a surface tension model that excels at closed domain multiphase scenarios. The CSS cohesion model is built on the expectation that in the scene each fluid particle has a neighborhood that is filled either with fluid or boundary particles. An example for which CSS is suitable is to simulate water and air in a closed bottle using two Preon solvers – one for water, one for air. The simulation would not work anymore when using CSS after removing the solver for air. More generally, CSS does not work for single-phase simulations.

PairwiseForce: The pairwise-force model implemented in PreonLab has been calibrated and validated for water by the TU Dresden.[3] 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.

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

[2] A. Krimi, M. Rezoug, S. Khelladi, *et al.*, "Smoothed particle hydrodynamics: A consistent model for interfacial multiphase fluid flow simulations," *Journal of Computational Physics*, vol. 358, pp. 53–87, 2018. DOI: `https://doi.org/10.1016/j.jcp.2017.12.006`.

[3] Sprinkling simulation with PreonLab by TU Dresden.

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
n-Butyl alcohol	1.81	Ethylene glycol	4.88
n-Amyl alcohol	1.5	Propylene glycol	1.56
m-Cresol	1.27	Diethylene glycol	2.23
Cyclohexanol	1.15	Triethylene glycol	1.1
Caster oil	1.3	Glycerol	2.07

Table 25: Ratio of bulk viscosity to shear viscosity for selected fluids, as given by Hirai and Eyring.[7]

9.1.4 Viscosity

PreonLab 3.2 included two different models for viscosity handling. The **Morris** viscosity model was mainly a shear viscosity model,[4] whereas the **Monaghan** viscosity model resembled a bulk viscosity model.[5] 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.[6] Table 25 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.

[4] 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.

[5] J. J. Monaghan, "Smoothed particle hydrodynamics," *Annual review of astronomy and astrophysics*, vol. 30, no. 1, pp. 543–574, 1992.

[6] 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.

[7] 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 **Newtonian**, **Herschel-Bulkley** or **None**. 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	-	Defines the flow behavior index of the fluid. For values < 1, the fluid is shear thinning. For values > 1, it is shear thickening. It is only available when the **Herschel-Bulkley** model is selected.
yield stress	Pa	Defines the yield stress of the fluid. This is the offset in the stress-strain relation.
stress growth exponent	s	Defines the stress growth exponent, which acts as a regularization parameter. It has been proposed by Papanastasious for Bingham-type fluids. It is only available when the **Herschel-Bulkley** model is selected.
artificial viscosity	-	If enabled, an artificial viscosity term as discussed in the work of Monaghan[8] is added to yield an additional stabilization when modeling higher Reynolds numbers. This property is only visible when **legacy viscosity values** 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 26: Properties in **Physics**→**Viscosity**.

Models

Newtonian

The default viscosity model which allows 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

[8] J. J. Monaghan, "Smoothed particle hydrodynamics," *Reports on progress in physics*, vol. 68, no. 8, p. 1703, 2005.

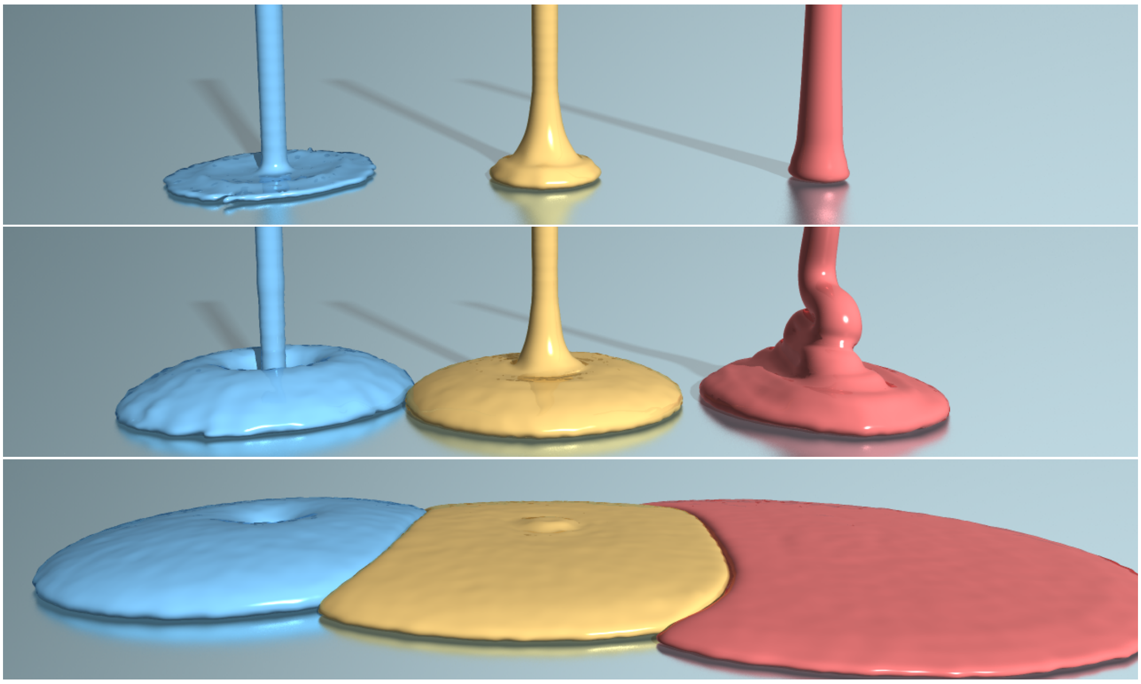


Figure 24: Demonstration of the power-law viscosity model. The fluid in the middle has a flow behavior index of one and corresponds to a Newtonian fluid. The fluids on the left and on the right have flow behavior indices smaller than and greater than one, respectively. The yield stress is kept zero in all the cases.

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

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

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

where K is the **flow behavior index** that corresponds to the **shear viscosity** and **bulk viscosity** properties of the Newtonian model, ε is the strain rate of the flow, n is the **flow behavior index**, τ_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[9], 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

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

[9] 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.

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 τ_y greater than zero, the behavior of viscoplastic or Bingham plastics can be simulated. Those are the fluids that behave like a plastic until a minimum shear stress τ_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 μ 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

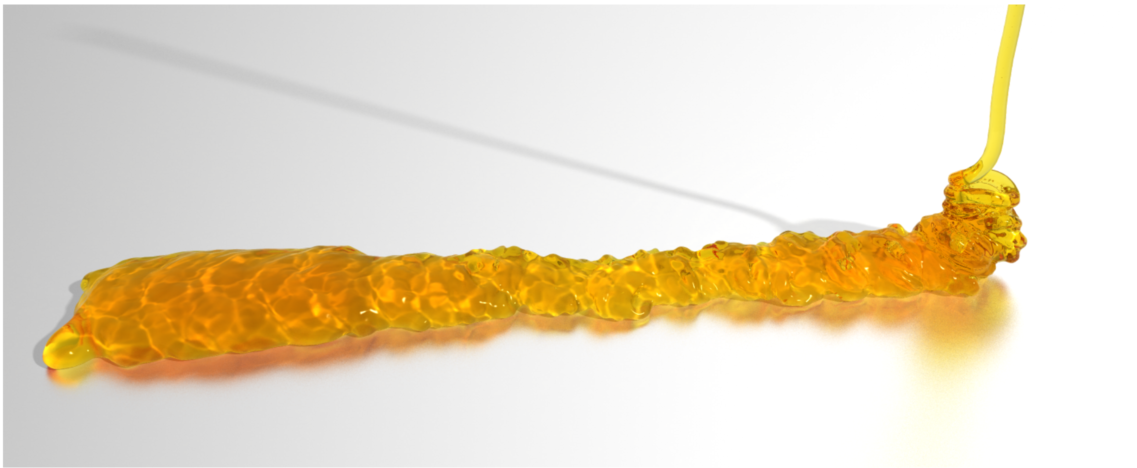


Figure 25: Highly viscous fluids can be simulated efficiently with the implicit viscosity solver.

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
implicit	-	If enabled, the implicit formulation is used instead of the explicit one.
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 viscosity solver. If set to **AvgResidual**, the solver does iteratively solve for the velocity field until the average residual of all particles is below the user-defined value (see **tolerance**). If set to **MaxResidual**, the maximum residual of single particles is taken into account, i.e., it is ensured that the residual of each particle is below **tolerance**.
tolerance	-	The tolerated residual. A higher tolerance reduces the number of iterations and shortens the computation time.

Table 27: 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.

9.1.5 Fluid Presets

Commonly used fluids are provided as presets for convenience of the user. These set the thermophysical quantites Table 28 provides the list of fluid presets which are currently available. These can be accessed via a right-click action on the Preon solver and clicking *Set preset*.

Preset Name	Description
Air	Air at 25°C and 1atm
Oil (5W30)	5W30-Oil[10] at 25°C
default_physics_settings	Water at 25°C and 1atm

Table 28: Fluid Presets

[10] Y.-C. Liu, B. Sangeorzan, and A. Alkidas, "Experimental investigations into free-circular upward-impinging oil-jet heat transfer of automotive pistons," *SAE International Journal of Engines*, vol. 10, no. 3, pp. 790–801, 2017.

9.1.6 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 corresponding solid velocity at that position, thereby realizing a no-slip boundary condition at the solid interface as proposed by, e.g., Hu and Adams.[11] 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
no gap	On/Off	If switched off, fluid particles are kept in a distance to solid walls (geometries) of the length of particle size. This results in a visible gap in the size of half the particle size and a thickening of geometries reducing the inner void volume. By switching, the **no gap** 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 **no gap** option is on which might increase the computation time and memory consumption.
Adhesion→use cohesion as adhesion	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.
Adhesion→solid adhesion	-	Adhesion value used in contact with solids. This property is only shown if **Adhesion→use cohesion as adhesion** is enabled.
Friction→distance correction	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 **roughness** set to 1.
Wall Function→von Kármán constant	-	Parameter κ used in the law of the wall. *The default value is 0.41.*
Wall Function→turbulent Prandtl number	-	Turbulent Prandtl number Pr_t of the fluid, used in the computation for **Wall Function→thermal** . *The default value is 0.86.*

Table 29: Solver properties in group **Physics→Boundary handling** and further subgroups.

[11] 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.

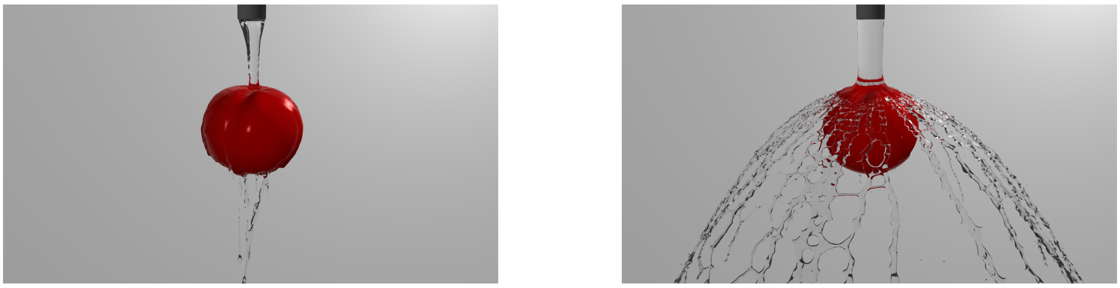


Figure 26: Solid-fluid interaction. Left: A tomato is washed under a water faucet. In the beginning the outflow is low. Cohesion effects are clearly visible at the concave water jet and the droplets beneath the tomato. Due to the established adhesion model (see Section 9.1.3 and Chapter 13), the water 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.

9.1.7 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.

Property	Unit/Type	What it does
maximal timestep	s	If **adaptive** time stepping is enabled, this value determines the maximal timestep. This property is automatically adjusted to ensure stability based on hydrostatic pressure.
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.
pressure guided timestep	On/Off	If **adaptive** and pressure guided timestep are enabled, the timestep is additionally adjusted for very dynamic scenes not only based on the predicted velocity field but also taking the pressure Poisson equation into account. This can reduce particle deletion.
max. new density error	-	If adaptive and pressure guided timestep are enabled, this property regulates the newly incoming density error allowed before the solve.
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.

adapt max timestep to particle size	On/Off	If **adaptive** and **pressure guided timestep** as well as **continuous particle size** are enabled, setting this property lowers the maximum time step dynamically according to the actual smallest existing particle in the simulation.

Table 30: Properties in group **Time Stepping**.

The time step used in the simulation is logged as a statistic: *Time Step*. The time step determined by the CFL number is logged as a statistic: *CFL Time Step*. The time step determined by the criterion **pressure guided timestep** is logged as a statistic: *Time Step: Dynamic Limit*'. For adaptive, the time step is also restricted by explicitly computed viscosity forces, i.e. **Viscosity solver**→**implicit** is turned off. This is additionally logged as a separate statistic: *Time Step: Explicit Viscosity*.

9.1.8 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. A deleted particles visualizer (see Section 16.16) identifies particle deletion and which of the criteria listed in Table 31 applied to each deleted particle.

Property	Unit/Type	What it does
on CFL violation	-	This property specifies what happens with particles that violate the CFL condition due to numerically challenging settings. **Ignore** does not adjust the particles. **DeleteParticles** deletes all particles violating the CFL condition and **AdaptTimestep** individually adapts the time step for each particle so they meet the CFL condition.
stuck prevention	On/Off	Enables or disables the stuck prevention which may delete particles based on their density.
predefined stuck values	On/Off	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 be represented accurately anymore for numerical reasons. This property specifies whether such a particle should be deleted (**Delete**), its pressure value should be clamped to a predefined maximum value (**Clamp**) or the pressure value is kept without adaption (**None**).

Table 31: Properties in group **Deletion criteria**.

9.1.9 Density Computation → Closed Domain Correction

For closed domain settings, it is important to fill the domain completely with particles to avoid the formation of holes. However even slightly overfilling the domain will result in an unstable simulation. The following properties can help to setup stable simulations in closed domains.

Property	Unit/Type	What it does
optimize for closed domain	On/Off	Enables or disables optimization for closed domain. If enabled, volume source emissions will attempt to achieve a specific fill ratio that ensures a stable simulation.
target fill ratio	-	The target fraction of fluid to void space for volume source emission. Only relevant if **optimize for closed domain** is used. *This property is experimental and only visible if experimental mode is activated.*
closed domain density correction	-	Dynamically adjusts particle densities to fill the domain completely while ensuring a stable simulation. This can fill small holes caused by imperfect fillings. Only relevant if **optimize for closed domain** is used. *This property is experimental and only visible if experimental mode is activated.*
hole detection threshold	-	Sets the density gradient magnitude above which a particle is considered to be near a hole. Only relevant if **closed domain density correction** is enabled. *This property is experimental and only visible if experimental mode is activated.*
density scale max	-	Sets the maximal density scale factor for the correction. Only relevant if **closed domain density correction** is enabled. *This property is experimental and only visible if experimental mode is activated.*

Table 32: Properties in group **Density Computation→Closed Domain Correction**.

When using area sources in a closed domain simulation, we further recommend to use the **correct area discretization** property (see Section 10.1).

9.1.10 Continuous particle size

In many applications, it can be beneficial to represent the fluid with different particle sizes. PreonLab supports non-uniform simulations with the **continuous particle size** option. After enabling it, you can specify the minimum and maximum particle size for this Preon solver. Refinement and coarsening is guided by domains that specify which particle size should be used in different regions of the scene. To set up a refinement domain, insert a **Box Domain**, set its condition type to **refine_continuous** and connect it to the Preon solver using the *FluidRefinement* connection slot. Particles inside the domain will then be refined or coarsened to the **target particle size** of the domain. As a shortcut, a pre-configured refinement domain can be created by right-clicking on the solver and clicking on *Create Refinement Domain*. Particles outside any domain will be coarsened to the maximum particle size.

For any connected source, area source and volume source, you have to specify the **particle size** if **continuous particle size** is enabled. Of course, the particle sizes defined in the domains and the sources need to be within the range specified by **min. particle size** and **max. particle size** of the solver. If you change this range later on, the properties of the connected domains and sources are automatically adjusted. This does not hold for *Undo* operations. So, please check these properties carefully before you start a simulation.

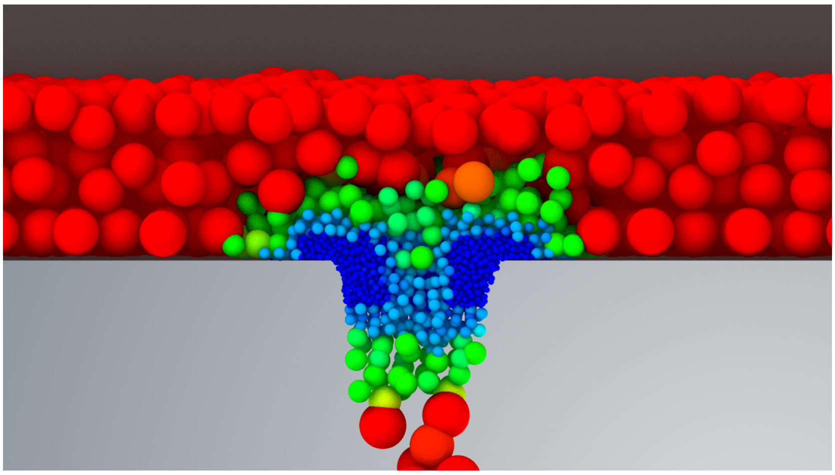


Figure 27: In this Torricelli CFD benchmark, a speedup of 82x can be achieved by using **continuous particle size** with **min. particle size** of 1.25mm(blue) and **max. particle size** of 1cm (red) instead of a uniform **particle size** of 1.25mm.

Coarsening

The following Preon solver parameters influence coarsening behavior (the merging of fine into coarse particles).

Property	What it does
coarsen particles	Enables or disables coarsening.
merge max. vel deviation	Sets the maximum standard deviation for the velocities of a cluster of particles to be eligible for merging during a simplification. The value is given as a multiple of the particle size.
merge max. relative vel deviation	Sets the maximum relative standard deviation for the velocities of a cluster of particles to be eligible for merging during a simplification.

Table 33: Properties of Preon solver in group **Adaptive Resampling**.

The two velocity-based merging conditions are evaluated for each velocity component separately. Only one criterion needs to be fulfilled for each component for a cluster of particles to be eligible for merging.

Upgrading from versions older than 5.2

Before PreonLab 5.2, refinement and coarsening was realized using multiple Preon Solvers that each represented one resolution level. These scenes can still be loaded, post-processed and simulated in PreonLab 5.2. If old simulation data is not essential, we recommend to upgrade to the new approach using continuous particle size. To upgrade, perform the following steps:

1. Delete the refined solvers and just keep the solver with the coarsest particle size.
2. Enable **continuous particle size** in the remaining solver and set **min. particle size** to the particle size of the previously deleted finest solver.
3. Change the condition type of connected refinement domains to **refine_continuous** and set the **target particle size** as required.
4. If volume or area sources previously emitted into one of the deleted refinement solvers, connect them to the remaining solver and set their particle size as required.

Limitations

- The maximum ratio of **max. particle size** to **min. particle size** is restricted to 32. Furthermore, this ratio must be a power of 2. Both is automatically ensured by PreonLab. We recommend to use a maximum ratio of 8 (please note that this can reduce particle counts up to $8^3 = 512$ times compared to a uniform simulation).
- **Preon Mesher** does not support different particle sizes per solver.
- Evaporation and the **Air Object** are not supported. Please note that this does not apply to the **Drag Force**, **Airflow** objects or using another Preon solver to simulate air, which are all supported.
- Area sources with seedpoint area and **static source area** set to off are not supported. Please note that we do not recommend to set this property to off in any case to ensure good performance.

9.1.11 Multiphase

Simulating multiple fluids with different physical characteristics is possible by creating one Preon solver per fluid and adjusting their properties. A first step to create such a scene is to set up the solvers and corresponding sources for the fluids, and adjust density, viscosity etc. according to the physical values of the fluids. See Table 34 and Table 35 for an overview of the settings in multiphase that depend

1. on the number of fluids, i.e., phases that should be simulated, and
2. on whether the domain is closed and fully filled with the fluids or if the flow has free surfaces.

In Table 34 and Table 35, the symbol ⊕ highlights the recommended options.

	2 fluids	
Property	Closed domain	Free boundary
cohesion	**potential force** **CSS** ⊕	**potential force**
fill method	**quality** (possibly with **optimize for cl. domain**) ⊕ **uniform**	**quality** ⊕ **uniform**

Table 34: Possible settings for multiphase with 2 fluids.

	3 or more fluids	
Property	Closed domain	Free boundary
cohesion	**potential force**	**potential force**
fill method	**quality** (possibly with **optimize for cl. domain**) ⊕ **uniform**	**quality** ⊕ **uniform**

Table 35: Possible settings for multiphase with 3 or more fluids.

A classical multiphase scenario would be to simulate water in the liquid phase using one Preon solver and, at the same time, air in the gaseous phase using another Preon solver. Similarly, combinations of motor oil and air can be simulated using Preon solvers.

The case of two fluids air/oil is notable for its high density contrast. The ratio of densities is in the order of 10^3. Another scenario would be a scene in which oil and water occur. Then, the density contrast would be lower. In the following we use the convention to call multiphase settings with fluids having a density ratio between 1 and 10 “low density ratio” and those having a density ratio of 100 or higher “high density ratio”. Oil and air for transmission is the most typical senario of a multiphase simulation with a high density ratio. When simulating multiphase with high density ratios it is recommended to use closed domains.

[12] J. Adelsberger, P. Esser, M. Griebel, *et al.*, “3d incompressible two-phase flow benchmark computations for rising droplets,” in *Proceedings of the 11th World Congress on Computational Mechanics (WCCM XI)*, (Barcelona), E. Oñate, Ed., 2014

[13] H. Liu, T. Jurkschat, T. Lohner, *et al.*, “Detailed investigations on the oil flow in dip-lubricated gearboxes

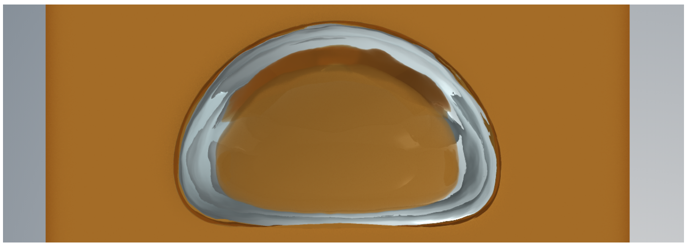


Figure 28: A 3d multiphase benchmark as proposed by Adelsberger, Esser, Griebel, *et al.*, test case 1 therein[12]. The involved fluids have a density ratio of 10 : 1. The figure shows the state at simulated time 1s. Approximately 4M particles were used.

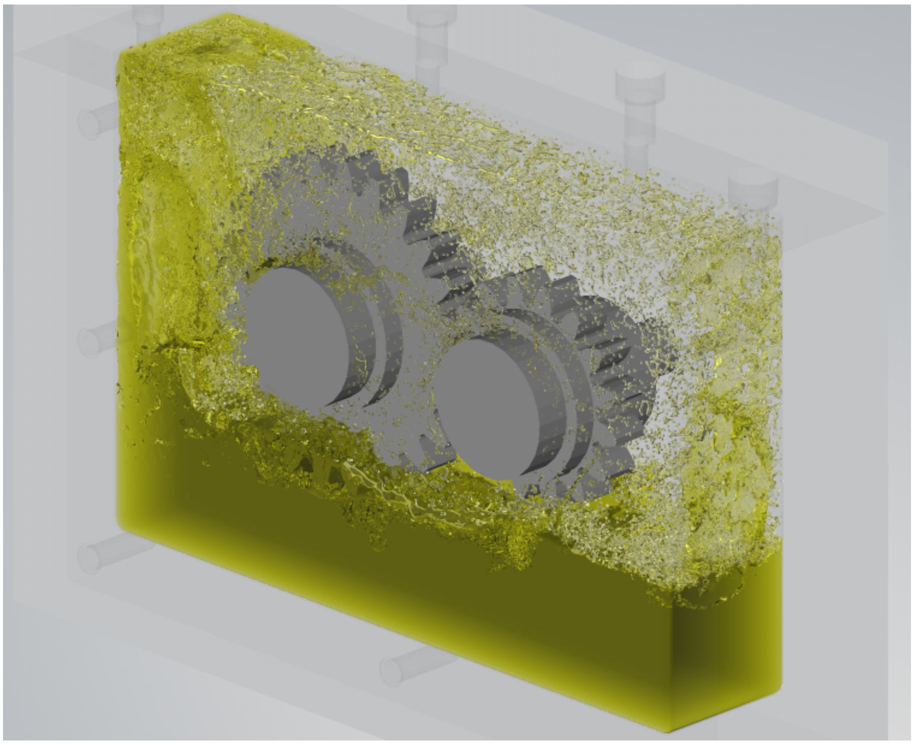


Figure 29: A multiphase simulation of a FZG no-load power loss test rig. The tangential speed of the gear is 10.5 m/s. The used lubricant is FVA3, i.e., has a viscosity grade of ISO VG 100. Experimental and other numerical results can be found in Liu, Jurkschat, Lohner, *et al.*[13].

Known limitations

CSS is presently designed for computing surface tension between exactly two solvers, i.e., two phases.

by the finite volume cfd method," *Lubricants*, vol. 6, no. 2, 2018. DOI: 10.3390/lubricants6020047. [Online]. Available: https://www.mdpi.com/2075-4442/6/2/47

Bulk modulus: Enabling the property **implicit equation of state** changes the pressure solver in the sense that it will not aim for incompressibility of the fluid. Instead, the solver will try to achieve a functional equation between pressure and density. The amount of compressibility is determined by the bulk modulus K. The ratio between present density ρ and rest density is given by ρ_0

$$\rho = \rho_0\left(1 + \frac{p}{K}\right) \tag{1}$$

Equation (1) is a linearized version of the Murnaghan equation of state (also called Tait equation), where r is an experimentally determined parameter and p_0 is a reference pressure,

$$\rho = \rho_0\left(\frac{r}{K}(p - p_0) + 1\right)^{\frac{1}{r}} .$$

for the simplification $p_0 = 0$. For setting the **bulk modulus** K, you should be guided by the physical values for bulk modulus. If you apply a fluid preset (see Section 9.1.5), it will assign a physical bulk modulus value. If you experience stuck deletion, turn **adjust interface bulk modulus** on.

Gradient correction: It is recommended to turn **gradient correction** to on, with **symmetric gc** on as well, for multiphase scenarios. In case the simulation results show too much fragmentation at the interface, turning gradient correction off can improve the simulation results. See Figure 30 for an example of a fragmented interface.

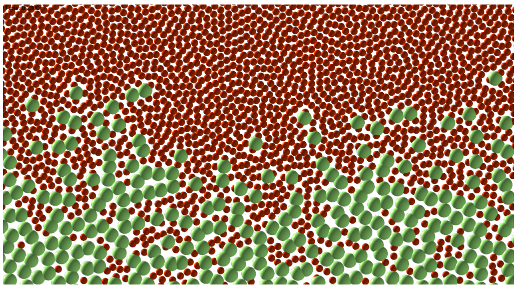


Figure 30: A 2-dimensional multiphase simulation with density ratio 100 : 1 that leads to a fragmented interface when using **gradient correction**.

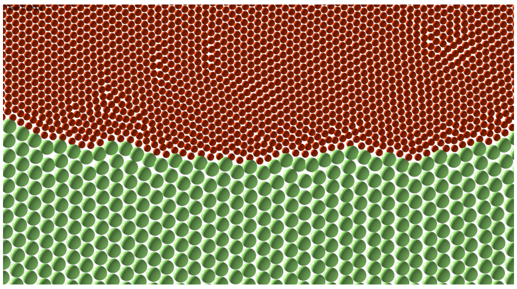


Figure 31: The same scene without gradient correction. In some instances of 2d scenes, setting **gradient correction** to **off** can lead to a smoother interface for high density ratios.

See Section 9.1.2 for more details on gradient correction (gc).

Surface tension: It is recommended to use the cohesion model **CSS** for multiphase scenes in closed domains. This surface tension model helps to create smooth interfaces between the fluids.

[15] *Revised release on surface tension of ordinary water substance*, International Association for the Properties of Water and Steam, Jun. 2014. [Online]. Available: `http://www.iapws.org/relguide/Surf-H2O-2014.pdf` (visited on 10/26/2021)

[15] R. J. William Jr. and L. D. Wedeven. (1971). Surface tension measurements in air of liquid lubricants to 200 c by the differential maximum bubble pressure technique, [Online]. Available: `https://ntrs.nasa.gov/api/citations/19710024039/downloads/19710024039.pdf`

Fluid	surface tension in N/m
water at 273.15 K	0.07564
water at 288.15 K	0.07197
water at 323.15 K	0.06794
Synthetic paraffinic oil at 296.15 K	0.0303
Superrefined paraffinic mineral oil at 296.15 K	0.0298

Table 36: Surface tension between water[14]and air and between oil[15]and air.

Each solid has a contact angle that it will enforce between the two fluids on the boundary. Figure 32 shows a simple example.

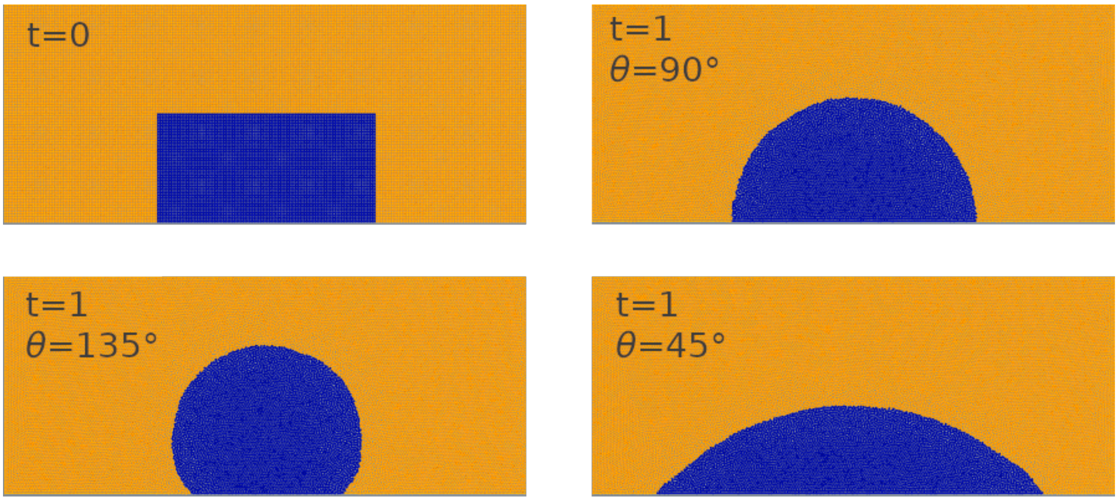


Figure 32: Effect of different contact angles. *Top left:* A higher density fluid is initialized as a block within another fluid with lower density. Simulation results with different contact angles are shown in the other images. *Top Right:* A contact angle of 90° is enforced on the boundary. *Bottom left:* A contact angle of 135° is enforced on the blue fluid with higher density. *Bottom right:* A contact angle of 45° is enforced on the blue fluid with higher density.

Setting up sources for multiphase in closed domains: In the case of initialization using volume sources, proceed as follows. Create one volume source for each fluid type, where each contains the full domain of simulation. Then, create a solid body, which you turn inactive. This will then serve to create the interface at initialization. For example, if you would like to create a spherical air bubble in a cuboid filled with water the inactive solid would be a sphere. Finally, create a seedpoint for each fluid type and connect the seedpoints to the volume source. See Section 10.2.2 to find out more about seedpoints. This procedure can be used with any imported geometry. Figure 33 shows a simple example.

It is recommended to use quality filling for both sources while simulating multiphase scenes. In settings with closed domains, the solver can perform too many iterations until convergence if the average density is too high. In such cases, closed domain

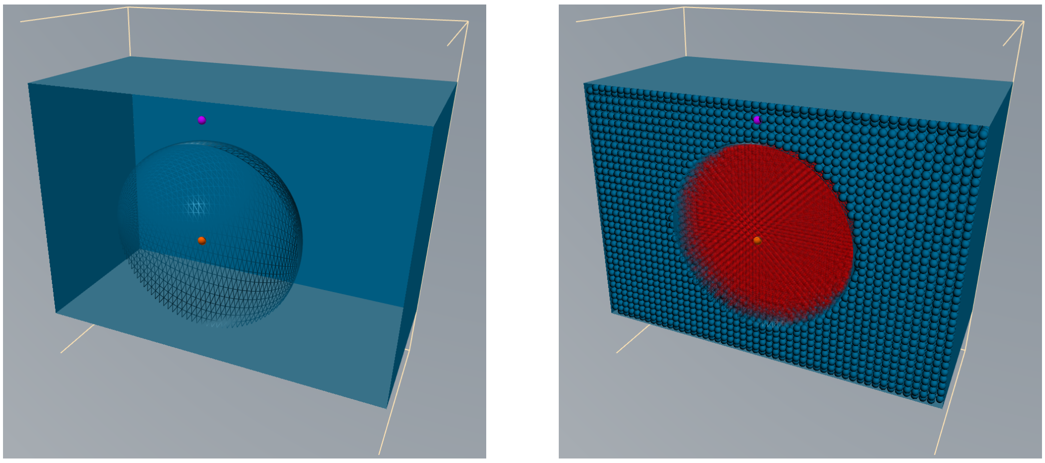


Figure 33: Setting up volume sources for two phases. Left figure: without fluid. Right figure: with fluids, air shown in opaque red. The sphere in the middle is set to inactive and becomes the interface between the two fluids. The orange sphere is the seed point for the volume source for air, the violet one is the seed point for the volume source for water.

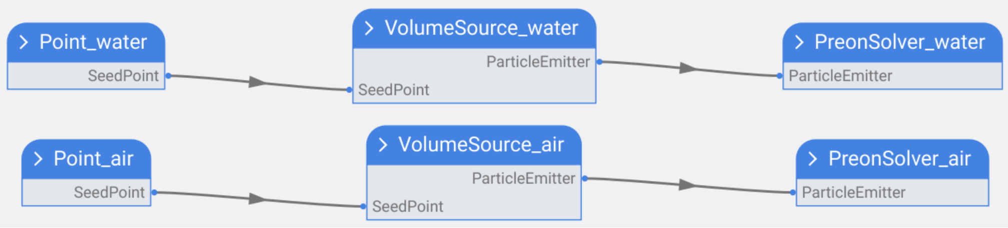


Figure 34: Connections for setting up volume sources for two phases.

correction can improve performance and is explained in Section 9.1.9.

Time stepping for multiphase simulations: Each solver has a separate property for timestep, i.e. maximal timestep. Similarly, each solver has different CFL properties. Since the respective particle velocities can also differ, various combinations can appear. In each timestep, the lowest of the Preon solvers objects is used to determine the actual performed timestep.

For multiphase simulation with a high density contrast, it is recommended to turn on the property **pressure guided timestepping** because these flows tend to be very turbulent.

If particle deletion occurs in a multiphase simulation, the following can be done step by step:

1. Check that **implicit equation of state** is enabled and bulk modulus is set as recommended.

2. Activate the deleted particle visualizer to get a clearer picture where and based on which criterion particles are deleted.

3. In case of significant particle deletion, make sure to activate pressure guided

timestepping. In most cases this should be sufficient and you don't need to lower the CFL number. In other cases, lowering the CFL number for the lighter fluid might improve stability. If unacceptable CFL deletion persists still, lower **max. new density error**.

Non-uniform multiphase simulations

As a rule of thumb, you should pick identical particle sizes for each Preon solver in your scene. An instance where picking different particle sizes makes sense are multiphase simulations. As an example, a scene may feature a Preon solver with particle size 1 cm representing air and another one with particle size 0.5 cm representing water. The particle-size ratio is then 2 : 1. Setting the particle size to different values can help in achieving two separate goals.

1. **Lowering the number of particles**: For a breaking dam for example, the volume in the simulation domain covered by air is typically much greater than the one covered by water. It would possibly cost too much computational resources to resolve both with high resolution. If the phase that is less in the focus of the numerical simulation is also the phase with the lower density, then you can lower its resolution. Figure 35 shows an example for this.

2. **Make mass per interpolation point for both fluids more equal:** This can be used to increase the performance of multiphase simulations, especially those with high density ratio. For a given density ratio a between the phases, the mass per particle for interpolation points from the different fluids has the same ratio a. Increasing the particle size for the fluid with lower density can be used to make the mass per interpolation point more similar in both fluids and help improve the performance of the simulation.

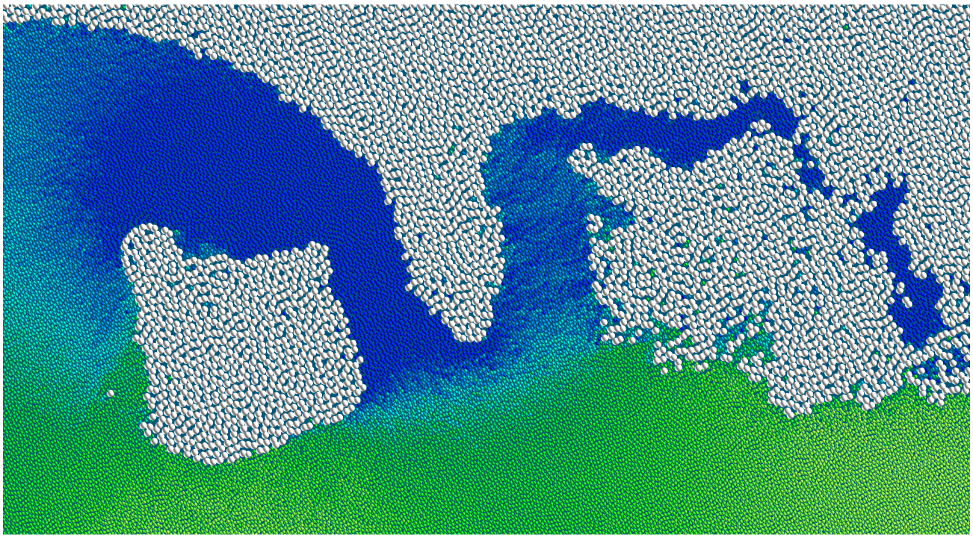


Figure 35: Detail from a 2-dimensional dam break simulation with smaller particle size, 0.5 mm for water (shown in blue and green) and bigger particle size 1 mm for air (white). The preset values for density of water, 998.2071 kg/m^3, and air, 1.22 kg/m^3, were used.

It is recommended to use particle-size ratios of at most 2.5. There are constraints to this tweaking of ratios. Clearly, if geometric details like pinions are around in a simulation, larger particles could get stuck or create other problems if they are too large for the geometric details.

9.1.12 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 **Thermodynamics→ thermodynamics** to on, see Table 37 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
thermodynamics	On/Off	Defines whether the fluid particles thermally interact with each other and with particles from other fluids or solids. If enabled, heat diffusion is allowed among the particles of this fluid and with particles of other fluids or solids, provided the other fluids or solids have thermodynamics enabled. If disabled, they are not considered in the thermodynamic equation at all. The following properties are enabled only if this property is enabled.
heat capacity	J/(K 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 K)	The thermal conductivity is the property of a material to conduct heat. The default is 0.6 (liquid water at 25°C).
implicit	On/Off	If enabled, heat conduction is computed using an implicit formulation which is often computationally 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 39.
convergence criterion	-	If an explicit solver is used, the timestep size needs to be controlled to maintain the stability of the thermal solver. The 2 options, **Diffusivity** and **PairwiseTempDiff**, are explained below in *Convergence criterion*. The default option is **Diffusivity**.

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 *Keyframing temperature-based viscosity* below for more details.
temp. based density	On/Off	This property is enabled, only if the **continuous particle size** property is enabled. If enabled, the rest density of each particle is based on the temperature of the respective particle. *This property is experimental and only visible if experimental mode is activated.*

Table 37: Properties of Preon solver in group **Physics** → **Thermodynamics**.

Property	Unit/Type	What it does
conductivity mean	-	If set to **Arithmetic**, the arithmetic mean from the conductivity values κ of the participating media (solid and fluid) are taken as the effective conductivity at the interface. If set to **Harmonic**, the effective conductivity is the harmonic mean computed as $\left(4 \cdot \kappa_{solid} \cdot \kappa_{fluid}\right) / \left(\kappa_{solid} + \kappa_{fluid}\right)$.
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 38: 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 **AvgResidual**, the solver does iteratively solve for the temperature field until the average residual of all particles is below the user-defined value (see **tolerance**). If set to **MaxResidual**, the maximum residual of single particles is taken into account, i.e., it is ensured that the residual of each particle is below **tolerance**.
tolerance	-	The tolerated residual. A higher tolerance reduces the number of iterations and shortens the computation time.

Table 39: Solver properties in **Physics**→**Thermodynamics**→**Thermo Solver**. These properties are only visible if **Physics**→**Thermodynamics**→**implicit** is enabled.

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

Convergence criterion

A good convergence criterion is essential to maintain the stability of the explicit thermal solver. The output of the convergence criterion calculation is the maximum allowed time step size Δt_{max}. When using the **PairwiseTempDiff** criterion, the following definition is used:

$$\Delta t_{\text{max}} = \min\left(\frac{\Delta T_{\text{max}}}{2\Delta T}\right)$$

For each particle, the maximum temperature difference to its neighbor particles (ΔT_{max}) is divided by the accumulated rate of temperature change for the particle (ΔT). When using the **Diffusivity** criterion, the following definition is used:

$$\Delta t_{\text{max}} = \frac{h^2}{8\alpha}$$

where h, and α are the particle size and the thermal diffusivity, respectively.

Best Practices

Using the implicit thermal solver removes the need for a convergence criterion that restricts the maximum time step size. To decide whether using an implicit thermal solver is beneficial, the following recommendations can be followed:

- If a moving fluid is present in the domain, the timestep size is often limited by the CFL number. In this case, the timestep size for the fluid solver is likely to be below the thermal convergence criterion requirement. Thus, leaving the explicit solver enabled is beneficial because the computations involved are cheaper and enabling the implicit solver would not improve performance.
- If only a solid solver is used, turning on the implicit solver could be beneficial. The thermal convergence criterion can be bypassed because the time discretization is unconditionally stable. Although unconditionally stable, this scheme does not guarantee that the transient behavior of the solver is independent of the timestep size. Please choose the timestep size carefully to have good performance without sacrificing quality.

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*.

Keyframing temperature-based density

A temperature-based density can be keyframed as a function of temperature, similar to how a temperature-based viscosity can be keyframed. This property is enabled only if the property **General→continuous particle size** is enabled. Enabling the temperature-based density allows the solver to define a density for each particle based on its temperature. Since the density varies with temperature, the volume of the particle also varies with temperature to conserve its mass. It is to be noted that the property **continuous particle size**, in this context, allows the particle sizes to be changed based on the particles' densities and is not related to the refinement process. The refinement or coarsening process is currently restricted in cases where temperature-based density is enabled. In case, the scene has the property **temp. based density** enabled and already contains at least one refinement domain, it should be ensured that all refinement domains are disconnected manually from the solvers to prevent any refinement or coarsening from occuring.

9.1.13 Wall Functions

In many applications the steepest gradients of e.g. velocity and temperature can be found at the walls. If a particle size which is too large is chosen, it may result in incorrect prediction of shear stresses and heat fluxes at the wall. A fine particle particle size is required to resolve these gradients, increasing computation time and costs drastically.

To assist with this problem **Wall Functions** can be used, which corrects these gradients at the wall, thus providing better prediction of shear stress and wall heat fluxes. This capability of the Preon solver can be activated by selecting the solid of interest and switching on the modelling under **Physics→Wall Function**.

Momentum Wall Function

This modelling approach is described in Bredberg[16], and is based on the law of the wall, which describes the variation of the dimensionless flow velocity u^+ with dimensionless wall distance y^+. Within the so-called ”log-law region” the relationship between u^+ and y^+ can be described by a log-law function:

$$u^+ = \frac{1}{\kappa} \ln E y^+$$

where κ is the **von Kármán constant** and E is the **roughness factor**[17].

Thermal Wall Function

The approach used for thermal wall functions is analogous to that used in momentum wall functions and is based on Jayatilleke[18]. If y^+ lies in the thermal log-law region of the fluid, the relationship between dimensionless temperature T^+ and dimensionless wall distance y^+ is described by

$$T^+ = \mathrm{Pr}_\mathrm{t}(u^+ + P)$$

where Pr_t is the turbulent Prandtl number that typically takes a value of 0.9, while P is a relation presented in Jayatilleke and is a function of the fluid’s Prandtl and turbulent Prandtl numbers.

Best Practices

The wall function model works best when the y^+ values of the considered regions in the scene lie within the validity range of the log-law, i.e. between $y^+ = 30$ and $y^+ = 300$. The y^+ values at a given solid can be measured by connecting a **Y+ Sensor** to the solid (see Section 16.9). For the roughness parameter E, a value of 9.8 is suggested for smooth walls while the literature suggests a value of 0.41 for the von Kármán constant κ. A higher roughness of the wall corresponds to a lower value for E.

Please note that wall functions are only compatible with the **boundary type→temperature** property of the solid at this time.

9.1.14 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

[16] J. Bredberg, “On the wall boundary condition for turbulence models, chalmers university of technology, department of thermo and fluid dynamics,” Internal Report 00/4, Göteborg, Sweden, Tech. Rep., 2000.

[17] In the literature it is often the case that the law of the wall is represented in the form $u^+ = \frac{1}{\kappa} \ln y^+ + C$. The parameter E is then $E = \exp(\kappa C)$.

[18] C. Jayatilleke, “The influence of prandtl number and surface roughness on the resistance of the laminar sub-layer to momentum and heat transfer,” 1966.

is inactive but can be activated by setting **evaporate** of the solver to **On**, see Table 40 for more details.

Table 40 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:

$$g_h = \theta \cdot A \cdot (\tilde{q}_s - \tilde{q}), \tag{2}$$

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:

$$\theta = \textbf{prefactor} + \textbf{wind\_speed\_prefactor} * v_{wind} = 11 + 19 * v_{wind} \tag{3}$$

is the evaporation coefficient θ, 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 **simulation frame rate** and **view frame rate** and make sure that **Thermodynamics→system type** is set to **None**.
prefactor	Sets the prefactor of the evaporation coefficient θ. By default, the prefactor is set to 11.
wind speed prefactor	Sets the wind speed prefactor of the evaporation coefficient θ. 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 **equilibration phase** 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 **Physics→ temperature** 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 **None**, the evaporation phase is never interrupted for fluid dynamics computation. If set to **DeletedParticles**, fluid dynamics of remaining particles is computed after a certain amount of particles have evaporated. If set to **FrameWise**, fluid dynamics is computed after each simulation frame of the evaporation phase. For the last two modes, a disrupted kinematic equilibrium state can be re-established. (It might have been disrupted during the evaporation phase where the fluid dynamics computation was switched off) .
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 **equilibration phase** is enabled (i.e. not **None**).

Table 40: Properties of Preon solver in group **Physics**→**Evaporation Solver**.

Best Practices

For the evaporation coefficient, the *Verein Deutscher Ingenieure* (VDI) recommends in its guideline[19] 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 40).

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 13.3 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

[19] VDI- Richtlinienausschuss 2089

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.

9.1.15 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
render mode	Particle rendering can be set to **visible** or **invisible**.
exclude from reflections	If enabled, this object will not be mirrored in reflective surfaces when using PREON® Renderer.
render scripted particles	Enables or disables the visualization of scripted particles. This is only relevant in combination with simulation boundary domains that script particles.
color	The base color of particles if the **coloring** is not set to any field variable, but set to **None**.
opacity	Controls the opacity of the particles. If 1, the particles are completely opaque. If 0, they are completely transparent.
coloring	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 41: Properties in group **Appearance**.

9.1.16 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
use compression	Enables or disables particle compression. Compression saves disk space, but can be computationally demanding.
max compression error	The maximum allowed compression error as a factor of the particle size. Higher compression errors allow more efficient compression. This is only relevant when compression is enabled.

serialize ID	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.
serialize Velocity	Enables or disables the serialization of particle velocities. Particle velocities are required in post-processing for most sensors.
serialize Pressure	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 42: Properties in group **Serialization**.

9.1.17 CSV export

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

9.2 Periodic boundary solver

Periodic boundary conditions are used to connect two remote regions of a 2D or 3D domain instantly. This type of boundary can significantly reduce the cost of simulating systems with infinities or symmetries. It is achieved by transporting particles traversing one of the boundary planes to the other plane placed in another location instantly while maintaining their state (velocity, pressure, and temperature). Particles can traverse the boundary bidirectionally.

The work flow to set up a scene containing periodic boundary conditions is the following. Set up your simulation using a Preon solver as usual. Insert the boundary solver by adding a *Add→Solver→Periodic boundary solver* object and connect the *Particles* output slot of the Preon solver to *Particle* input slot of the periodic boundary solver. The solvers are now set up, but we still need to define the domain. Add two *Add→Boundary Domains and Conditions→Periodic boundary plane* objects and connect their *PeriodicBoundaryDomain* output slot to the periodic boundary solver. Now, all connections are drawn and you just need to position and orient the planes at the border of your domain. All particles that enter one plane will leave the other, and vice versa. Please note that the normal axes on the planes point outwards the domain, i.e., the fluid is supposed to hit the planes on the side *without* axis in the direction of the normal axis. The fluid exits the other plane in the direction opposite to its normal axis.

9.3 Experimental: Solid volume solver

By default, the solid objects in PreonLab are sampled only on the outer surfaces. By adding an additional **Solid Volume Solver** and a **Solid Volume Source** (see section

Section 10.4), the sampling can be done over the entire volume occupied by the solid and the heat dissipation within a solid volume is accounted for. For more details on how to set this up, please refer section Section 10.4.1. The solver settings are described in the following sections.

9.3.1 General settings

Property	Unit/Type	What it does
particle size	m	The particle size controls the resolution of the solid. The volume of a solid particle is (particle size)^3.
dimension	-	The default is **ThreeDimensional** which means that PreonLab simulates in 3D. If this property is set to **TwoDimensional** or **OneDimensional**, heat diffusion is restricted to two dimensions or one dimension.
individual frame rate	On/Off	If enabled, a simulation **frame rate** can be set which is different to the **simulation frame rate** provided in the **Scene** object. Consequently, particle data of this solver are written to disk less or more often than those of other solvers. For example, serializing the particles of a solid volume solver with a lower frame rate than the fluid solver might be acceptable due to the comparatively slow thermal conduction in solid volumes.
frame rate	-	The individual simulation frame rate of this object. Only visible, if **individual frame rate** is enabled.
rest density	kg/m^3	The density of the solid.

Table 43: Solver properties.

Note that the restriction on the solver to lower numbers of dimensions is done along predefined axis:

- Setting the solver to **TwoDimensional**, restricts the position of the solver particles to $z = 0$. The other two coordinates are left unchanged.

- Setting the solver to **OneDimensional**, restricts the position of the solver particles to $x = 0$ and $z = 0$. The y-coordinate is left unchanged.

Particles are not automatically projected to fit the restrictions on their position. This means that when using a volume source, which is a 3-dimensional object, to create particles for a 2d solver, make sure that it intersects the xy-plane in the area where you would like to create the fluid.

The view and other objects are not affected by setting the solvers to a lower dimension. For example, you can still rotate the view in all 3 dimensions.

9.3.2 Thermodynamics

Property	Unit/Type	What it does
heat capacity	$\mathrm{J/(K\,kg)}$	The specific heat capacity of the substance on a per mass basis, i.e., the isobaric mass heat capacity. The default is $460\,\mathrm{J/(K\,kg)}$, which is the value for cast iron at 25 °C.
thermal conductivity	$\mathrm{W/(m\,K)}$	The thermal conductivity κ is the property of a material to conduct heat. The default is $55\,\mathrm{W/(m\,K)}$, which is the value for cast iron at 25 °C.
implicit	On/Off	If enabled, heat conduction is computed using an implicit formulation which is computationally 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 46.

Table 44: Properties of Solid volume solver in group **Physics→Thermodynamics**.

Property	Unit/Type	What it does
conductivity mean	-	If set to **Arithmetic**, the arithmetic mean of the conductivity values κ of the participating media (solid and fluid) is taken as the effective conductivity at the interface. If set to **Harmonic**, the effective conductivity is the harmonic mean computed as $\left(4 \cdot \kappa_{solid} \cdot \kappa_{fluid}\right) / \left(\kappa_{solid} + \kappa_{fluid}\right)$.

Table 45: 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 maximal 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 **AvgResidual**, the solver does iteratively solve for the temperature field until the average residual of all particles is below the user-defined value (see **tolerance**). If set to **MaxResidual**, the maximum residual of single particles is taken into account, i.e., it is ensured that the residual of each particle is below **tolerance**.
tolerance	-	The tolerated residual. A higher tolerance reduces the number of iterations and shortens the computation time.

Table 46: Solver properties in **Physics**→**Thermodynamics**→**Thermo Solver**.

9.3.3 Adaptive particle size

The solid volume particles close to the surface can be refined, in order to achieve an even more accurate heat transfer. The refinement is applied a bit differently from the refinement of the usual **Preon Solver**, described in sec. 9.1.10. Anyway it is recommended to read first the section about the refinement for the **Preon Solver** before applying the refinement for the **Solid Volume Solver**.

First you need to insert one or two refinement solvers, depending on if you want to refine 1 or 2 levels. This can be achieved via the right-click action on the **Solid volume solver** *Refinement* → *Create refinement solver (level 1)* or *(level2)*.

Next, insert a **Solid Volume Source** and connect it via the *ParticleEmitter* connection with the finest solver (i.e. the ”level 1” or ”level 2” solver, depending of whether you have added one or two refinement solvers. This results in having a layer of particles with a fine particle size near the solid´s surface and getting coarser particles toward the solid´s center, see fig. 36.

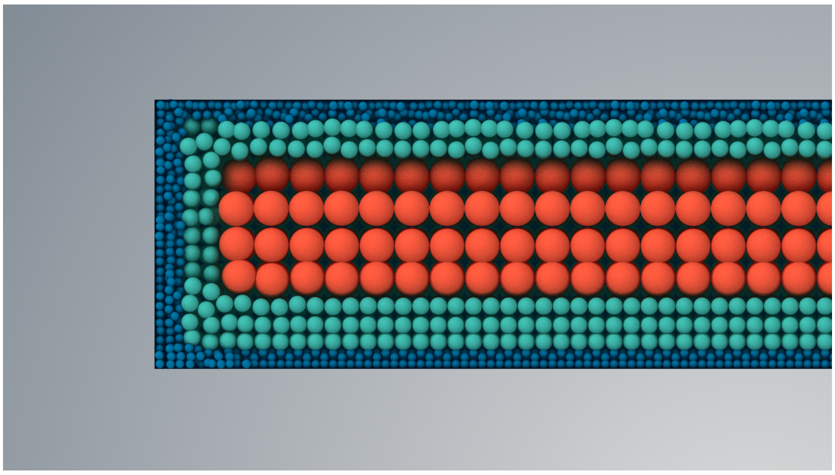


Figure 36: Particle rendering for a cube filled with fine and coarse solid volume particles.

9.3.4 Known limitations

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

9.4 Experimental: Void solver

The **Void Solver** can be used to apply a force, which can close holes in the fluid phase. This feature is specifically developed to improve the functionality of the **Outflow domain**. For this reason, we recommend to use the **Void Solver** only for scenes, in which the target flow rate of the **Outflow domain** is not reached due to the voids. The void solver particles should always be covered with the fluid, in order to avoid non-physical behavior. In order to use the **Void solver**, you need to manually connect it to a **Volume Source**. The emitted particles have fixed positions in space and will not get accelerated by gravity or other external forces. Moreover, note that while void solver particles have an influence on the fluid particles, fluid particles does not have any effect on them. Please contact the support if you encounter problems when working with this feature.

Property	Unit/Type	What it does
void force factor	–	Determines the strength of the force by which the fluid particles get pulled into the voids covered by the void solver particles.

Table 47: Additional Void Solver properties.

9.5 Snow solver

The snow model implemented in PreonLab is built upon the model proposed by Stomakhin, Schroeder, Chai, Teran, and Selle.[20] 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.

[20] 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`.

This model lets you for example simulate car snowing scenes as shown in Figure 37. 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.

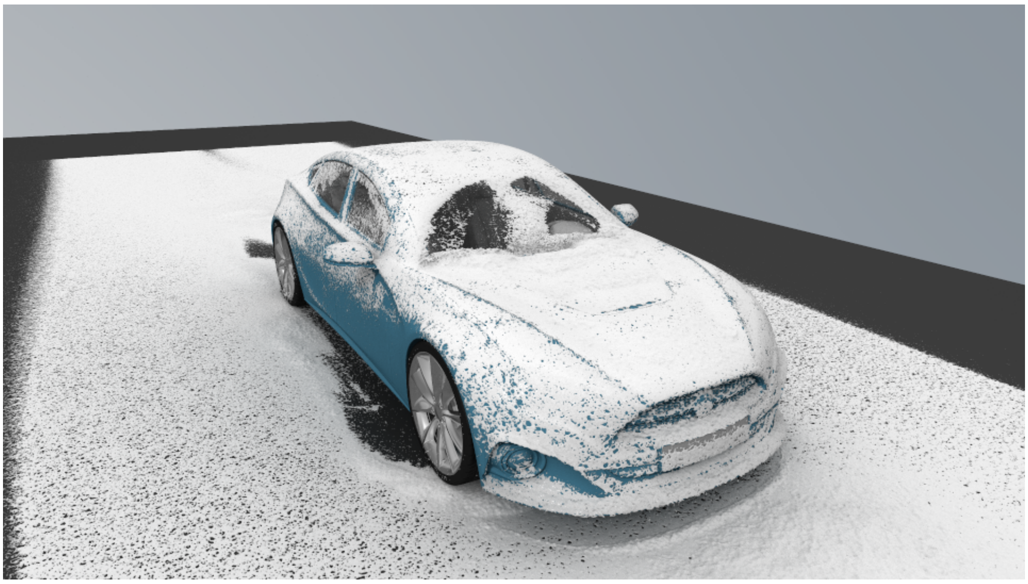


Figure 37: Car snowing simulation with moving wipers at a particle size of 5 mm visualized with Preon renderer using global illumination and the material preset for snow.

Please refer to Section 9.5.3 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 9.5.1.

9.5.1 Properties

Table 48 shows the properties of the snow solver. Additional properties are shared with Preon solver. The shared properties are for example the particle size of the snow as well as friction and adhesion at the boundary.

Property	Unit/Type	What it does
rest density	kg/m^3	The initial density of the snow. Together with the **Poisson's ratio** and **rest density**, this determines the purely elastic stress response. A lower **rest density** makes the snow stiffer, thus requiring a higher **Young Modulus** 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^3.

Young modulus	Pa	Together with the **Poisson's ratio** and **rest density**, 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 **Young modulus** and **rest density**, 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 48: 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 **Bicgstab** solver is recommended to ensure convergence. *This property is experimental and only visible if experimental mode is activated.*
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.

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

Table 49: Properties of the linear elastic solver of the snow solver in the group **Elastic Solver**.

Property	Unit/Type	What it does
corr. for vel. grad.	On/Off	Decides if the velocity gradient is computed using the corrected kernel gradient.
corr. for Cauchy accel.	On/Off	Decides if the Cauchy acceleration is computed using the corrected kernel gradient.
recalculate plastic det.	On/Off	If enabled, the plastic deformation is computed based on the current configuration instead of being accumulated over time.
use preconditioner	On/Off	If enabled, the linear system is solved using a preconditioned method. In general, a preconditioned solver converges with less iterations which saves computation time. If you expect few iterations each time step, disabling this property might improve computation time.
only use vol. grad.	On/Off	If enabled, the boundary particles only contribute to the volume change of a particle but not to the shear deformation.
clamp pressure	On/Off	If enabled, the pressure mirrored at the boundary is clamped to be positive.
boundary model	Stress/ Impulse	Impulse features a boundary model enabling coefficient of restitution behavior. All solids in the scene will get a new property **Impulse Boundary Handling**→**Coefficient of Restitution** to model elastic response on boundary contact.

Table 50: 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.

9.5.2 Snow Parametrization

A structured work flow 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 51 provides some parametrizations we found helpful. These can also be accessed in PreonLab via right-click on the snow solver in the menu *Set preset*. The corresponding snow behaviors are also shown in Figure 38.

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 51: Some helpful snow parametrizations.

9.5.3 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 48. However, the friction and adhesion to the boundary are

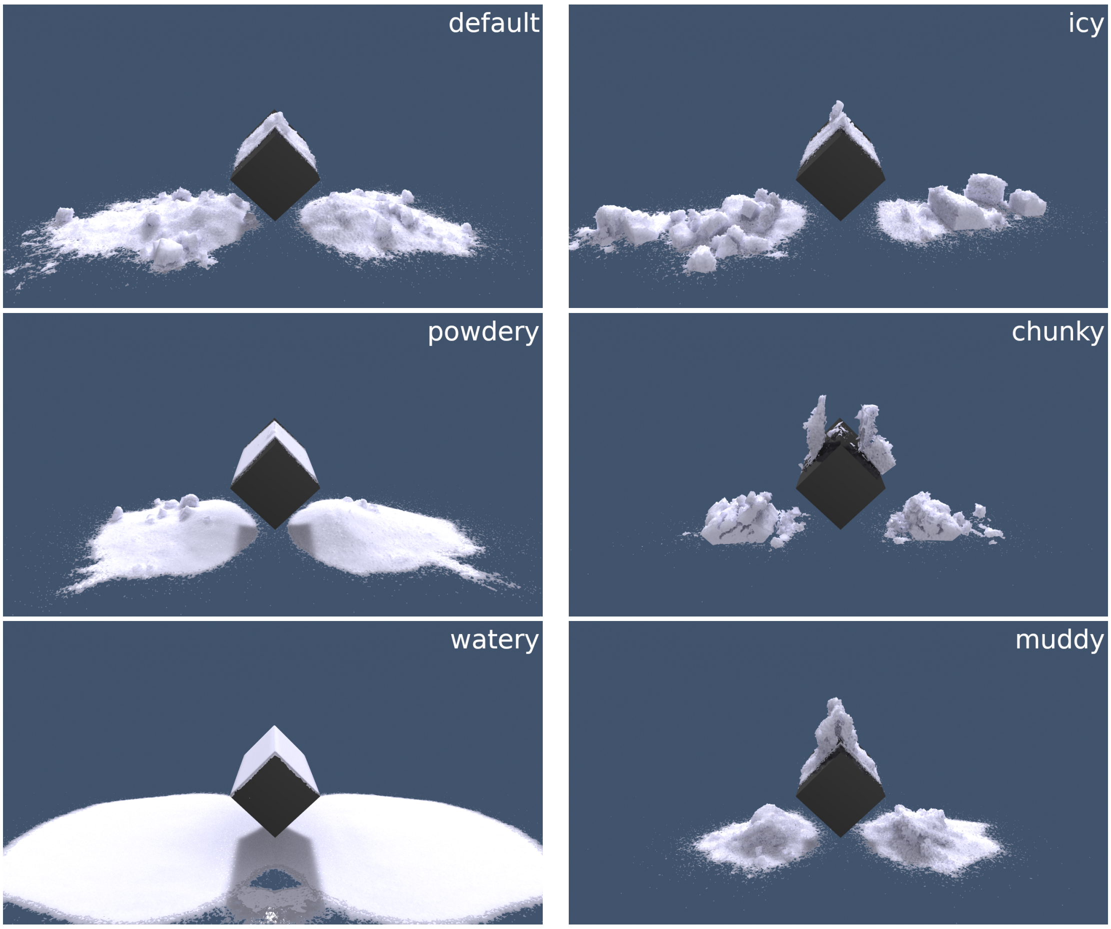


Figure 38: A block of snow falls onto a wedge. The snow used here has a minor deviation from default settings by using a rest density of 350kg/m^3. The different snow types out of Table 51 are compared.

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 **time step**.

9.6 Preon mesher

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

9.6.1 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

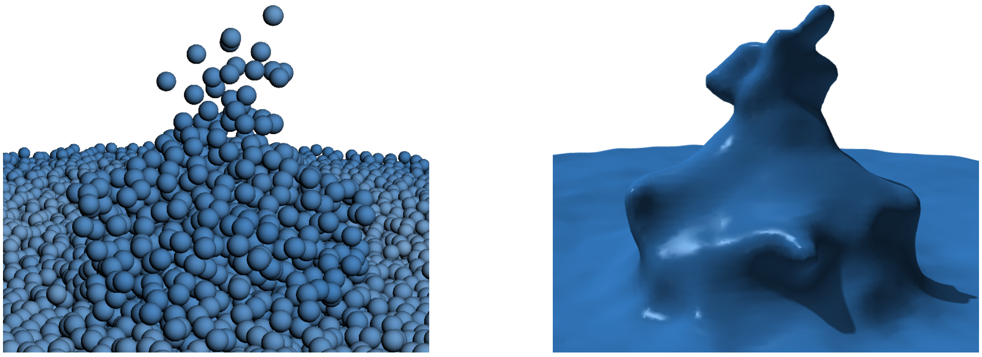


Figure 39: Generating a surface mesh from particles.

the connection editor. If there is only a single fluid, the mesher is already connected to the fluid by default.

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.

9.6.2 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.

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 **LOD max normal deviation**.
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 **LOD max error multiplier**.
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 52: 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 53: Properties in group **Distance field**.

9.6.3 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 properties listed in Table 54 are shared by both, volume and area sources.

For the user, it is sometimes difficult to predict the number of particles a source will generate. Generating a massive amount of particles may cause a system freeze or crash on some systems. The **emission particle limit** explained in Table 54 helps to prevent this.

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 million particles per emission. The default limits are 100, that means 100 million particles per emission, for a volume source, and 1, that means 1 million particles per emission, for an area source. Before generating the actual particles for a source, PreonLab estimates how many particles will be generated. If this estimate is higher than the **emission particle limit**, the error message *Estimated number of generated particles exceeds user-specified limit* is printed in the message window and no particles are generated at all.
temperature	K	Sets the initial temperature for the generated particles.

Table 54: 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 properties listed in Table 55 control the emission and are available for all area types:

Property	Unit/Type	What it does
area type	-	Determines how the source area and the outflow direction is specified. You can choose between **Seedpoint**, **Rectangle**, **Circle**, **FlatJet** and **Cone**. 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	On/Off	Enables / disables the generation of a finite amount of fluid volume controlled by the **volume** property.
volume	m^3	Sets the total volume that is emitted by the source. Only available if **finite volume** is enabled.
inflow unit	-	Sets the unit to be used for the inflow when using continuous emission. You can choose either **velocity** or **volume flow rate**. 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 **precipitation height** or **volume flow rate**. Please note that the magnitude of the selected unit will not be updated if you change the scale of the source.
emit velocity	m/s	The initial velocity of generated particles. When using continuous emission, this is only available if **inflow unit** is set to **velocity**. When using rain emission, you can always specify the velocity. When **velocity profile** is set to **PipeFlow**, you can choose to set the **averageVelocity** of the pipeflow as **inflow unit**.
correct area discretization	On/Off	If enabled, the outflow velocity will be adapted slightly to counter the error introduced by the area discretization. This results in a more accurate volume flow rate that is also more resolution independent. This property is only relevant if the **inflow unit** is set to **velocity**.
script nearby particles	On/Off	The area source always scripts the velocity of newly emitted particles. If this option is enabled, the source will also change the velocity of other nearby particles to ensure a stable outflow.
volume flow rate	m^3/s	The input fluid volume generated by the source per second. Only available if **inflow unit** is set to **volume flow rate**.

precipitation height	mm/min	The precipitation height in mm per min which equals L per m^2 per min. This parameter defines the amount of volume generated per square meter. Only available when using rain emission and **inflow unit** is set to **precipitation height**.

Table 55: 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

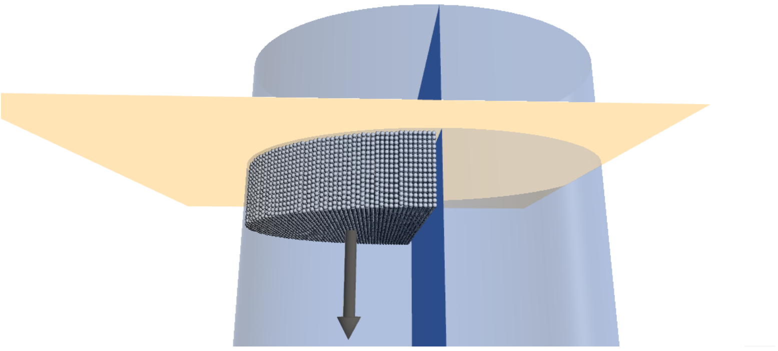


Figure 40: In this example, the emit area is bounded by a cylinder and a plane.

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 the the seedpoint without going through the solid boundary. Thereby, only solid objects connected to the area source via the *TriangleMesh* connection are considered. Figure 40 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	Unit/Type	What it does
static source area	On/Off	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 56: Properties in group **Seedpoint area**.

Flat jet

The **FlatJet** area type allows to emit fluid in a defined angle (see figure Figure 41. 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	Unit/Type	What it does
fan angle	°	The spray angle of the source in degrees. Has to be a value between 0 and 180 degrees.
emission radius	meter	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 **fan angle**.

Table 57: Properties in group **Flat jet area**.

Cone

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

Property	Unit/Type	What it does
cone angle	°	The opening angle of the cone in degrees. Has to be a value between 0 and 180 degrees.

Table 58: Properties in group **Cone area**.

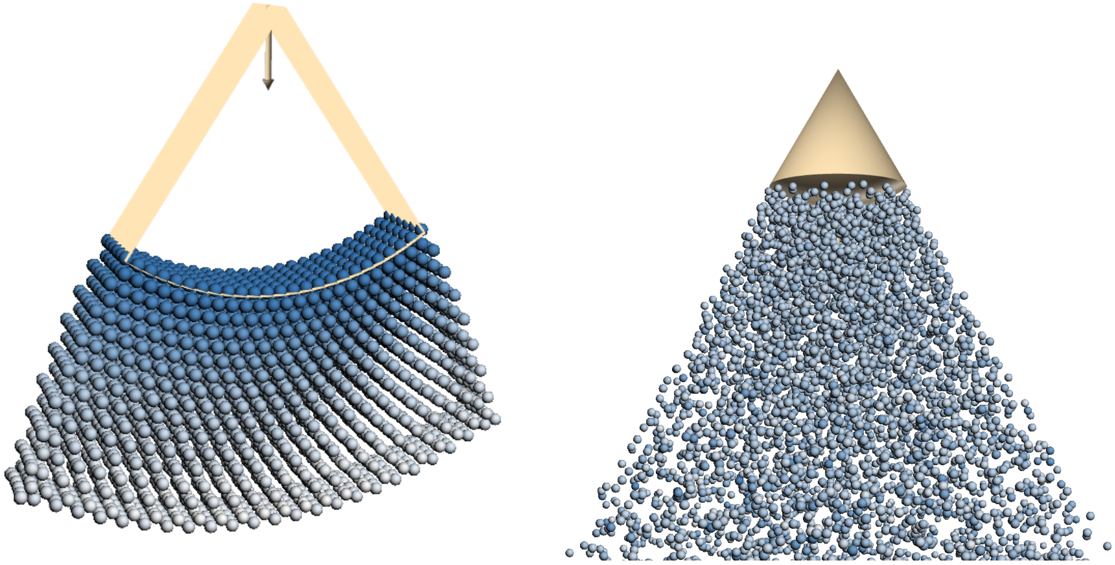


Figure 41: Two area sources with different emission and area types. Left: continuous emission and area type set to **FlatJet**. Right: rain emission and area type set to **Cone**.

Pipe flow velocity profile

A laminar pipe flow velocity profile can be set on a circular **Area Source**. The following equation will then be used to set the velocity of a new particle:

$$u(r) = 2V_{avg}(1 - r^2/R^2)$$

where r, $u(r)$, V_{avg}, and R are respectively the distance between the center of circular source and the new particle, the resulting velocity as a function of r, the average velocity, and the pipe radius. In order to use this functionality, change the **velocity profile** property to **PipeFlow**. Note that **area type** and **emit type** will be respectively set to **Circle** and **Continuous**. Furthermore, those two options are then hidden because they cannot be changed in this configuration. The average velocity of the inlet can be adjusted by setting the **inflow unit** to **averageVelocity** and by then adjusting the **emit velocity**. The other option for the **inflow unit** is **volumeFlowRate** where the corresponding average velocity is calculated using the diameter of the area source.

10.1.2 Arbitrary velocity and temperature profiles

Point Cloud Resource objects can be used to set the initial velocity and/or temperature of the incoming particles. A velocity field can be imported by the procedure described in Section 17.6.3. The velocity profile can then be set by connecting the *VelocityField* slot of the **Point Cloud Resource** object to the *TensorField* slot of the **Area source** object. The velocity vectors contained in the **Point Cloud Resource** are projected in the direction normal to the area source, yielding the emission speeds, respectively. For setting the temperature at the inlet, first import a temperature profile (see Section 17.6.1), then connect the *TemperatureField* slot of the **Point Cloud Resource** object to the *TensorField* slot of the **Area source** object. Once connected and when the simulation starts, a blue box will be generated to visualize the location and the domain of influence of the imported point cloud. When using a **Point Cloud Resource**, the values at the current particle location are obtained by interpolating the data from the field. For the parts of the inflow that are not covered by the **Point Cloud Resource** object, the default values for velocity and/or temperature will be used, respectively.

As the data is interpolated from the field samples, the domain of influence can be larger than the volume described by the original point cloud. In order to restrict the influence to the axis-aligned bounding box of the imported point cloud, enable the property **aabb from sample points**.

Best practice: For the best results, make sure that space sampling of the imported field is sufficiently fine to reduce the interpolation errors.

10.2 Volume source

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

Property	Unit/Type	What it does
emit frame	-	The frame at which the volume source emits its particles. The resulting time depends on the simulation frame rate.
emit velocity	m/s	The initial velocity of generated fluid particles.

fill method	-	Specifies *how* the volume is sampled with particles. Fill method **uniform** will generate particles aligned in a uniform grid. Fill method **hexagonal** 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 **quality** 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 **poisson_hybrid** is only recommended when filling objects using fill type **inside** or **seedpoint** 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. **poisson_full** 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, **poisson_surface** only samples the surface and generates no other particles.
use max volume	On/Off	If enabled, the volume source will not emit more volume than the specified limit, defined by **max volume**.
max volume	m^3	Only visible if **use max volume** is enabled. Specifies the maximum volume the source generates if a regular filling would exceed this. Otherwise, a warning will be displayed. If the entered value is smaller than the maximal possible volume, the particles will be removed from the volume source, starting from the z-direction with respect to the coordinate system of the volume source, until the specified volume is reached.

Table 59: Volume source properties in group **Settings**.

Property	Unit/Type	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 **all** will fill the source box completely with particles. Fill type **inside** 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 **outside**, which generates particles outside of objects. Fill type **seedpoint** 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). Finally, **surface proximity** will generate particles within proximity to the surface of connected objects. Thereby, the border size specifies the proximity distance.
manual border size	On/Off	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.
border size	meter	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 particle size of the fluid. When using the no-gap option in the fluid solver, it should be set to half the particle size of the fluid.

Table 60: 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

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 42 shows a box filled with fluid using seedpoint filling.

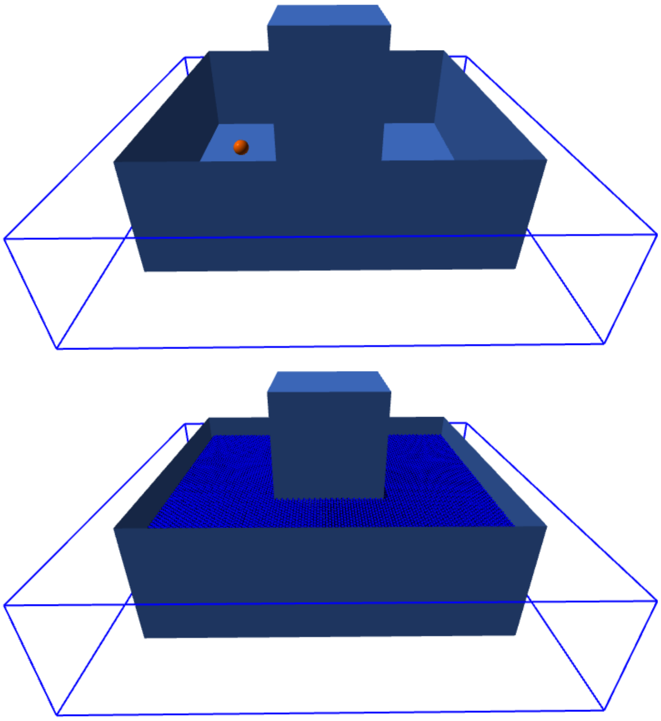
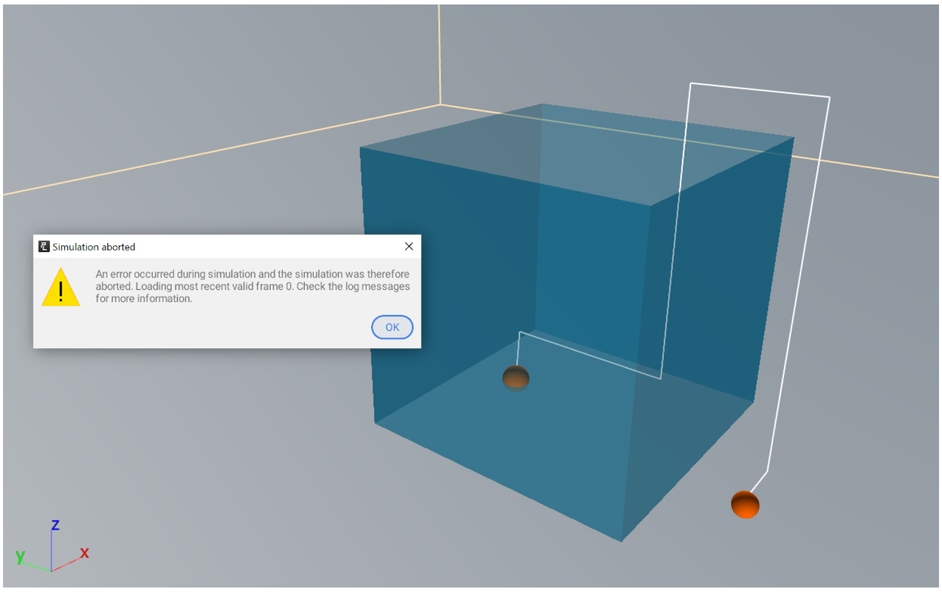


Figure 42: Top: Seedpoint (orange dot) placed in a box with obstacle in the middle. Bottom: Box filled with particles.

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, open the context menu of a volume source by right-clicking on it in the scene inspector or in the graphics window (in the graphics window the volume source needs to be selected) and choose *Add seedpoint*. A second method to insert a seed point is the following: Click *Add→Transform→Point*. In this method, you need to setup the connection manually using the connection editor (slot *Seedpoint*).

Finding holes in your geometry

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.



Simulation aborted

An error occurred during simulation and the simulation was therefore aborted. Loading most recent valid frame 0. Check the log messages for more information.

Figure 43: Error after starting the simulation because the badpoint was reached during filling. A white line indicates the connection between seedpoint and badpoint.

To specify a badpoint, open the context menu of a volume source by right-clicking on it in the scene inspector or in the graphics window (in the graphics window you need to select the volume source first) and press *Add badpoint* (the third option from below). Another possibility to specify a badpoint is to insert a **Point** object and connect it to the *Badpoint* input slot of the volume source. Figure 43 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 PreonLab adds automatically by default.

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 size. Right-click on the box in the scene inspector and choose *Adjust scale to particle size* 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 17.6.3). 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.

10.2.5 Specifying initial temperature

The initial temperature of the particles inside a **Volume Source** can uniformly be set using the **temperature** property under **Settings**→**Thermodynamics**. For an arbitrary temperature distribution a temperature field needs to be imported following the procedure presented in Section 17.6.1. You can then connect the *TemperatureField* output of the **Point Cloud Resource** object to the *TensorField* input of the **Volume Source** object. The initial temperature of the particles is obtained by interpolating the imported data. The temperature is set using this method only once when the particles are created via a **Volume Source** and handed over to the solver.

10.3 Rain source

The rain source mimics a rain like inflow.

Property	Unit/Type	What it does
emit velocity	m/s	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 $\approx 9\,\mathrm{m/s}$.
precipitation height	mm/min	The precipitation height in mm per min which equals L per m^2 per min. This parameter defines the amount of volume generated per square meter.
specify drop diameters	On/Off	Defines whether diameters of emitted drops can be set manually or not. If disabled, all emitted drops have a fixed diameter equal to the particle size of the fluid. 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} = **min drop diameter** and d_{max} = **max drop diameter**. The mean of the normal distribution is $d_{mean} = \frac{d_{min}+d_{max}}{2}$. The standard deviation is $\frac{d_{mean}-d_{min}}{3}$ while a drop will never be smaller than d_{min} or bigger than d_{max}.
min drop diameter	m	The minimum diameter of emitted drops in meters.
max drop diameter	m	The maximum diameter of emitted drops in meters.

Table 61: Properties of rain source.

10.4 Experimental: Solid volume source

A solid volume source fills the user specified solid volume(s) with particles. The properties of the particles are specified in the corresponding solid volume solver as explained in Section 9.3. Following are the properties corresponding to a solid volume source.

Property	Unit/Type	What it does
automatic resize	On/Off	If enabled, the bounding box of the volume source is fitted to the connected geometry such that its closed volume is completely filled with particles. Disable this, if the volume needs to be only partially filled.

Table 62: Solid volume source properties in group **General**.

Property	Unit/Type	What it does
temperature	K	Sets the initial temperature of the emitted particles.

Table 63: Solid volume source properties in group **Settings**→**Thermodynamics**.

10.4.1 Setting up a solid volume with thermodynamics

In PreonLab, a **Solid** object can be used to set up thermal boundary conditions on its surface to interact with the surrounding fluid particles (see Section 13.1). PreonLab also allows the user to enable the heat interaction within a solid volume and also to exchange heat between any other solvers. We start with a solid object, i.e., a geometry, already in the scene. To enable the thermodynamics for this solid, add a **Solid Volume Solver** and a **Solid Volume Source**. Connect the solid to the added solid volume source through the *TriangleMesh* slot and ensure that the solid volume source is connected to the solid volume solver through the *ParticleEmitter* slot. Properties such as density, heat capacity and thermal conductivity of the solid can be set in the solid volume solver. The initial temperature of the solid particles can be set in the **Solid Volume Source**. By default, its property **automatic resize** is set to **Off**. Switch this to **On** to automatically fill all the solids connected to the solid volume source. Once the settings are finished, starting the simulation generates the particles within the required solid(s). Please note that for a moving solid, the solid should be connected to the solid volume solver through the *Transform* slot in the connection editor, to allow the generated particles to follow the motion of the solid. By default, the solid volume solver has thermodynamics switched on. It is possible to simulate the heat transfer within a solid without adding a Preon solver. If a Preon solver is added in the scene, the interactions between the solid particles and the fluid particles are also taken into account.

10.4.2 Specifying initial temperature

For initializing the temperature of a **Solid Volume Source** object the same procedure described in Section 10.2.5 can be followed. In this case, the **Point Cloud Resource** needs to be connected to a **Solid Volume Source** instead of a **Volume Source**.

Best practice: This feature can be used to accelerate Conjugate Heat Transfer (CHT) simulations by providing a realistic initial temperature field in the solid.

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 multiple control types: **deletion of fluid particle, solid volume particles, rigid particles or all types of 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 multiple options: Choose **deleteFluidParticles** to delete only fluid particles, **deleteRigidParticles** to delete only rigid particles, **deleteSolidVolumeParticles** to delete only solid volume particles or **deleteAll** to delete all of the three types - fluid, rigid and solid particles. If you want to delete rigid particles, you need to additionally connect the *DeletionDomain* slot to the respective solid. If you want to specify a constant fluid velocity inside the domain, choose **scriptFluidVelocity**. For non-uniform particle simulations with the **continuous particle size** feature, set the type to **refine_continuous** in order to be able to specify the **target particle size** in this domain. For the old level approach, refinement is specified by selecting **refine_level1** or **refine_level2** depending on the desired refinement level. Please note that additional steps might be necessary to setup a non-uniform simulation (see Section 9.1.10 for more details).
target particle size	Specifies the size of particles that particles will be coarsened or refined to inside or outside this region (depending on property **region**. Only relevant and visible if **types** is set to **refine_continuous**.

Table 64: 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 *Transform* parents of the domain.

Table 65: 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 59 for more information.

manual border size	See Table 59 for more information.
border size	See Table 59 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 66: 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.

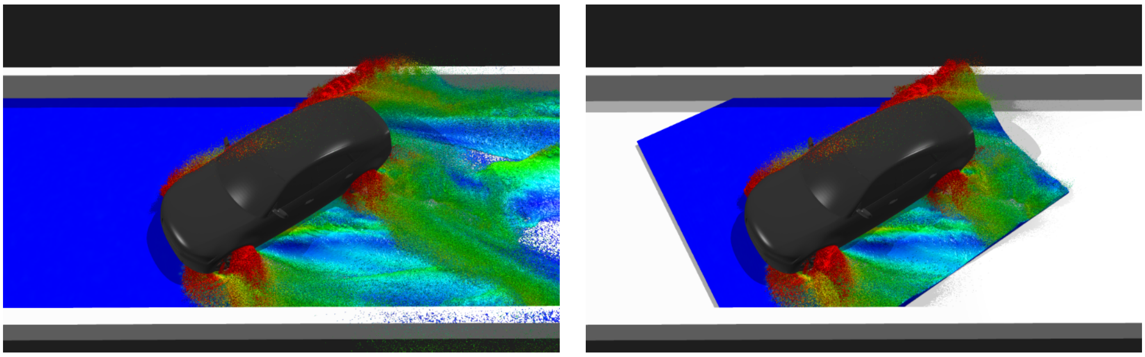


Figure 44: Use-case for simulating a restricted domain around a car to save memory and increase performance. Comparison of unrestricted (left) to restricted simulation domain (right).

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.

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

Table 67: Maximum velocity condition.

11.3 Air Object

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	K	Sets the temperature (can be varied over time via keyframes).

Table 68: 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 69. 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 **Humidity**, it is defined by the user (either as a constant or keyframed over time). If set to mode **Volume** or **TriMesh**, 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^3	Defines the volume that is occupied by the air object.

Table 69: Properties in group **Physics** → **Evaporation**.

11.3.2 Thermodynamics with air objects

The thermodynamics of the **Air Object** can be enabled by switching **Physics** → **Thermodynamics** → **thermodynamics** to on. Further properties as listed in Table 70 are then visible.

Property	Unit/Type	What it does
thermodynamics	On/Off	Defines whether the air phase thermally interacts with fluid or solids. If switched off, it is not considered in the thermodynamics computation at all. If switched on, it thermally interacts with fluids that have their **thermodynamics** switched on. 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). The below properties are visible only when this property is enabled.

heat capacity	J/(K 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 K)	The property of a material to conduct heat. The default is 0.026 (air at 25°C).

Table 70: 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 size.
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 **copy**, **project** or **interpolate**. The type **copy** copies the complete velocity of the particle to be mirrored, whereas the type **project** just uses the part pointing in normal direction of the plane. Finally, the type **interpolate** does an SPH interpolation of the velocity instead of using the velocity of the single particle to be mirrored.
particle size factor	The factor in terms of the particle size that decides the gap between a real fluid particle and its correspondingly mirrored ghost particle.
collision factor	The factor in terms of the particle size that decides the distance the mirroring axis can be shifted to avoid a fluid particle and a ghost particle sharing the same position.

Table 71: Properties of **Open boundary plane**.

11.5 Experimental: Outflow domain

The **Outflow domain** allows to prescribe a fixed outflow of fluid. 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 72).

Note: Currently, you need to manually make sure that the **Outflow domain** is placed correctly in the z-direction. To guarantee correct behavior, **Transformation**→**position**→**Z** should be set, such that the domain extends a few particle sizes into the outflowing fluid.

Property	What it does
outflow type	Defines the type of boundary condition of the outflow domain This can be **velocity**, **flowrate** or **continuative**.
min target speed	Only relevant if **outflow type** is set to **velocity**. Particles with lower speed will get this speed prescribed, keeping the direction of the velocity.
max target speed	Only relevant if **outflow type** is set to **velocity**. Particles with higher speed will get this speed prescribed, keeping the direction of the velocity.
direction deletion threshold	Only relevant if **outflow type** is set to **velocity** or **continuative**. 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 **outflow type** is set to **flowrate**. This is the flow rate of the fluid flowing through the domain and also the rate at which the domain deletes the fluid from the domain. The unit of this property is m^3/s.
flow rate correction method	Only relevant if **outflow type** is set to **flowrate**. **None** disables any correction and has the same behavior as PreonLab5.0 or lower. **adjust velocity** corrects the outflow velocity such that **target flow rate** is reached.
max velocity multiplier	Only relevant if **flow rate correction method** is set to **adjust velocity**. Sets the maximum allowed adjusted velocity as a multiple of the estimated velocity when the whole cross section area of the domain is filled. Adjust this property when the cross section area of the outflow domain is insufficiently filled with fluid and the flow rate is not reached.
shape	Allows to change the shape of the outflow domain to either **box** or **cylinder**.

Table 72: Properties of **Outflow domain**.

11.5.1 Experimental: Void solver

An outflow domain can be used in conjunction with a void solver to ensure that the target flow rate is reached. In this case the void solver will create pumping forces where other external forces, e.g. gravity, are not enough to supply the outflow domain with fluid. For more details on the void solver, please refer to Section 9.4.

11.5.2 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** and hidden. Furthermore, the **emit type** of the connected sources is automatically set to **rain** and hidden. If the **Outflow domain** is connected to several **Area sources**, the **flowrate** is distributed according to the area of each source. For Instance you have two connected sources with 4 m^2 and 1 m^2, the larger source would emit four times more particles than the smaller one. Please note that adding or deleting these connections during simulation might lead to an erroneous behavior.

11.6 Periodic boundary plane

Periodic boundary planes are used in conjunction with a *Periodic boundary solver* to set up periodic boundary conditions in you simulation. For more details on the workflow, please refer to Section 9.2.

11.7 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 45). 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.

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

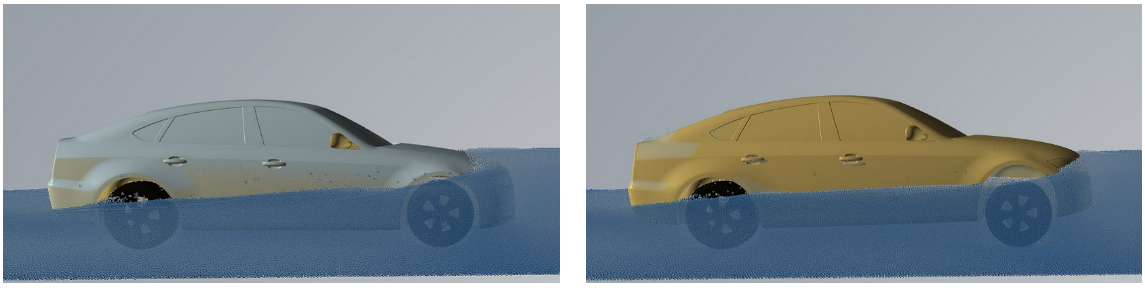


Figure 45: Comparison of a wading scenario computed without (left) and with car suspension model (right). For the simulation with car suspension model, there is no fluid on the front lid of the car, as the fluid forces acting on the car are pushing the sprung car parts upwards until the front damper is fully deflected.

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 46. The car suspension model requires the connection of wheel or axle geometry to two 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 73.

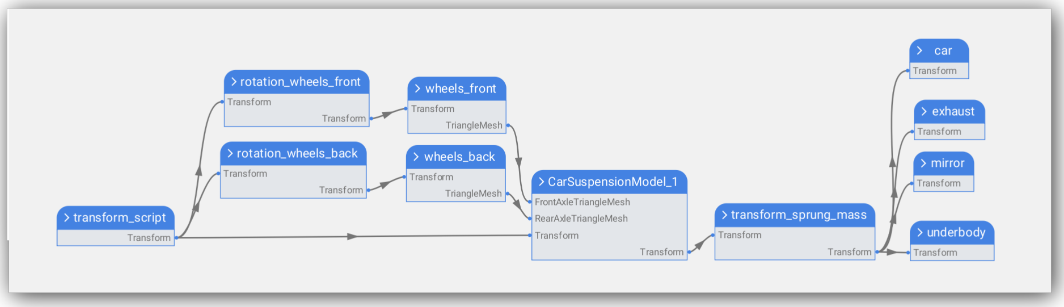


Figure 46: Example connection graph for scene with car suspension model. The **TransformGroup_1** defines the overall transformation due to velocity and orientation keyframes of the car and the wheels. The **CarSuspensionModel_1** applies forces acting from fluid onto the connected sprung car parts.

Property	What it does
Transform	The sprung car parts have to be connected via the *Transform* 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 *Transform* input slot.
FrontAxleTriangleMesh	The meshes belonging to the front axle, i.e, wheels and axle, have to be connected here. Use the connection type filter *General→TriangleMesh* if you want to filter the connection graph for this particular type.

RearAxleTriangleMesh	The meshes belonging to the rear axle, i.e, wheels and axle, have to be connected here. Use the connection type filter *General→TriangleMesh* if you want to filter the connection graph for this particular type.

Table 73: Connection slots of the car suspension model.

11.7.1 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 axle and rear axle meshes connected to the car suspension model.

Tables 74 to 76 list the properties of groups **General** and **Suspension** and respective subgroup properties. Note that all suspension property values must refer to the suspension of one wheel.

Property	Unit/Type	What it does
sprung mass	kg	The mass of all sprung parts of a car and possible additional weight, i.e, passengers or luggage.
weight distribution	-	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.
center of mass height	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 74: General properties.

Property	Unit/Type	What it does
Nonlinear Springs→ nonlinear springs	On/Off	If enabled, the spring constant can be keyframed based on the deflection of the spring. Based on the property **mapped**, the deflection is mapped to a spring rate or to a force, see Table 77.

Nonlinear Springs→ reference point	-	If set to **compressed state**, a spring deflection of zero corresponds to the spring configuration under axle load (given the gravity). If set to **uncompressed state**, 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→ 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 75: 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. This is not necessary if your spring deflection curve maps against forces measured with the car in an unloaded state. In the latter case, the car suspension model will compute a compression of the springs due to the additional weight and transform the sprung car parts accordingly.

In case you enable **nonlinear springs**, additional properties are shown in subgroup **Nonlinear Springs** (see Table 77) and respective suspension properties are shown or hidden in subgroups **Front** and **Rear** (see Table 76).

Property	Unit/Type	What it does
max. spring compression	m	Defines the maximum possible compressive deflection of a spring relative to the static deflection.
max. spring expansion	m	Defines the maximum possible expansive deflection of a spring relative to the static deflection.
spring constant	N/m	Defines the per-wheel spring constant k of the suspension and is only visible if **nonlinear springs** are disabled.
nonlinear spring force	N	Defines the per-wheel spring force dependent on the deflection of the spring via a deflection-to-force 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. This property is only visible if **nonlinear springs** are enabled and **mapped** is set to **Force**.

nonlinear spring coefficient	N/m	Defines the per-wheel spring coefficient dependent on the deflection of the spring 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. This property is only visible if **nonlinear springs** are enabled and **mapped** is set to **Coefficient**.
damper coefficient	N s/m	Defines the per-wheel constant damper coefficient *c* of the suspension and is only visible if **nonlinear dampers** are disabled.
nonlinear damper coefficient	N s/m	Defines the per-wheel damper coefficient dependent on the damper 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. This property is only visible if **nonlinear dampers** are enabled.

Table 76: Suspension properties in groups **Front** and **Rear**. They are applied to the wheels at the respective axle individually.

Property	Unit/Type	What it does
mapped	-	Select **Coefficient** if the spring deflection is mapped to a spring rate (in N/m). Alternatively, select **Force** if the spring deflection is mapped to a force (in N).

Table 77: 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 47. It matches the analytical solution perfectly.

11.7.2 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.
3. The maximum spring compression and maximum spring expansion is considered regardless of whether you provide linear or nonlinear suspension param-

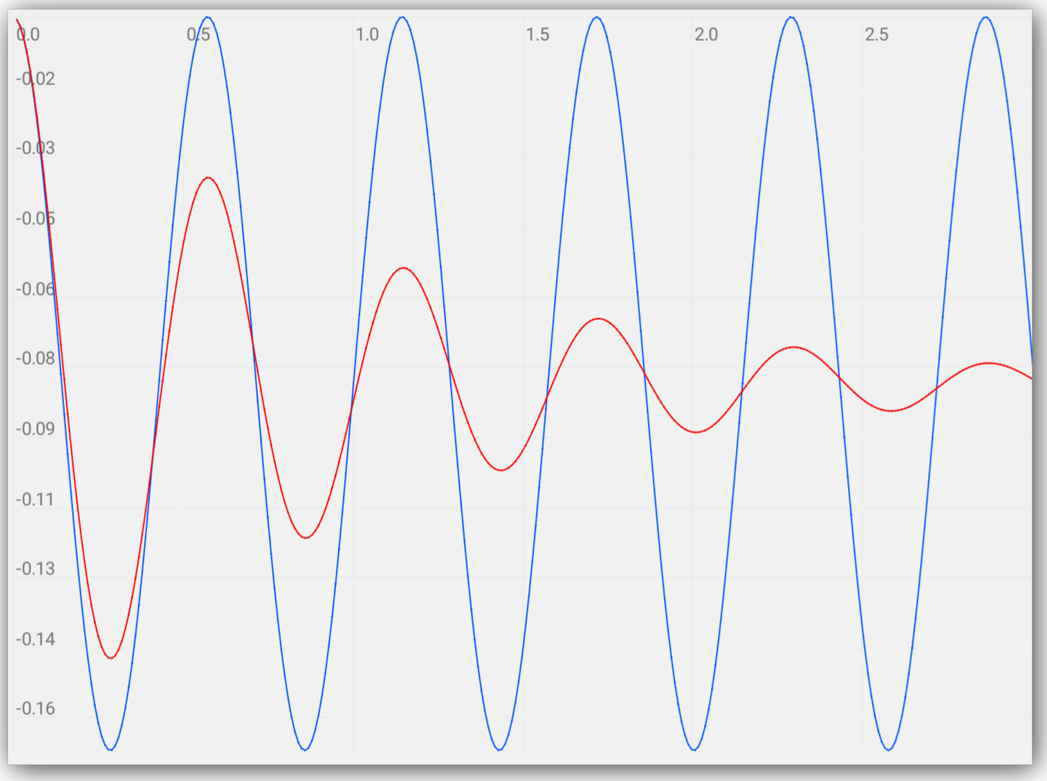


Figure 47: Gravity forces act on a car suspension model with uniform parameters at all four wheels. The graphs show the deflection of the sprung mass (in meter) for an undamped (blue) and a damped (red) configuration as a function of time (in seconds).

eters.

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 the TG.

The resulting connection graph is illustrated in Figure 46. Please consider reading the tutorial **Vehicle Water Wading** provided separately from this manual that assists you further in setting up your wading scenario.

11.8 Experimental: Vehicle simulation model

Starting with PreonLab 5.2, we introduce the **Vehicle Simulation Model**. Using this object, a co-simulation with AVL VSM™is realized. AVL VSM™is a vehicle dynamics simulation software [1] providing features like quarter-car-suspension modeling, track modeling and a virtual driver, among others. It can be used as an alternative to PreonLab's own Car Suspension Model provided. In the next sections, the **Vehicle Simulation Model** object is briefly described as well as technical prerequisites, like licensing, installation of a VSM GUI for project setup and running simulations on (Linux) clusters.

11.8.1 Licensing

AVL VSM™requires its own license. It employs the FlexNet Licensing, which is similar to the Reprise License Manager (RLM), see Section 2.2. In case a node-locked license is employed, PreonLab will find it automatically. For floating license models, some prerequisites need to be fulfilled such that PreonLab is able to find the licenses.

First, the location to the feature license has to be specified. This can be realized in two ways:

- either via an environment variable:
  `export PREON_VSM_LICENSE_FILE = <path_to_feature_license_file>`
- or by renaming the *lic* you have received to *avlvsm.lic* and copying it to `C:/Users/<USER>/AppData/Local/PreonLab/vsm/` (Windows) or `~/.config/PreonLab/vsm/` (Linux).

In a second step, the FlexNet server has to be specified, again in one of the following two ways:

- either via an environment variable:
  `export AVL_LICENSE_FILE =27001@<host_name_or_IP>`
- or by creating a file named *flextnet.txt* in the folder `C:/Users/<USER>/AppData/Local/PreonLab/vsm/` (Windows) or `~/.config/PreonLab/vsm/` (Linux) and writing `27001@<host_name_or_IP>` into the first.

Regarding the setup of the FlexNet license server itself, please contact the AVL VSM team from which you got the VSM licenses.

11.8.2 Installing AVL VSM™

In order to set up the vehicle properties, track conditions and driver profile of the AVL VSM™model, you have to install the AVL VSM™GUI installer. Note that this installer

[1] AVL VSM™Vehicle Simulation https://www.avl.com/-/avl-vsm-4-

is currently only available for Windows operating systems. We are not going into the details of the setup, but generally you start off by choosing from a list of templates the one that suits the vehicle you want to simulate the most (e.g., a compact class car) and modify the values known to you and that deviate from this template (e.g. the vehicle mass). Variations of this initial setup can be stored as individual parameter sets, each with its own name. The template and the parameter sets are stored in a project database. Contact the AVL VSM™team for manual and installer [1].

11.8.3 The Vehicle Simulation Model object

Within PreonLab, go to *Add→Boundary Domains and Conditions* and select the **Vehicle Simulation Model** to add it to the scene. An AVL VSM™license is required from the moment you set a **project path**. It will be returned as soon as a scene containing a **Vehicle Simulation Model** object is closed again. See Table 78 for additional properties.

Property	What it does
project path	Provide a path to the database (with file extension *.vsmprj*) containing the template and parameter sets you have prepared with the AVL VSM™GUI (see Section 11.8.2).
parameter set	Select the parameter set from the drop down that should be used for the co-simulation. Note that currently, only up to 20 parameter sets can be selected from one database.
start frame	Provide the simulation frame at which the co-simulation should start.

Table 78: Connection slots of the car suspension model.

11.8.4 Simulating on clusters

In order to perform a co-simulation on a cluster or on a Linux machine (i.e., on a machine without an AVL VSM™GUI installed), the following has to be ensured:

- The license server must be found. See Section 11.8.1 for details.
- The project database has to be accessible.
- The AVL VSM™redistributable files are accessible.

Regarding the second precondition, make sure that external resources of the **Vehicle Simulation Model** object are copied into the scene folder by checking the respective box in the save dialog. You can verify success by checking for the existence of a folder named *VSM* in your PreonLab scene directory.

For the latter, three options are available:

- Let PreonLab handle it. It copies the redistributable files into a sub-folder of the scene folder (i.e., *VSM_Redistributables*). Note: This requires that files within this folder are executable which might not be the case on share drives (i.e., it depends on your IT´s security guidelines)

- Copy the *VSM_Redistributables* folder of the scene to some other location where files are permitted to be executable if above option is not feasible. Then let PreonLab know via an environment variable:
`export PREON_VSM_REDISTRIBUTABLE_FOLDER =<path_to_files_folder>.`

- Similar to option 2, copy the *VSM_Redistributables* folder of the scene to some other location where files are permitted to be executable. Then let PreonLab know by creating a file *redistributable_folder.txt* in the folder `C:/Users/<USER>/AppData/Local/PreonLab/vsm/` (Windows) or `~/.config/PreonLab/vsm/` (Linux) and writing the path to the folder into the first line of the file.

Furthermore, note that on machines with Linux operating system, AVL VSM™requires the *gfortran* library to be installed. If this is not the case install it by, e.g, performing `yum install libgfortran` on RHEL systems.

11.8.5 Best practices

Please consider the following preconditions and hints when setting up a wading scene that includes a **Vehicle Simulation Model** (VSM):

1. The vector going from rear wheel to front wheel should initially be directed along the positive x-axis.
2. The vector going from right wheel to left wheel should initially be directed along the positive y-axis.
3. Connect sprung mass geometry of the car (e.g. the car body) to the VSM object by connecting the outgoing *TransformSprungMass* slot of the VSM object to the ingoing *Transform* slot of the sprung mass geometry object. If the sprung mass geometry consists of multiple **Solids** objects, add a **Transform group** in between.
4. Connect the wheel geometry of the four wheel to the VSM object by connecting the respective outgoing transform slot of the VSM object to the ingoing *Transform* slot of the respective wheel geometry object. For example, use the *TransformFLWheel* slot for the front left wheel. If a wheel geometry consists of multiple **Solids** objects, add a **Transform group** in between, respectively.

12 Tensor 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 $\mathrm{m/s^2}$ 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
air velocity	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.
apply force everywhere	Defines if the force should be applied everywhere or just in the defined box region.
drag model	Specifies how the drag coefficient and the (unobstructed) particle area is calculated. The different options are discussed in more detail in the subsections below.
Cd	The constant drag force coefficient. Only relevant if **drag model** is set to **Constant**.
terminal velocity	Specifies the terminal velocity a single fluid particle should achieve. Only relevant if **drag model** is set to **TerminalVelocity**.
force factor	Factor multiplied to the resulting drag force. This is only shown and applied when as drag model either **Constant**, **TerminalVelocity** or **AutomaticViaTerminalVelocity** is selected.
density	Density of the air modeled by the drag force. This is only relevant and shown in the **Liu Model Settings** group when **drag model** is set to **LiuModel**.

Table 79: 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 square of the particle size.

12.2.2 Terminal Velocity

The user can specify a terminal velocity that 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 square of the particle size.

12.2.3 Automatic Terminal Velocity

A formula is used to compute the terminal velocity based on the particle size. 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 square of the particle size.

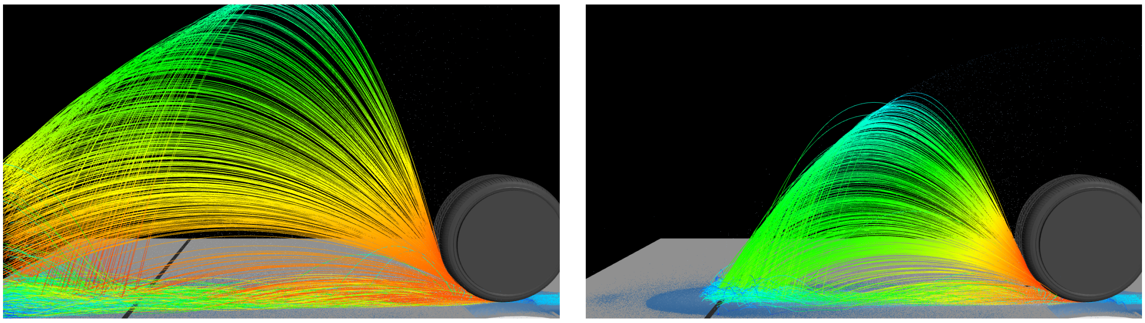


Figure 48: The fluid spray for a wheel rotating with a speed corresponding to 20 km/h is displayed using path lines. In the left image no drag force is enabled while in the right image the Liu Model is used. The result as shown in the right image matches experiments.

12.2.4 Liu Model

This drag model is based on a paper by Liu et al.[1] 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 and Acceleration Field

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 **Acceleration Field** object is similar but utilizes acceleration instead of velocity. The purpose of those objects is to couple PreonLab fluid simulations with air flow or acceleration fields simulated in another software.

PreonLab supports the import of an air flow or acceleration 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 the desired field. You have to specify the **Output Path** to the location where you want to store the converted file. Note that PreonLab uses SI units, i.e., positions are given in meter, velocities in meter per second, and accelerations in meter per second squared. **Velocity Offset** or **Acceleration Offset** can be set to a non-zero vector to transform the input field from a moving to a fixed reference frame.

[1] Modeling the Effects of Drop Drag and Breakup on Fuel Sprays, Liu A., Mather D., Reitz R., 1993

12.3.1 Static data import via the CSV format

For files in CSV format, each line in the CSV file should contain position and either velocity or acceleration for one sample point. The first line should contain a text per column that describes the corresponding value. The order of position and velocity/acceleration 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 Tensor Field*. The type of data for import can be selected with the **Tensor** drop-down menu. 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/acceleration 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 data import via the EnSight Gold format

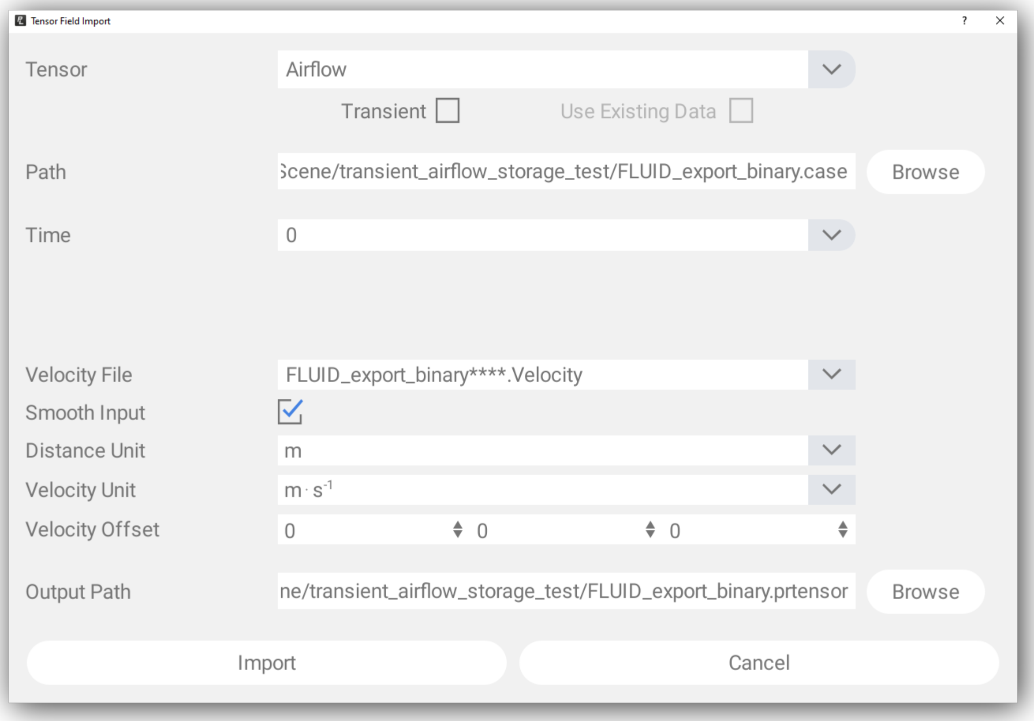
To import files in EnSight Gold format, provide an EnSight Gold *.case* file in the path field. The import dialog then displays the respective input fields (see Figure 49). First, the **Time** frame has to be specified that should be imported. For transient data, the drop-down list contains more than one entry. Second, the **Path** 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

To import transient data the **Transient** box needs to be checked. The import dialog for transient 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 50). The time range which is covered by the transient data is presented with **Start Time** and **End Time**. The range can be narrowed down. The property **Time Offset** controls the temporal offset added to the imported data. If the property **Time Duration** is extended past the original duration, the data will be looped.

Next, a **Velocity File** or an **Acceleration 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*). If the conversion was already done previously, you can check the box **Use Existing Data** to reuse the converted file.

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



Tensor: Airflow

Transient □ Use Existing Data □

Path: Scene/transient_airflow_storage_test/FLUID_export_binary.case

Time: 0

Velocity File: FLUID_export_binary****.Velocity

Smooth Input ☑

Distance Unit: m

Velocity Unit: $m \cdot s^{-1}$

Velocity Offset: 0 0 0

Output Path: ne/transient_airflow_storage_test/FLUID_export_binary.prtensor

Figure 49: The import dialog for a static **Air Flow** field represented with the EnSight Gold format.

Limitations

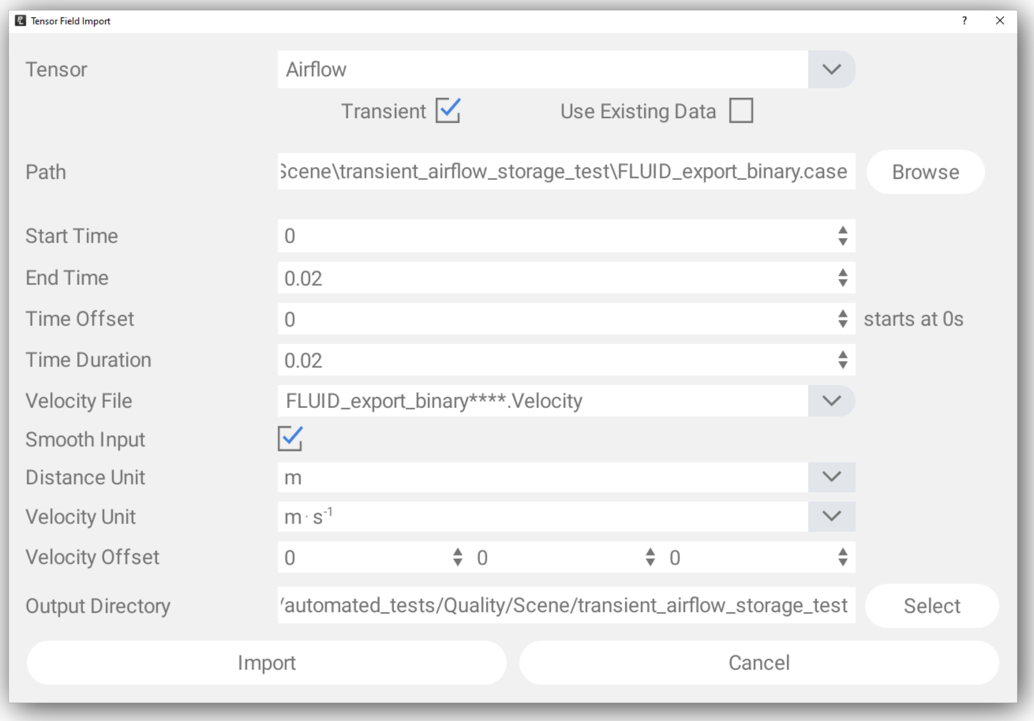
Currently, the following limitations are known:

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.
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.

Valid .case file example

A *.case* file may look like follows:

```
FORMAT
type:   ensight gold
GEOMETRY
```



Tensor Field Import

Tensor: Airflow

Transient ☑ Use Existing Data □

Path: ɜcene\transient_airflow_storage_test\FLUID_export_binary.case

Start Time: 0

End Time: 0.02

Time Offset: 0 starts at 0s

Time Duration: 0.02

Velocity File: FLUID_export_binary****.Velocity

Smooth Input: ☑

Distance Unit: m

Velocity Unit: $m \cdot s^{-1}$

Velocity Offset: 0 0 0

Output Directory: 'automated_tests/Quality/Scene/transient_airflow_storage_test

Figure 50: The import dialog for transient airflow data represented with the EnSight Gold format.

```
model: 1 ********/my_example.geo
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 imported 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. Caution is advised here, because for very large data sets this visualization may require too much GPU memory for your system. Another way to visualize the imported fields is the **Vector field visualizer** object which displays the quantities (velocities or accelerations) 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 16.15.

12.3.5 Air flow parameters

Property	What it does
air flow path	Sets the path to the *.prairflow* 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 80: 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 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 grid-based velocity storage.
velocity storage implementation	Specifies how the air flow velocity field is represented internally. **Grid** 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. **Gridless** uses a particle-based representation for the velocity field which accurately captures the imported air flow and typically requires less memory.

Table 81: Properties in group **Tensor Field Storage**.

Property	What it does
rank	Defines a rank if you have connected multiple (overlapping) air flows to the same receiving object and want to apply a subset of them exclusively in particular regions. The air flow with a higher rank is applied. Air flows with equal ranks are applied simultaneously.
air velocity	Sets the velocity that is used in areas where the grid stores no data.

Table 82: 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 83: 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 84: Properties in subgroup **Sample Points**.

12.3.6 Acceleration field parameters

Property	What it does
acceleration path	Sets the path to the *.prairflow* file created by PreonLab that stores the sample points in binary form.
acceleration offset	This allows to add an offset to all imported acceleration samples. This is for example helpful if the acceleration field 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).

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 85: Properties in group **Point cloud import**.

Property	What it does
cell size	Sets the minimal cell size in the acceleration field grid. This property is only relevant when using grid-based acceleration 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 grid-based acceleration storage.
acceleration storage implementation	Specifies how the acceleration field is represented internally. **Grid** will represent the acceleration field using a uniform grid, that allows fast access but can consume a lot of memory. Also, the imported acceleration field may not be represented accurately if the cell size is not small enough. **Gridless** uses a particle-based representation for the acceleration field which accurately captures the imported acceleration and typically requires less memory.

Table 86: Properties in group **Tensor Field Storage**.

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

Table 87: Properties in group **General**.

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

Table 88: 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 acceleration at the sample point positions:

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

Table 89: Properties in subgroup **Sample Points**.

12.3.7 Air flow 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 a **Tensor Field box** (see Section 12.4).

12.4 Tensor Field box

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

12.5 Air Pressure

The **Air Pressure** object allows you 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 90 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 **Box** or **Cylinder**.
air pressure	Pa	The pressure of the air in Pascal.
force type	-	The type of the force. Defines if the force acts into (**AirForce**) or out of the fluid (**VacuumForce**).
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 90: Properties of the **Air Pressure** object.

12.6 Heat Field

The **Heat Field** object allows you to apply a volumetric thermal source to a solver. The source term is added to the right-hand side of the heat equation. With this object, heat is generated within a volume independent of thermal boundary conditions.

The strength of the heat field can be controlled in two ways. Either the volumetric power in W/m^3 is directly provided and used in the heat equation or the total power in W is provided and the volume of the object or the connected mesh is used to compute the volumetric power for the heat equation. The grayed out **volume** property of the **Heat Field** can be used as an indicator of the volume used.

Note that for the second option, total power in W, the volume considered for the calculations is not the volume of the particles. For example, if the volume is partially filled, the volume of the heat field object or the total volume enclosed by the mesh is considered and not the volume occupied by the particles.

For the heat field to have an effect, it is required to connect it through the *ThermalPower* slot to a solver that has **thermodynamics** turned on. The **PreonSolver** and **Solid Volume Solver** currently support this feature. It is possible to control where the heat field is applied by either connecting a solid through the *TriangleMesh* slot or by using a **Point cloud resource** object in **Power** or **VolumetricPower** mode (see Section 17.6.2) connected to the *TensorField* slot.

Table 91 shows the properties of a **Heat Field** object.

Property	Unit/Type	What it does
volume	m^3	Information field about the effective volume used by the heat field. The field cannot be modified.
thermal power unit	-	Controls how the heat source strength is set. For **Watt**, the total power is set by the user and the volumetric heat source is derived from the volume calculated from the mesh (**volume**). For **Watt per cubic meter**, the volumetric heat is directly set by the user and the computed volume (**volume**) is not used.
power	W	The total power applied within the heat field.
volumetric power	W/m^3	The volumetric power applied within the heat field.

Table 91: Properties of the **Heat Field** object.

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 (particle size). 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 13.9 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, an error message will be printed to the message window. In this case, you either need to increase the fluid particle size 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).

adaptive sampling	Enables or disables the potential usage of adaptive boundary sampling which may generate boundary particles of different sizes. Only relevant if a fluid with multiple refinement levels is used and if **dynamic sampling** is enabled. Not available if the solid is connected to a sensor. Please note that visualization of the adaptive particles is not implemented yet and the **show particles** will only show the finest particles. This does not influence the quality of the simulation.
dynamic particle deletion	Enables or disables dynamic particle deletion during the simulation based on boundary domains that are connected to this solid via the slot *DeletionDomain*. 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.
sample triangle mesh	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 92: Properties of solid objects in group **General**.

Property	What it does
rigid type	Defines whether the solid acts as **scripted** (animated) object, or as **dynamic** 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 **stationary**. 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 **guided** is used in conjunction with the car suspension model, see Section 11.7.
roughness	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).

adhesion	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 **PotentialForce** 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 **PairwiseForce** 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. Not visible with **CohesionModel**→**CSS**.
contact angle	Enforces a contact angle from a fluid with higher density to a fluid with lower density, with a defined range from $[0°; 180°]$. An illustration is given in Figure 32. Only visible with **CohesionModel**→**CSS**.
density	The density of the object. Together with the volume, this defines the mass.

Table 93: 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 **boundary type** to either **temperature** or **heat flux**, see Table 94 for more details. Solid objects are surfaces that can act as heat sources or sinks for fluids or solid volumes as described in Section 9.1.12. The diffusion of the temperature inside solid volumes can be additionally achieved using the *Solid Volume Solver* and *Solid Volume Source* as described in Section 9.3. Please note that the thermal interaction of solids and air is currently not computed.

Property	Unit/Type	What it does
boundary type	-	Defines whether and how the solid thermally interacts with the fluid. If set to **None**, 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 either **temperature** or **heat flux**.
constant temperature	K	Sets the temperature of the particles. Note that this property is only available if **boundary type** is set to **temperature** and there is no temperature field connected via the *TemperatureSamples* slot that provides a temperature distribution (see Section 17.6).
constant heat flux	W/m^2	Sets the heat flux of the particles. Note that this property is only available if **boundary type** is set to **heat flux**.

Table 94: Properties of solid objects in group **Physics** → **Thermodynamics**.

13.2 Wall Functions

Solid objects can be set to utilize the wall function model for fluid particles in contact with them, as described in Section Section 9.1.13. This is done by switching on the **momentum** and **thermal** properties in the **Physics**→**Wall Function** group of the solid. For more details, see section Section 9.1.13.

Property	Unit/Type	What it does
momentum	On/Off	If switched on, computes a corrected viscosity value for fluid particles in contact based on a log-law velocity distribution.
thermal	On/Off	If switched on, computes a corrected thermal conductivity value for fluid particles in contact based on a log-law temperature distribution.
roughness factor	-	Sets the roughness parameter E in the wall law. By default, E is set to 9.8 for a smooth wall.

Table 95: Properties of wall functions in group **Physics**→**Wall Function**.

13.2.1 Known limitations

Wall functions are only compatible with the **boundary type**→**temperature** property of the solid at this time. Please verify this in **Physics**→**Thermodynamics** of the solid before starting the simulation.

13.3 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 solid's surface. The area they cover on the surface is in the resolution of the Preon solver (particle size). However, their height, i.e., the thickness of the wetting film they represent can be arbitrarily smaller than the Preon solver (particle size). It is derived from the *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.

13.3.1 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 particle size of the Preon solver.
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 96: 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.

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

Best practices

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

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

assuming we have set **film max** to 1.0. Accordingly, with *mass* $=$ *volume* $\cdot$ *density*, we end up with a maximum possible mass of $\approx 0.027\,\mathrm{kg}$ per sample point as the fluid density is $\approx 1000\,\mathrm{kg/m^3}$. 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 ≈ 0.005 in order to take advantage of the full coloring range.

13.3.3 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).

13.4 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 **dynamic sampling** is **On**, this depends on the proximity to fluid particles.
two-sided lighting	Enables / disables two-sided lighting for the rigid.
Coloring	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 **dynamic sampling** is **off**. Note: Use coloring type **wetting film** 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 98: Properties of solid objects in group **Appearance**.

13.4.1 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.

13.5 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
segments	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 99: Properties for sphere.

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

Table 100: Properties for cylinder.

13.6 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 13.6.1), 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), .stp (STEP ISO 10303-21), .step (STEP ISO 10303-21), .prmesh (PreonLab's mesh format).

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 **Mesh_Independent**, which generates a sampling independent of the mesh structure. As an alternative, there exists the **Mesh_Thinned** 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 particle size.

Table 101: Properties for Mesh.

13.6.1 Mesh resource

A Mesh resource object loads a mesh file from disk and provides an output slot in the connection editor for each contained sub-mesh (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 sub-meshes and it may be necessary to represent these sub-meshes 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 sub-meshes separately. During the import, a Mesh object is created for each exposed sub-mesh.

Property	What it does
mesh file	The path to the mesh file.
units	Rescales the mesh if anything different than meter is specified. Choose the unit that was used to model the mesh in order to rescale it for PreonLab. This property is read-only for mesh resources pointing to a ‘prmesh‘ file, as they are stored always in meters. Please use the scale functionality to rescale the connected rigid(s).
number of triangles	The total number of triangles belonging to the rigid(s), connected to the corresponding mesh resource. This is a read-only property. It can be of use when comparing different tessellation settings for an imported STEP file.

number of vertices	The total number of vertices belonging to the rigid(s), connected to the corresponding mesh resource. This is a read-only property. It can be of use when comparing different tessellation settings for an imported STEP file.

Table 102: Properties for Mesh resource.

STEP file

Starting from version 5.2, PreonLab supports the STEP file format, which is a widely used format in **C**omputer-**A**ided **D**esign (CAD) representing a mesh in terms of raw, non-tessellated geometry. To be able to further process the data coming from a STEP file, PreonLab needs to tessellate it. Table 103 lists the properties used to guide the tessellation algorithm. These properties can be set in the import dialog when importing a STEP file. The choice of the tessellation properties will have an effect on the resolution of the resulting mesh. If you want to experiment with different tessellation settings, we recommend you to import the STEP file in question with the option ”Resource object” selected. This would enable you to re-tessellate an already imported STEP file by right-clicking the corresponding mesh resource and clicking the ”Re-tessellate” menu item. Be aware though, that importing STEP files is an expensive operation, and the import times would be magnitudes larger than importing a mesh with formats such as OBJ, PLY and STL.

Property	What it does
linear deflection	The linear deflection specifies the distance between a curve and its tessellation. Also known as Chordal distance. This is a read-only property. If you want to re-tessellate an imported STEP file, please use the ”Re-tessellate” right-click action on the corresponding mesh resource.
angular deflection	The angular deflection limits the angle between subsequent segments in a polyline. This is a read-only property. If you want to re-tessellate an imported STEP file, please use the ”Re-tessellate” right-click action on the corresponding mesh resource.

Table 103: Properties in group **Tessellation** only visible for STEP file.

PRMESH file

Starting from version 5.2, PreonLab adds its own mesh format - PRMESH. This is a binary file format for storing triangulated mesh data in meters (as PRMESH files have a fixed unit, the **unit** of the mesh resource representing the PRMESH file is read-only), which preserves the sub-meshes with their names coming from the original file. The format offers faster loading times (compared to conventional mesh file formats) due to its data layout tailored for PreonLab and small file size due to its binary representation and the use of compression. You can convert a mesh file on import to the PRMESH file format using the ”convert to ’prmesh”’ toggle in the import dialog, or by converting an already imported mesh through right-clicking its mesh resource and choosing the ”convert to ’prmesh”’ menu item. Note that you can only convert meshes, which are represented by mesh resource objects.

When converting a file to the PRMESH format, you have the option to apply compression (the ”compress” toggle in the import dialog). The use of compression would decrease the file size further, approximately by a factor of 3, but might increase the loading time compared to PRMESH without compression by approximately 40 percent.

13.7 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. Table 104 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 **off**, only the first mesh sample is loaded from the file. This property is automatically set to **off** or **on** 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 **off** 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.
animation time offset	This property allows to specify a time offset (in seconds) for the animation loaded from the Alembic file.
animation time factor	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 104: Properties for **Alembic Mesh**.

13.8 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 particle size 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.

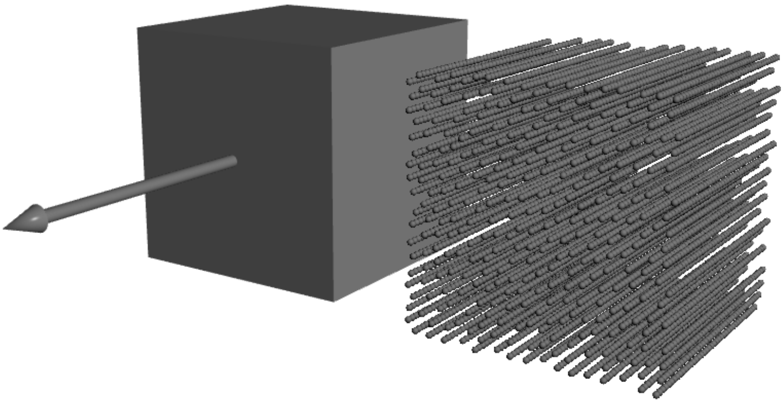


Figure 51: Porous rigid with indicated direction of porosity/channels (left). A porosity of 0.8 visualized by setting **Appearance**→**show particles** to **On** (right).

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

Table 105: Properties for porous rigid.

13.8.1 Best practice

The porous rigid allows fluid particles to pass a geometry that would otherwise be impenetrable due to the chosen fluid particle particle size. 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.

13.9 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. Is 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.

13.9.1 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.

Is 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.

13.9.2 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.

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 106, general properties of a rigid body object are shown which are also relevant for scripted objects. In Table 107, object properties are shown that are only relevant for dynamic rigid bodies. More settings that are specific to PreonLab's particle-based rigid body solver are shown in Section 14.3.

Property	What it does
rigid friction	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.
linear velocity	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.
angular velocity	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.

ignore in rigid body simulation	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 106: General rigid body settings in the group **Rigid Body Settings**.

Property	What it does
two-way coupling	Defines whether the object is influenced by fluids.
volume	Shows the current volume of the object.
mass	Shows the current mass of the object. This is a read-only property which depends on the **density** and **volume** of the object.
use custom inertia tensor	If enabled, a custom inertia tensor can be defined using the **custom inertia tensor** property.
inertia tensor	The unrotated inertia tensor of the object. Note that this is only correctly calculated when starting the simulation. This property is only shown if **use custom inertia tensor** is disabled.
custom inertia tensor	The unrotated inertia tensor of the object. Note that PreonLab does not ensure that the values of **mass** and **custom inertia tensor** have a correct relation to each other. Accordingly, if you adapt **custom inertia tensor**, you probably also need to adapt **manual volume** and **density** of the rigid object to get the correct corresponding **mass**. This property is only shown if **use custom inertia tensor** is enabled.
Constrain DOF	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.
use manual volume	Only shown if the solid is of type Mesh. If true, the user can manually specify a volume using property **manual volume** instead of using the automatically calculated one.
manual volume	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 107: Properties of rigid body objects in group **Dynamic Settings** which are 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 13.9 for more information.

14.2 Volume

Volume and mass are required for rigid body dynamics and are computed automatically based on the mesh by default. However, it is possible to override these computed values by enabling **use manual volume** and entering the desired value (see Table 107). In case you want PreonLab to compute the volume once more, switch off **use manual volume** again and use the right-click action *Compute volume*.

14.3 Particle-based rigid body solver

The particle-based rigid body solver 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.

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 108.

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 particle size. 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 **max. time step** 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 108: 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 particle size.

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 17.10).

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.

far clip	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.
focus distance	The distance from the camera position to the camera pivot (only relevant when you rotate the camera using the mouse).
orthographic	Toggles between orthographic and perspective projection.
fixed up axis	Defines whether the up axis of the camera is fixed. Fixing the up axis is recommended for more stable camera control.
locked	If on, the camera can not be manipulated per mouse. This property can also be changed by right-clicking the camera in the *Scene Inspector* and clicking on *Lock control*.

Table 109: Properties for **Camera**.

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

Table 110: 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 due to OpenGL limitations only up to five active clipping objects per camera are supported.

Property	What it does
shape type	Defines the shape of the clipping object which can be either an infinite *Plane* or a finite *Box*.

flipped	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 111: 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
color	Sets the color of the light, located in the Appearance group.
intensity	Scales the intensity of the light.
casts shadows	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 112: 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
photon mapping	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.
shadow softness	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 113: Directional light properties.

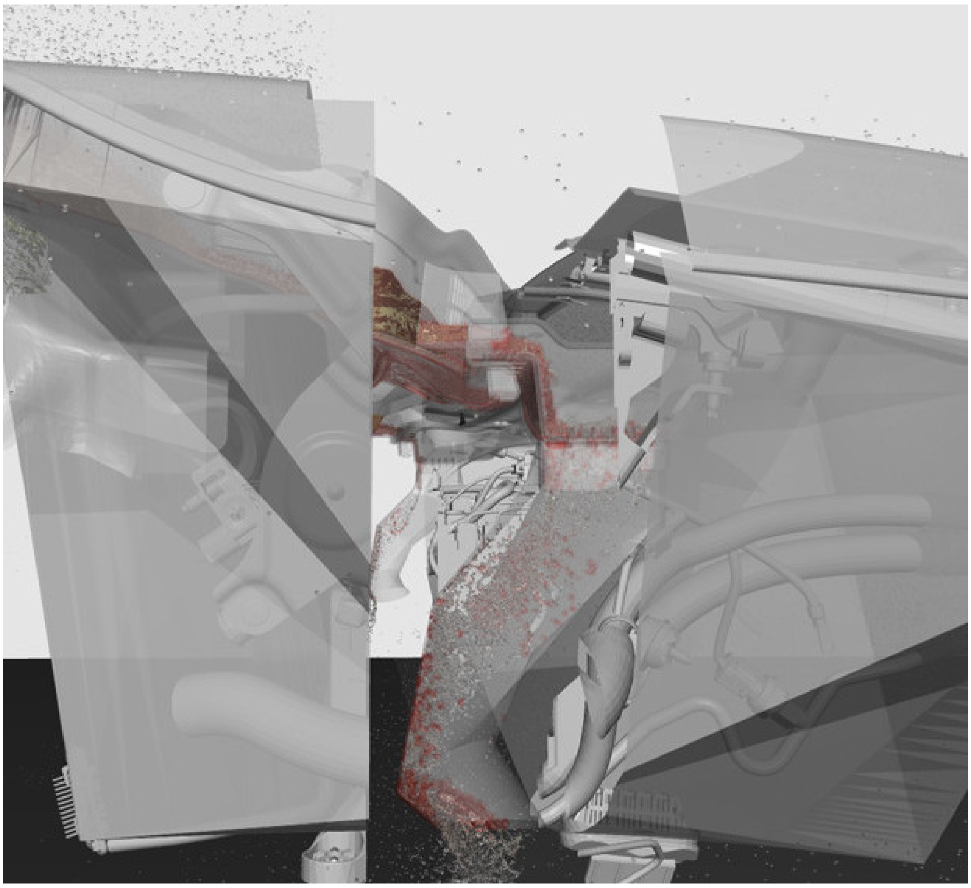


Figure 52: *Box-shaped clipping object* clips some parts of the engine compartment. Water is not clipped. Wet surfaces are highlighted in red by wetting sensor. Image is rendered with Preon renderer.

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**. Furthermore, 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.

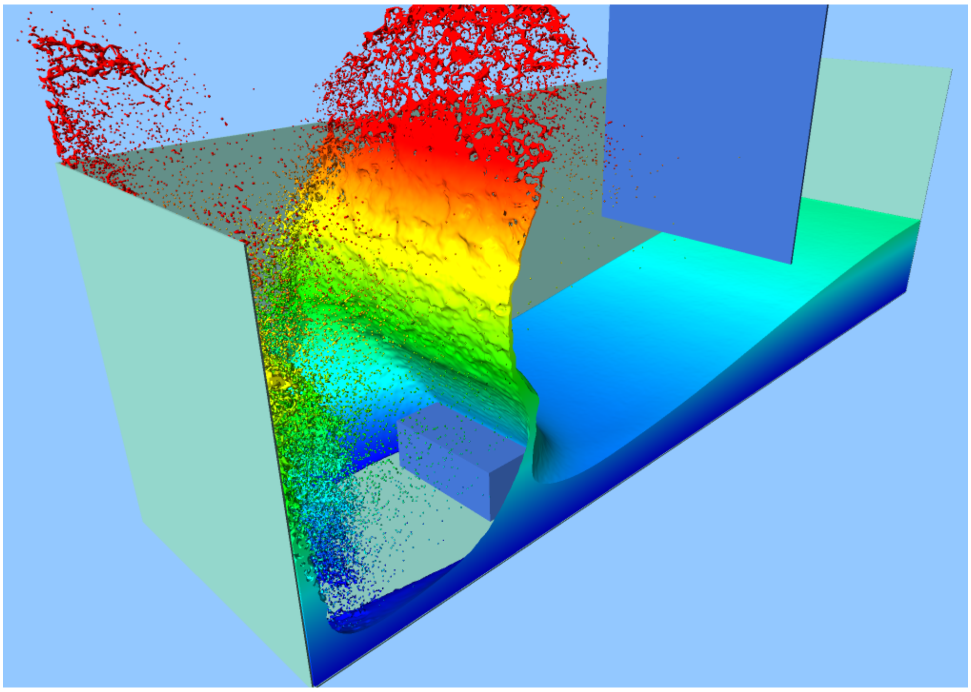
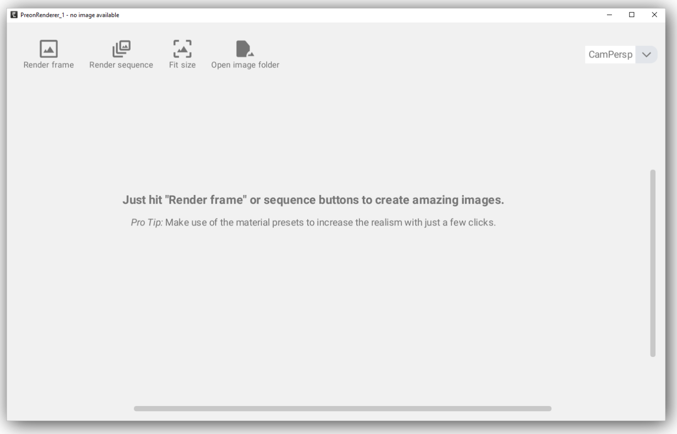


Figure 53: Rendering for a breaking dam scenario colored by fluid height.



Just hit "Render frame" or sequence buttons to create amazing images.

Pro Tip: Make use of the material presets to increase the realism with just a few clicks.

Figure 54: Rendering dialog.

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 a graphics 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	Unit/Type	What it does
individual frame rate	On/Off	If enabled, a view **frame rate** can be set which is different to the **view frame rate** provided in the **Scene** object. By this, images can be generated with a different frequency than the updates of other post-processing objects (e.g., sensors).
frame rate	-	The individual view frame rate of this object. Only visible, if **individual frame rate** is enabled.

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

Property	What it does
transparent background	If enabled, the background will be transparent. Do not use this if you intend to generate a video from the rendered images later.
background color	Specifies the background color.
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.
background dithering	Specifies the amount of dithering applied to the background. Dithering prevents color banding. The value should be between 0 and 5.
max. solid recursion depth	Sets the maximum recursion depth for solid ray tracing. You may need to increase this in scenes with many transparent layers.

max. reflection bounces	Sets the maximum number of times a ray may be reflected from a solid surface. Only relevant for reflective materials.
max. fluid recursion depth	Sets the maximum recursion depth for fluid ray tracing.
artificial secondary hemisphere	If enabled, an artificial hemisphere will be used for secondary reflections. This will improve the visual quality for scenes with monotone or dark backgrounds.
show color legends	If set to true, color legends for fluids and sensors are included in the rendered images.
show time	If set to true, the elapsed time will be drawn in the rendered images.
samples per fragment	Sets the number of stochastic samples per pixel fragment for materials with enabled **monte carlo light transport**. More samples reduce noise but also require more time to compute.

Table 115: Properties in group **Rendering**.

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

Table 116: Properties in group **Fluid rendering**.

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

Table 117: 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
max number of photons	Specifies the maximum number of photons. If this is exceeded, the photon spacing will be decreased automatically.
photon spacing	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.
photon mapping	Enables or disables photon mapping.

Table 118: 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 paramaters

The following parameters exist for all materials:

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

Table 119: Properties shared by all materials 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 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 **color** 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 **glass reflection** 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.
enable diffuse reflections	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.
monte carlo light transport	Enables or disables monte carlo light transport. This is necessary to render rough reflections or soft shadows.
subsurface scattering weight	Sets the weight of the subsurface scattering approximation as a value between 0 an 1. Please note that this currently only works for fluids and snow, but not for solids.
subsurface scattering radius	Sets the spatial radius used for approximating subsurface scattering. Please note that this currently only works for fluids and snow, but not for solids.
light sources factor	Sets a multiplier that is applied to lighting received by light source objects in the scene.
skylight factor	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.
headlight factor	Sets a multiplier for the headlight that illuminates the material from the camera's point of view.
render single particles	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 (*particles_fast* or *particles_quality*) that also adapt lighting parameters in order to achieve good visual quality.

Table 120: Properties of **Surface material** in group **Material properties**.

15.5.3 Textured surface material

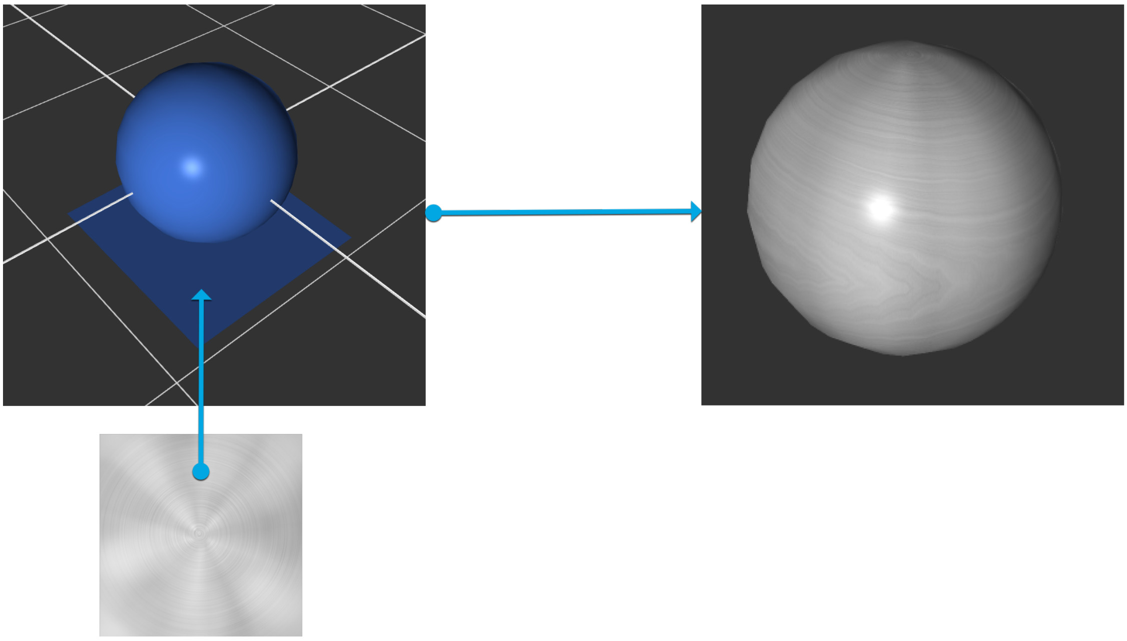


Figure 55: Texture is projected onto sphere using a plane as *UVGenerator*.

A **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 55.

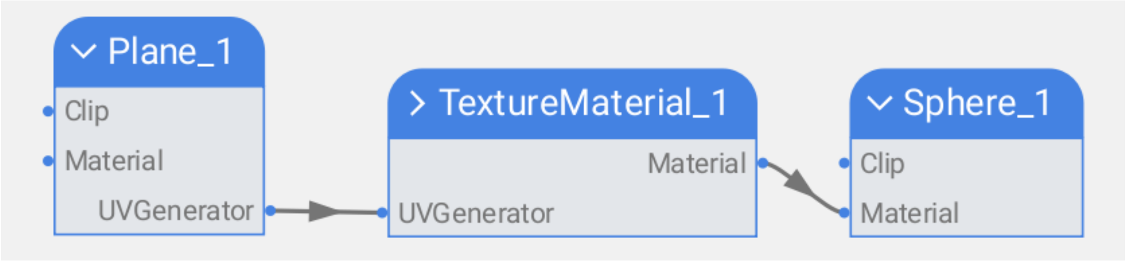


Figure 56: Connections needed for applying a texture onto a sphere using a Textured surface material.

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 to which the texture is meant to be applied to as material, as depicted in Figure 56. 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 121.

Property	What it does
texture file	Path to the PNG texture image file.

Table 121: Additional properties of **Textured surface material** in group **Material properties**.

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
absorption coefficient	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.
specular exponent	Sets the specular exponent which determines the shininess of the light reflection. Typically, values between 50 and 100 are used.
specular factor	Scales the specular term of the Phong illumination system.
index of refraction	Sets the index of refraction for the material.

Table 122: 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 58. Note that the spray model requires a detailed simulation to look convincing. It is not recommended for scenes with low particle counts.

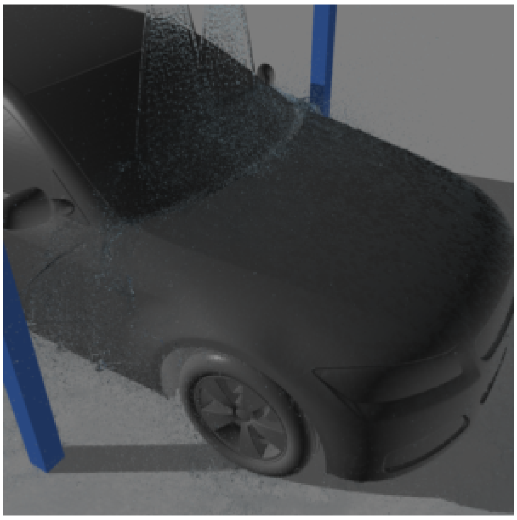


Figure 57: Fluid rendered without spray model.

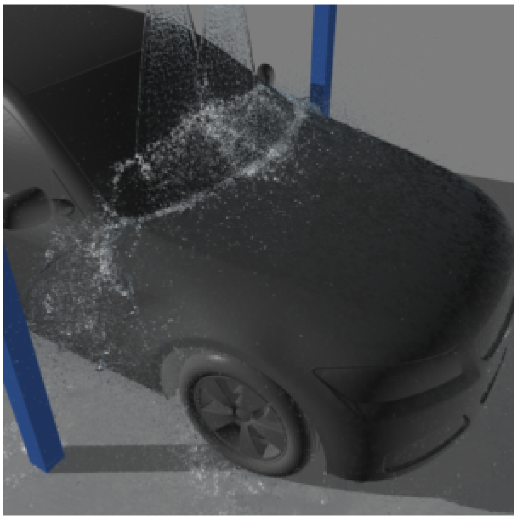


Figure 58: Fluid rendered with spray model.

Property	What it does
enable spray	Enables or disables the spray model.
spray color	Sets the spray color.
spray factor	Sets the factor weighting the amount of spray (should be between 0 and 1).
spray velocity min	Sets the minimum fluid velocity for rendering fluid as spray.
spray velocity max	Sets the velocity above which fluid is fully eligible to be rendered as spray (if mixed air is detected as well).
spray threshold	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.
ignore transp. background	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 123: Properties of **Volumetric 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

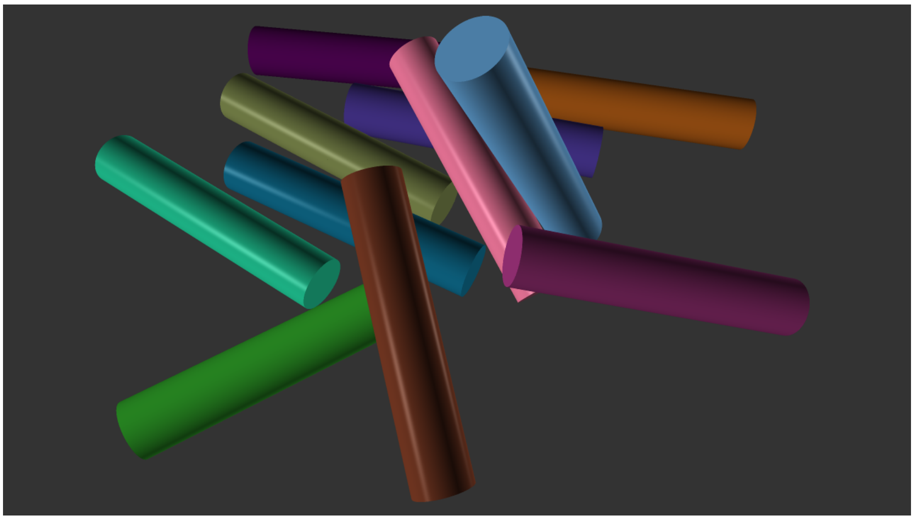


Figure 59: Cylinders on a plane rendered with the default material.

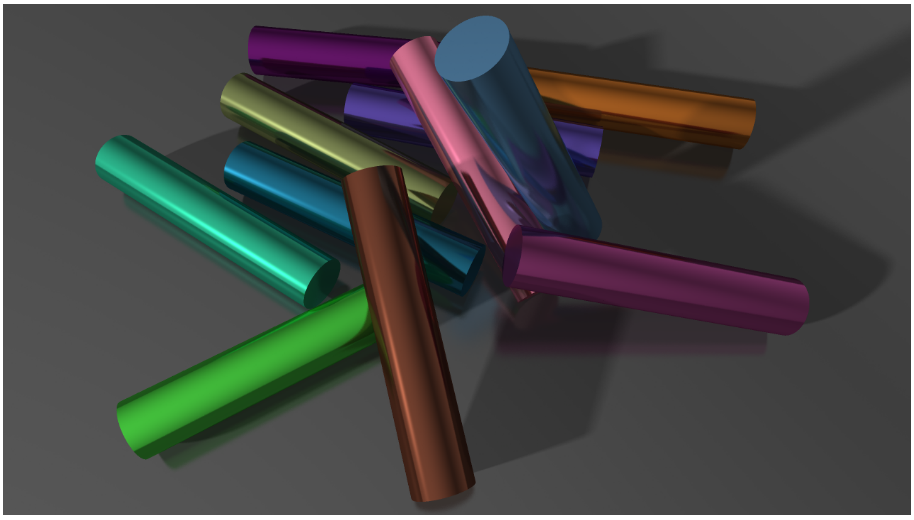


Figure 60: Here, the *metallic paint* preset was applied to the cylinders. The plane uses the *default preset (quality)*. Furthermore, shadows were enabled in the light source.

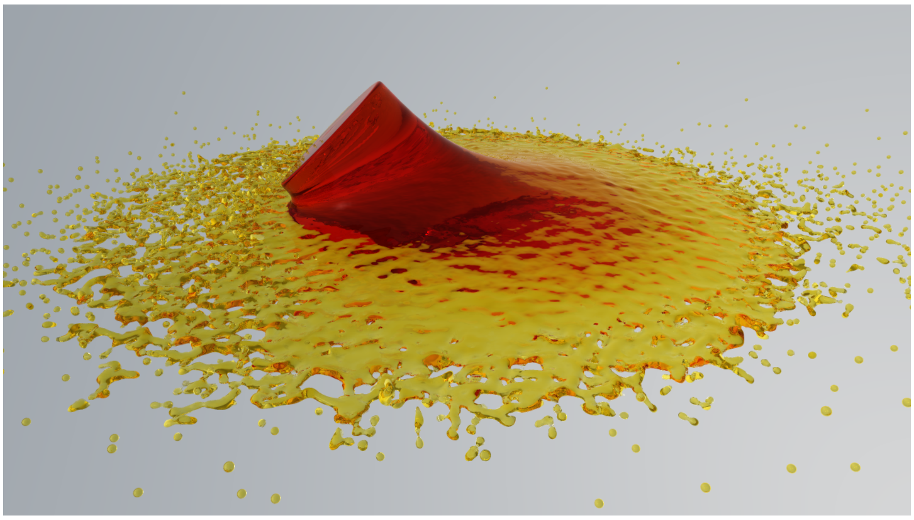


Figure 61: Fluid rendered with Volumetric material and the *motor oil* preset.

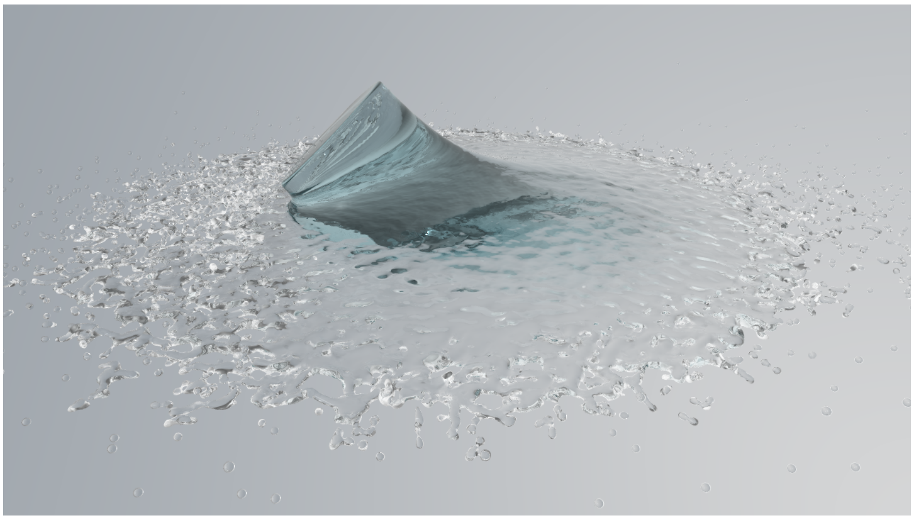


Figure 62: Fluid rendered with Volumetric material and the *water* preset.

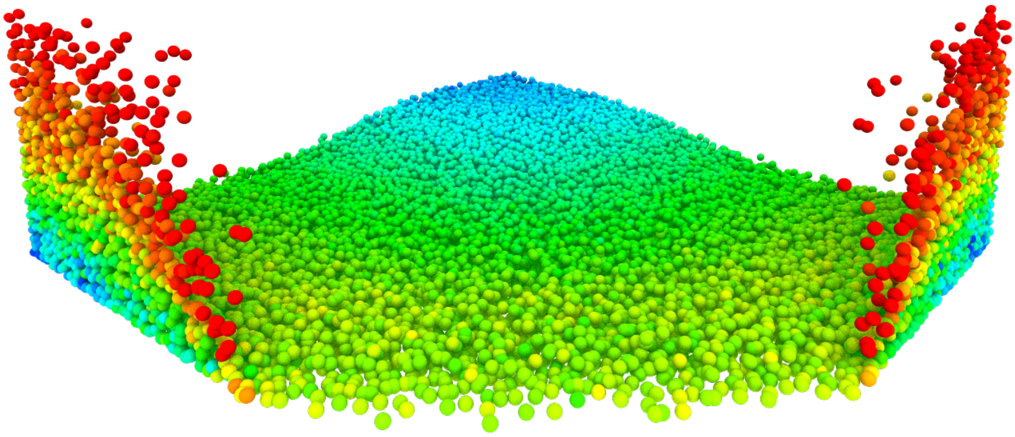


Figure 63: Fluid rendered with the Surface material and the *particles_quality* preset.

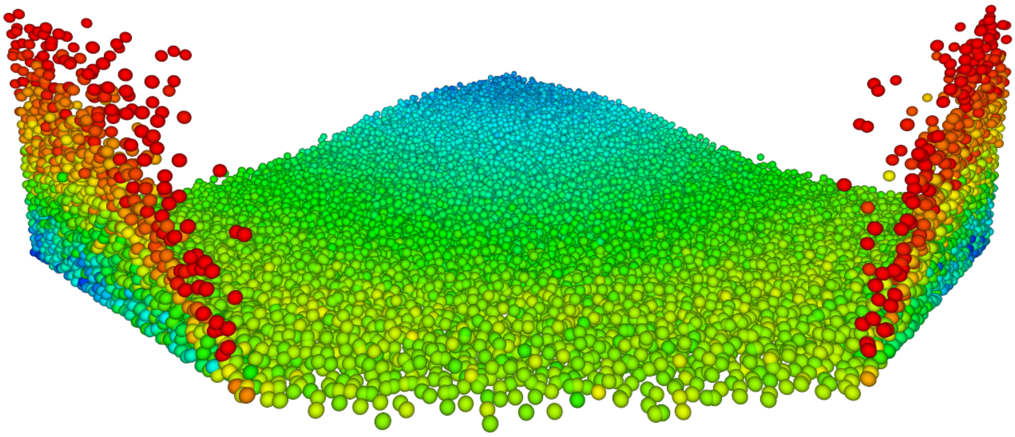


Figure 64: Fluid rendered with the Surface material and the *particles_fast* preset.

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*.

For every sensor, you can enable an individual frame rate that allows you to specify an update frequency different to the **view frame rate** provided in the **Scene** object. Therefore, set the property **individual frame rate** to **On** and then specify the **frame rate**. Both properties can be found in the property group **General** of the respective **Sensor** object.

Best practice: In some cases, it might be beneficial to set the **update sensors at substeps** to **Off** in the **Scene** object and enable individual frame rates for a better control of the update frequency and without having to increase the simulation frame rate or view frame rate on a universal level in order to guarantee sufficient sampling rates for specific sensors.

16.1 Mesh-based sensors

The mesh-based sensors (i.e., force sensor, thermal sensor, height sensor, wetting sensor and sensor mesh) need to be connected to a solid object which provides the mesh for the sensor. More precisely, these sensors 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.

If the solid object already exists in the scene, then the easiest method to create a mesh-based sensor is to right-click on the solid, which opens the context menu which contains a menu item *Connect sensor*. The menu options *New Force sensor*, *New Height sensor*, *New Sensor mesh*, *New Thermal sensor* and *New Wetting sensor* add the respective sensor to the scene and connect it to the solid object. If there are already such sensors connected to the solid, there are some exclusions (especially sensor mesh and the other types of mesh-based sensors). The excluded options are grayed out. If there exist mesh-based sensors in the scene, the context menu of a solid lists them after clicking on *Connect sensor* below the standard options. Clicking on the listed sensor connects the slot *TriangleMesh* of the solid and the sensor.

Note that you can connect a force sensor, a thermal 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^3. 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, **Box** or **Cylinder**, in which the volume is measured.
sub particle precision	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 124: 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 59 for more information.
manual border size	See Table 59 for more information.
border size	See Table 59 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 125: Volume sensor properties in group **Volume settings**.

16.5 Wetting sensor

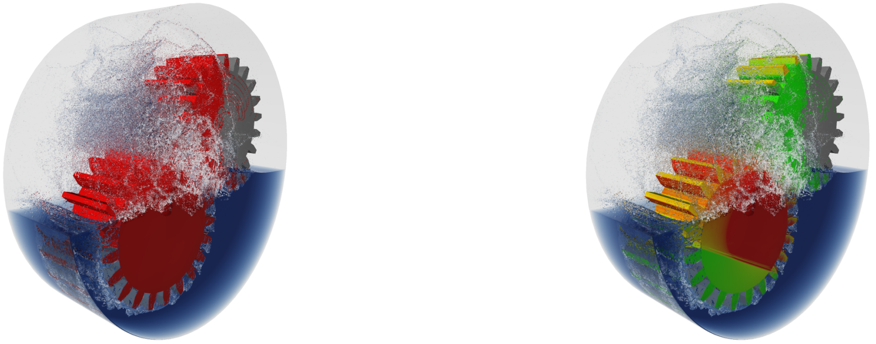


Figure 65: A gear setup after a simulation time of $0.2s$. The gears start rotating after $0.1s$. The wetting sensor in the left picture visualizes accumulated wetting whereas the wetting sensor in the right picture visualizes the wetting time. For the latter, the color legend is as follows: Green: $> 0.0s$, Yellow: $0.1s$, Red: $\geq 0.2s$.

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
show wetting	Specifies which type of wetting should be visualized. **AccumulatedWetting** visualizes all areas that were touched by fluid at some previous point in time while **CurrentWetting** only marks areas that are currently in contact with fluid. **WettingTime** visualizes the duration for which the sensor was touched by the fluid.
Coloring (binary)	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.
Coloring	Provides options to specify minimum, middle and maximum wetting time and their respective coloring (see Section 16.2 for more details) if **show wetting** is set to **WettingTime**.

Table 126: Properties of wetting sensor in group **Appearance**.

Property	What it does
track wetting time	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 **show wetting** is switched to **WettingTime**. If you are not interested in the wetting time, disable it again to improve the performance and to reduce disk usage.

Table 127: Properties of wetting sensor in group **General**.

The wetting statistics are printed in percentage in the graphics window. Please note that the *Saturated Wettage* 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 file”. 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.

16.5.1 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 pressure 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.
track max shear stress	If enabled, each sensor sample will store the maximum shear stress 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 128: Properties in group **General** of **Force sensor**.

Property	What it does
coloring mode	When choosing **current pressure** or **current shear stress**, the sensor will be colored according to currently measured corresponding force components. When choosing **cum. max. pressure** or **cum. max. shear stress**, the sensor will colorize each sensor sample according to the corresponding maximum force components (per m^2) that acted on this sample at some point in time. Accumulated data **cum. max pressure** and **cum. max shear stress** will only be available if **track max pressure** and **track max shear stress** respectively was enabled during the last sensor update run.

Table 129: 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 9.1.16 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 file*. 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.

Center of rotation for torque measurements

Torque is measured with respect to a center of rotation. By default, the pivot position of the connected solid object will be used as the center. One way to change this center is to adapt the pivot of the connected solid. Another possibility is to connect the outgoing *Transform* connection slot of another object to the *CenterOfRotation* slot of the force sensor. In this case, the global position of the connected object will be used as the center of rotation. This also allows to share the same center of rotation between multiple force sensors. Please note that the torque will still be measured in global space. If the torque should be measured in local space, also connect the *Transform* connection slot as described below.

Transforming forces into local space

By default, all measured force vectors are in global space. Forces can be transformed into local space by connecting the *Transform* connection slot of another object to the *Transform* connection of the force sensor.

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 Thermal sensor

Depending on the tracking mode of the thermal sensor, it measures either heat flux or temperature and heat transfer coefficient (HTC) at the surface of the solid connected to the sensor. Minimum, maximum and average values of the respective quantities are gathered. The units are W/m^2 for the heat flux, $W/(m^2\,K)$ for the heat transfer coefficient and K or °C for the temperature.

The tracking mode is derived from the thermal properties of the connected solid. If **Thermodynamics**→**boundary type** is set to **temperature** in the solid, the sensor measures the heat flux and the HTC. Set it to **heat flux** and the sensor provides the solid surface temperature and the HTC. If no thermal boundary condition is defined on the solid surface (i.e., **Thermodynamics**→**boundary type** is set to **none**), no heat transfer is observed unless you enable the tracking mode for conjugate heat transfer. Therefore, keep the **Thermodynamics**→**boundary type** set to **none** in the connected solid and, additionally, connect a solid volume solver and a fluid solver to the thermal sensor through the *Particles* slot in the connection editor. In this case, heat flux, solid surface temperature and HTC values are gathered.

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, statistics on the measured thermal quantities ignore the surface areas where no fluid is in contact with the solid.
track time-avg→heat flux	If enabled, data for recording the average heat flux are generated and stored for any point in time over the duration of the sensor being active. Whether this property is visible depends on the tracking mode (see above).
track time-avg→ temperature	If enabled, data for recording the average temperature are generated and stored for any point in time over the duration of the sensor being active. Whether this property is visible depends on the tracking mode (see above).
track time-avg→heat transfer coefficient	If enabled, data for recording the average heat transfer coefficient are generated and stored for any point in time over the duration of the sensor being active.

Table 130: Properties in group **General** of the **Thermal sensor**.

Property	What it does
coloring mode	Specifies according to which property the connected mesh is colored. The available options depend on the tracking mode of the sensor (see above).

Table 131: Properties in group **Appearance** of the **Thermal sensor**. These properties determine what is displayed on the mesh.

The heat transfer coefficient is computed as follows:

$$HTC = \frac{q}{T_S - T_F}, \tag{4}$$

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 **Thermal sensor**.

You can export the thermal quantities on a per-sample basis by right-clicking the sensor and choosing ”Export sample data to CSV file”. 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. If the **ignore dry areas** is enabled, the heat transfer of the surface areas where no fluid is in contact with the solid, are undefined in the exported CSV file. To replace this with user-defined data, a point cloud resource can be utilized. To achieve this, import a CSV file containing the heat flux, heat transfer coefficient or temperature using a point cloud resource and select the sample type as per the heat transfer property type the thermal sensor senses. The tensor field slot name of the point cloud resource changes depending on the imported sample type to either *TemperatureField*, *HeatFluxField* or *HeatTransferCoefficientField*. In the connection editor, connect this slot of the point cloud resource to the *TensorField* slot of the thermal sensor. Now exporting the sensor data as CSV with the **ignore dry areas** on, will use the heat transfer values from the connected point cloud resource for the dry areas. The exported files are stored in the respective subfolder of the folder ’SensorData’ in the scene directory.

To track and visualize a time-averaged heat flux, temperature or HTC, the properties **track time-avg**→**heat flux**, **track time-avg**→**temperature** and **track time-avg**→**heat transfer coefficient** must be enabled, respectively. The temporal average is then calculated from the point in time when the respective properties were enabled until the current time in the simulation. Since these properties are keyframeable, the averaged values are only stored for the points in time where the respective property is enabled. This allows to track them over multiple time intervals, where the average is computed per time interval. In order to visualize the time-averaged quantity, the respective quantity has to be selected from **Appearance**→**coloring mode**. However, the time-averaged quantity can be chosen under the **coloring mode** without enabling the tracking of the respective quantity. In this case, the tracking is automatically enabled with a warning message indicating this. A similar warning message is also displayed when the tracking is switched off while the respective time-averaged quantity is being shown.

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

16.9 Y+ Sensor

The Y+ sensor measures the dimensionless wall normal distance (y^+) between the fluid particles in contact with the solid. This is the same quantity used in the calculation process of the wall functions (see Section 9.1.13 for more information). The sensor tracks the maximum, minimum and average y^+ values over the solid. See Section 16.1 for information on how to connect a solid with the sensor and the rules that apply.

Property	What it does

direction pointing	Defines the normal direction in which the y^+ is measured. Setting **OneSided** ignores fluids in the opposite normal direction of the solid surface while **OneSided_Flipped** ignores fluids in the normal direction of the solid surface. Setting **TwoSided** measures both sides and gives an average.
scaleRadius	Scales the radius of the sphere around each solid particle in which it can see fluid particles, as a multiple of the particle size. The default value is 1.

Table 132: Properties in group **General** of the **Y+ Sensor**.

You can export the measured y^+ value on a per-sample basis by right-clicking the sensor and choosing ”Export sample data to CSV file”. 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.

16.10 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 you 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*.

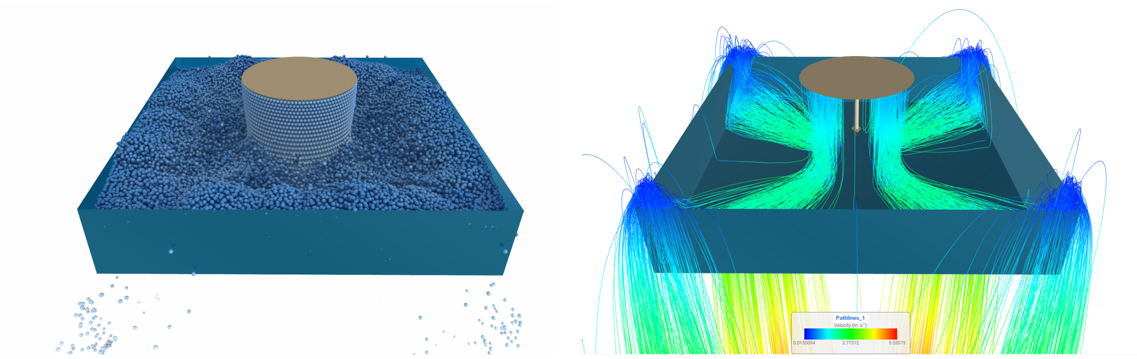


Figure 66: Pathlines for fluid particles flowing over the box.

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 67.

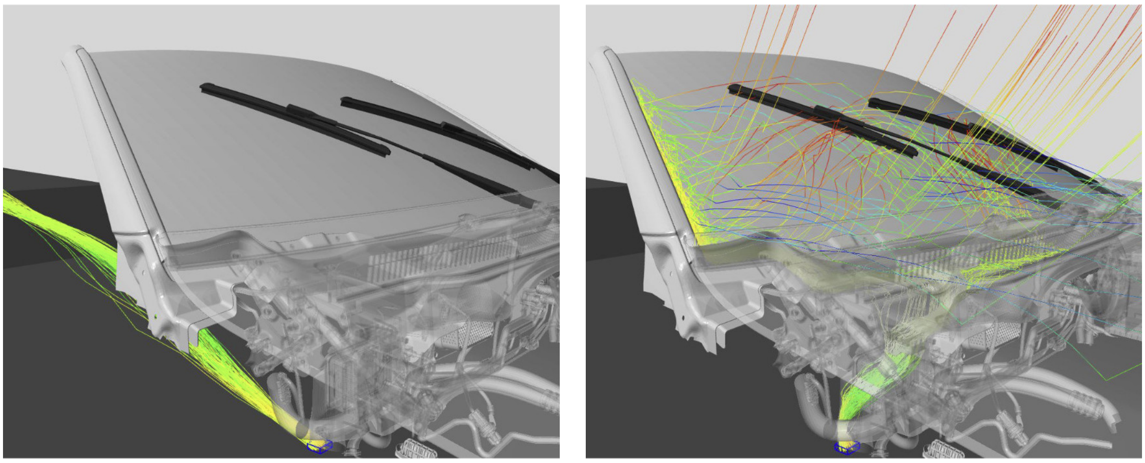


Figure 67: Global pathlines (left) versus pathlines relative to a moving object (right).

16.10.1 Parameters explained

Property	What it does
capture frame	Specifies the frame in which particles to be tracked are identified.
thinning	Specifies whether a grid should be used to reduce the number of tracked particles, resulting in less pathlines. Disabled by default.
thinning factor	Only relevant if Thinning is enabled. Specifies the ratio between ignored and tracked particles in one dimension.
vertex limit	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.
playback	Enables or disables playback of pathlines over time.
restrict lifetime	Enables or disables the ability to restrict the lifetime of pathline segments. Thus, pathlines disappear again after a specified time during the playback. Prerequisite: **playback** has to be enabled.
lifetime	Specifies for how long a pathline segment is shown during the playback of pathlines over time. Prerequisite: **restrict lifetime** has to be enabled.
Appearance→line width	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 size. You may have to adjust the line width parameter when using Preon Renderer to achieve results similar to OpenGL.
Appearance→show box	Specifies if the box visualizing the capture area should be visible.

Table 133: Properties in group **Pathlines**.

16.10.2 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.

16.11 Sensor plane

The sensor plane measures and displays different quantities of one or multiple fluids or **Solid Volume Solvers** 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. The sensor plane is not automatically connected to a **Solid Volume Solver**.

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 size 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 the simulation. This is because only selected particle states at each frame are available for the sensor plane to perform its measurements during postprocessing. If the fluid flow varies considerably in the time between two consecutive frames, there is insufficient information to correctly reconstruct the flow in order for the sensor plane to be able to measure as accurately as during the simulation.

Property	What it does
property	Sensor planes always measure all possible properties, e.g., **FlowRate**, **VelocityMagnitude**, **Density**, **Pressure** or **Temperature**. The user can set via **property** which value should be displayed on the plane. The sensor plane is connected to the fluid solver by default.

cell size	The cell size defines the resolution of the sensor grid. Higher resolutions give finer results, but take more time to compute. By default, the particle size of the fluid solver is used, larger values are not recommended and can not be set.
automate cell size	Synchronizes the cell size with the smallest particle size of the connected solver at all times.
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 *[sceneFolder]/SensorData/[SensorName]/[PropertyName].png*.
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 *image* allows to mask out active cells using a bitmap image. Setting it to **Triangle mesh** allows you to select active cells by projecting a triangle mesh on the sensor plane. To specify the mesh, connect the *TriangleMesh* 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. **TwoSided_Diff** will sum up signed flow rates leading to a positive and negative arrow. A **TwoSided_Sum** 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 134: 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 **Alpha** value to zero.
display arrows	Toggles the **flow rate mode** arrows **on** or **off**. The property group **Arrow colors** will be hidden if arrows are toggled **off**.
Arrow colors	Within this property group, you can define **positive color** and **negative color** for the **flow rate mode** indicator arrows.

Table 135: Properties in group **Appearance**.

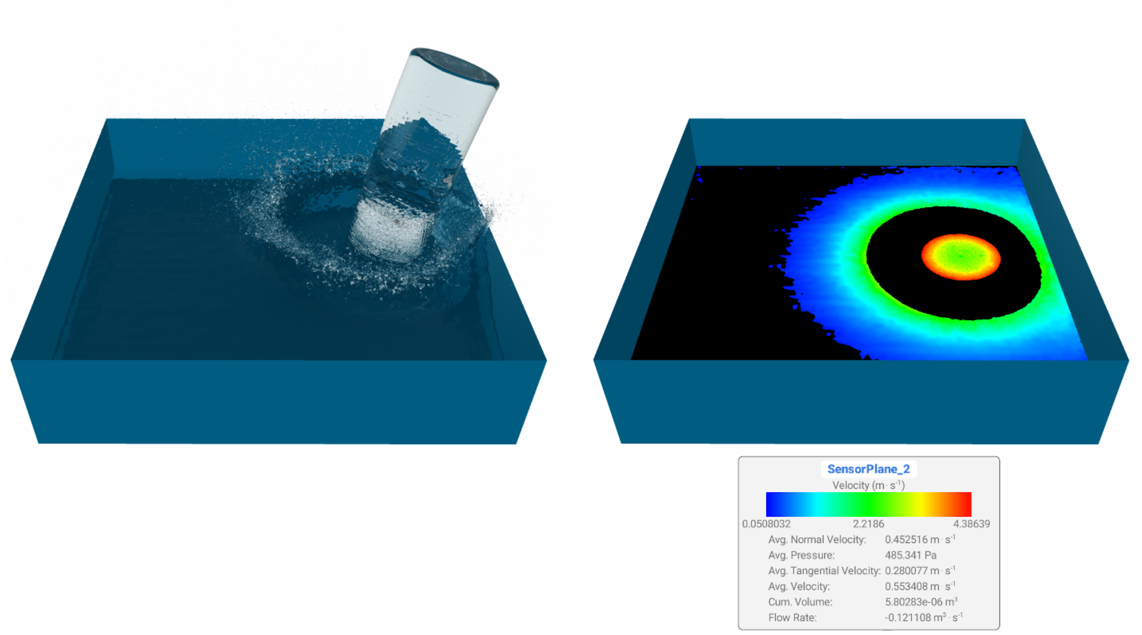

SensorPlane_2
Velocity (m · s^{-1})
0.0508032
2.2186
4.38639
Avg. Normal Velocity: 0.452516 m · s^{-1}
Avg. Pressure: 485.341 Pa
Avg. Tangential Velocity: 0.280077 m · s^{-1}
Avg. Velocity: 0.553408 m · s^{-1}
Cum. Volume: 5.80283e-06 m^3
Flow Rate: -0.121108 m^3 · s^{-1}

Figure 68: Sensor plane showing velocity magnitude for fluid flowing into a box.

16.11.1 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 file*. 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 134. 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.

16.12 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 69 shows an example of using this feature in PreonLab. See Section 16.1 on how

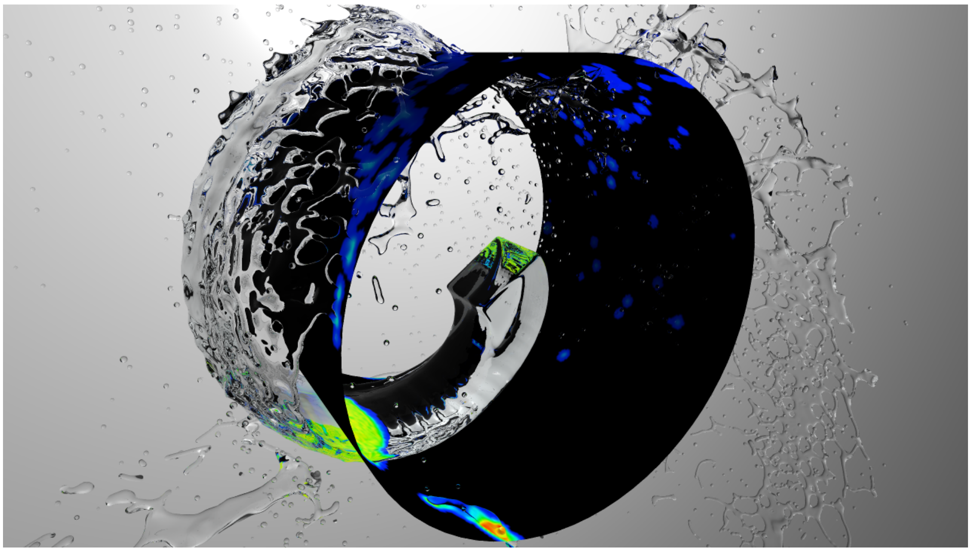


Figure 69: Cylindrical sensor mesh measuring flow rate. Image rendered with Preon renderer.

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 134.

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

16.13 Projection fields

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

16.13.1 Velocity projection field

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

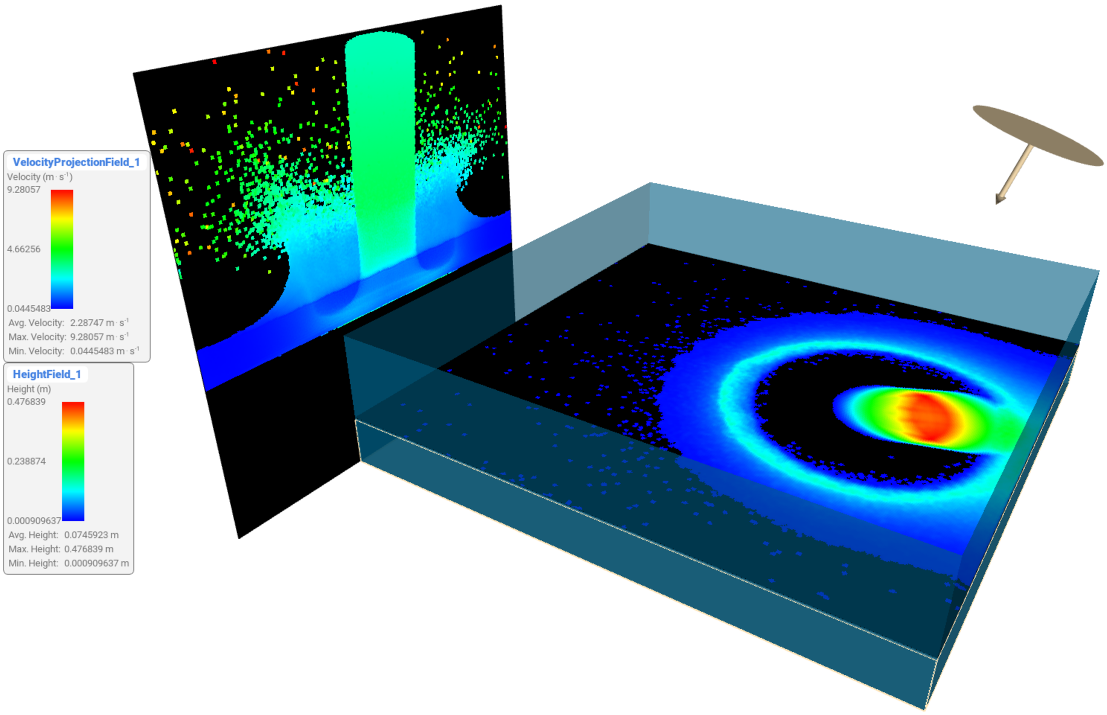

VelocityProjectionField_1
Velocity (m · s⁻¹)
9.28057
4.66256
0.0445483
Avg. Velocity: 2.28747 m · s⁻¹
Max. Velocity: 9.28057 m · s⁻¹
Min. Velocity: 0.0445483 m · s⁻¹
HeightField_1
Height (m)
0.476839
0.238874
0.000909637
Avg. Height: 0.0745923 m
Max. Height: 0.476839 m
Min. Height: 0.000909637 m

Figure 70: Invisible fluid is pouring from a circle source into a box. A height field is placed at the bottom of the box. Its **type** is set to **VolumeHeight**. A velocity projection field is placed to the left of the box.

16.13.2 Height field

The height field visualizes fluid height on a plane.

16.14 Height Sensor

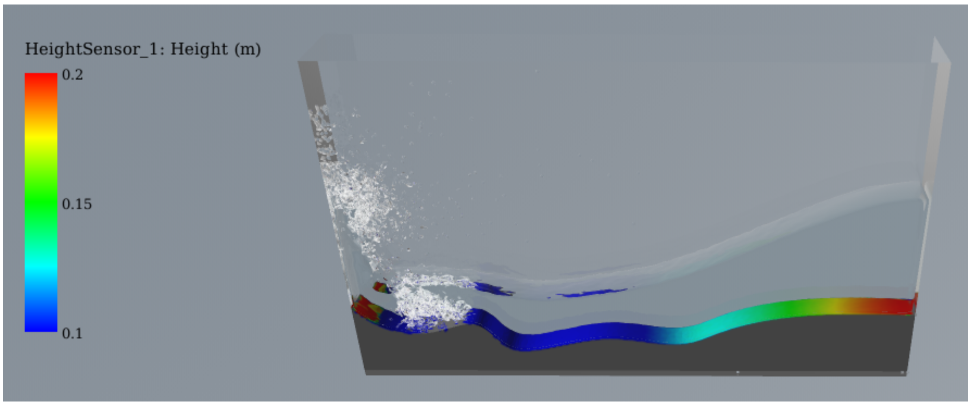


Figure 71: Height sensor used with arbitrarily shaped solid mesh.

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 71 and Figure 72 demonstrate how the height sensor can be used with Preon Solver and Snow Solver,

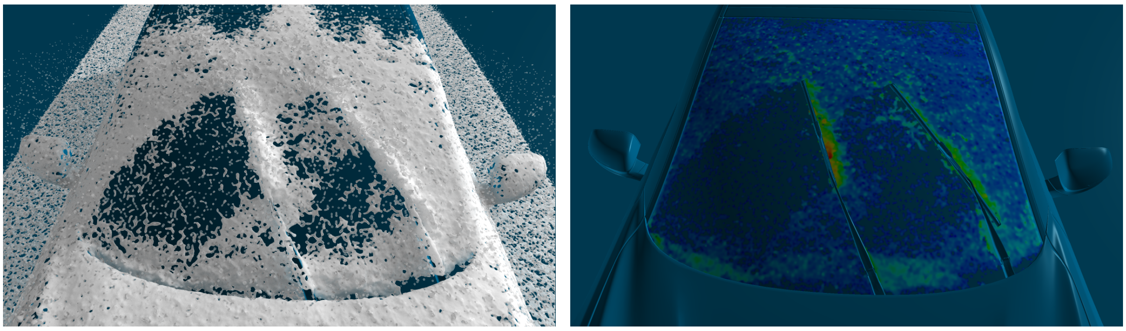


Figure 72: A car has been snowed in and the wipers just got active clearing the front window. A height sensor is connected to the front window and measures the heights on top of the front window.

respectively. Figure 73 illustrates how splashed particles are not considered by the 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. Note that the height sensor does not generate data when connected to an inactive mesh. If you want to use it anyway, you can activate the mesh after the simulation and use the post-processing to get the desired data.

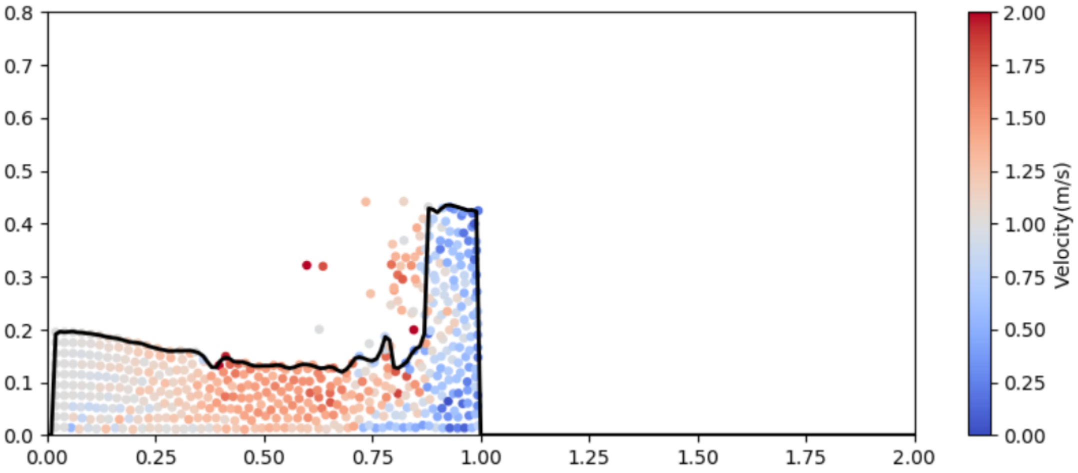


Figure 73: Fluid is released from a dam on left side and hits a wall at 1m on the right side. Results are obtained in PreonLab and visualized in matploblib with a PreonPy buffer set. The black line represents the heights obtained by the height sensor. Only fluid in direct contact with the surface is measured. Splashed particles are ignored.

Property	What it does
sampling rate	Specifies the rate with which the fluid surface is sampled as a factor of the particle size of the fluid. A lower sampling rate trades accuracy for performance.
height threshold	Factor which multiplied with the particle size of the connected fluid computes to the tolerated distance between fluid and sensor that still counts the fluid as on top. You may need to tweak this if **direction pointing** is set to **Custom** and measuring near parallel to the surface.

min culling neighbor count	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.
direction pointing	**OneSided** measures height in normal direction and **OneSided_Flipped** in inverted normal direction. **TwoSided** measures both sides and outputs the maximum. **Custom** measures in a user-defined direction.
direction	Only shown for **direction pointing** set to **Custom**. Sets the direction in which to measure.
local transform	Only shown for **direction pointing** set to **Custom**. When enabled, **direction** is in the local coordinate system of the mesh and will rotate with the object. Set to **off** the **direction** is global and fixed.

Table 136: 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 and normal used in computation on a per-sample basis by right-clicking the sensor and choosing *Export sample data to CSV file*. 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.

16.15 Vector field visualizer

You can visualize an acceleration field or an air flow (see Section 12.3) using a **vector field visualizer** which can be added to your scene via *Add→Sensor→Vector field visualizer*. The vector field visualizer works like a sensor plane for the connected field. When hovering the mouse on the visualizer the data at the cursor is displayed on the OSD (see Figure 74).

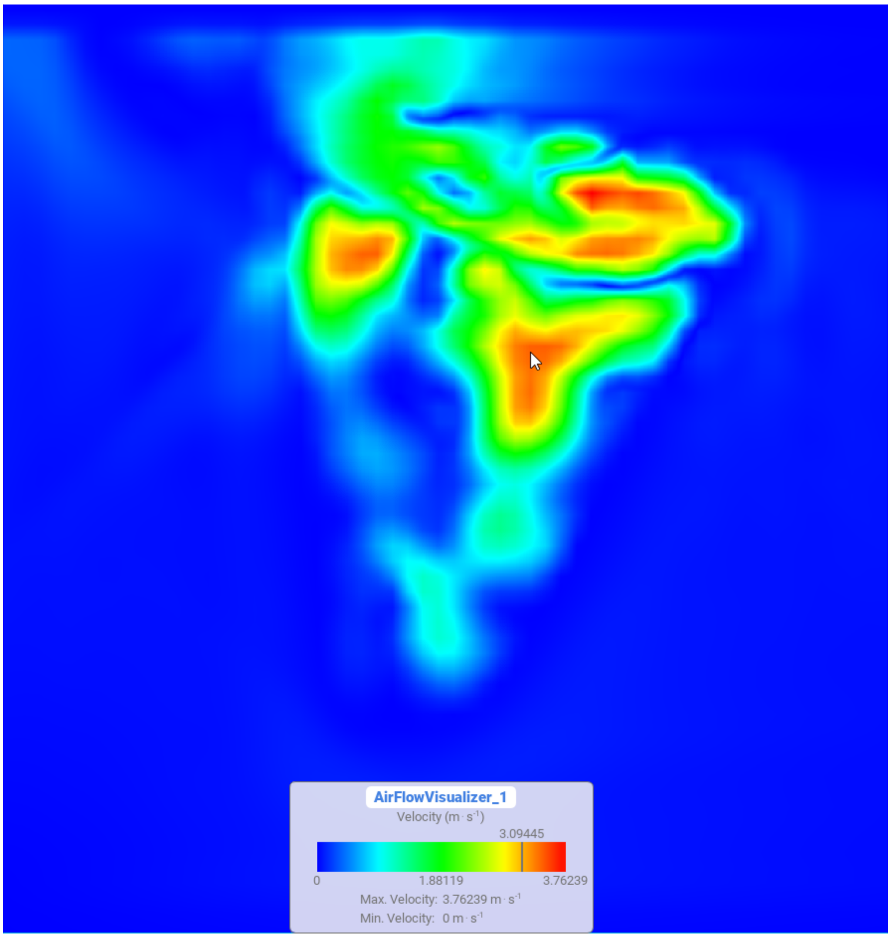

AirFlowVisualizer_1
Velocity (m · s^{-1})
3.09445
0
1.88119
3.76239
Max. Velocity: 3.76239 m · s^{-1}
Min. Velocity: 0 m · s^{-1}

Figure 74: The vector field visualizer displays velocity magnitudes of an air flow object on a two-dimensional plane.

Property	What it does
cell size	Sets the cell size of the vector field visualizer. When using the grid-based storage for the connected field, this will be automatically set to one tenth of the cell size of this field (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 *[sceneFolder]/SensorData/[SensorName]/[PropertyName].png*

Table 137: Properties in group **Settings**.

16.16 Deleted particles visualizer

As mentioned in (see Section 9.1.8) particles may be deleted according to the solver's deletion criteria. There might be cases where it would be useful to know where and how many particles are deleted due to the different deletion criteria. Knowing this, one could improve the scene set up to reduce the amount of particles deleted and eventually improve the quality of the simulation.
To do so, you can use a **deleted particles visualizer**. It renders the deleted particles, colored according to the criterion because of which they were deleted. The number of deletions for each criterion can be found in the OSD and its evolution can be seen in the plot dialog. Note that different particles may be deleted at exactly the same position.
When added, this sensor connects automatically to all the solvers present in the scene, thus showing the deleted particles of each of them. Adding a solver to a scene containing a **deleted particles visualizer** will also make an automatic connection. To exclude the deleted particles of a certain solver, use the connection editor to delete the outgoing connection from the *DeletedParticles* slot of that solver.

You can choose the color to map each deletion criterion from **Appearance** subgroup **Criteria**

Property	What it does
scale	Allows to scale the visualized deleted particles to locate them easily.
restrict lifetime	Enables or disables the ability to restrict the duration of the deleted particles being shown in the visualizer after being deleted. Thus, they disappear again after a specified time during the simulation/ playback.
lifetime	Specifies for how long (in seconds) a deleted particle is shown. Prerequisite: **restrict lifetime** has to be enabled.

Table 138: Properties in group **General**.

16.17 Python particle access

In some situations, 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 their attached values, e.g. 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 75). 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).

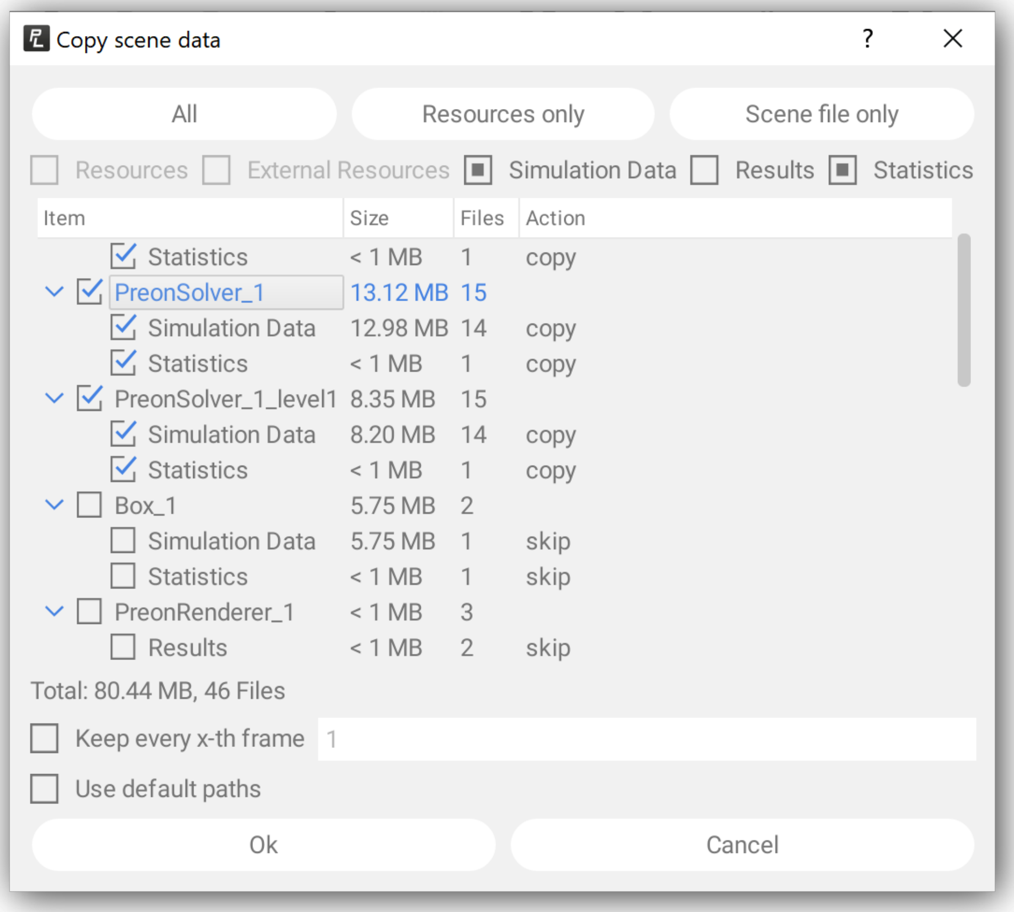
External Resources: The same type of data as *Resources*, but located outside of the scene directory. Selecting these entries will copy the respective data to the scene directory. Deselecting these entries will keep the scene intact, because the path to external resources will be still valid after saving. 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



☐ Resources ☐ External Resources ▣ Simulation Data ☐ Results ▣ Statistics

Item	Size	Files	Action
☑ Statistics	< 1 MB	1	copy
☑ PreonSolver_1	13.12 MB	15	
☑ Simulation Data	12.98 MB	14	copy
☑ Statistics	< 1 MB	1	copy
☑ PreonSolver_1_level1	8.35 MB	15	
☑ Simulation Data	8.20 MB	14	copy
☑ Statistics	< 1 MB	1	copy
☐ Box_1	5.75 MB	2	
☐ Simulation Data	5.75 MB	1	skip
☐ Statistics	< 1 MB	1	skip
☐ PreonRenderer_1	< 1 MB	3	
☐ Results	< 1 MB	2	skip

Total: 80.44 MB, 46 Files

☐ Keep every x-th frame 1

☐ Use default paths

Figure 75: 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 a 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 frames 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 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.

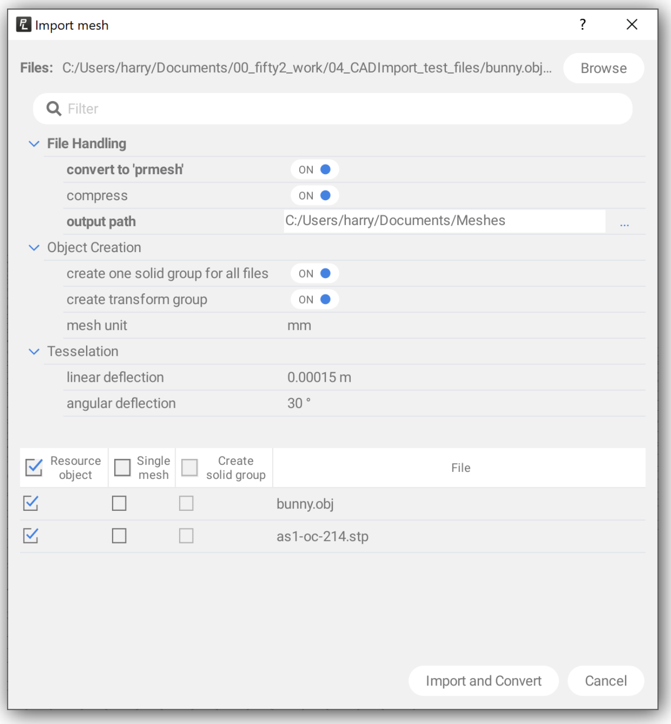
17.2 Import meshes

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 resources, which allows to create separate mesh objects for each sub-mesh contained in the file. This can be selected for all files or for each file individually.

Exposing all of the sub-meshes for a mesh file can lead to many distinct solids in the scene. Thus, on import you can choose to group solids. By default, *Create one solid group for all files* groups all sub-meshes into one big group. You can also choose to *Create solid group* for a single file to group its sub-meshes into one group.

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. Keep in mind that transform groups are distinct from object groups.

You can also specify the unit in which the mesh was modeled in for all files. This value



Import mesh

Files: C:/Users/harry/Documents/00_fifty2_work/04_CADImport_test_files/bunny.obj...

File Handling	
convert to 'prmesh'	ON ●
compress	ON ●
output path	C:/Users/harry/Documents/Meshes
Object Creation	
create one solid group for all files	ON ●
create transform group	ON ●
mesh unit	mm
Tesselation	
linear deflection	0.00015 m
angular deflection	30 °

☑ Resource object	□ Single mesh	□ Create solid group	File
☑	□	□	bunny.obj
☑	□	□	as1-oc-214.stp

Figure 76: Mesh import dialog.

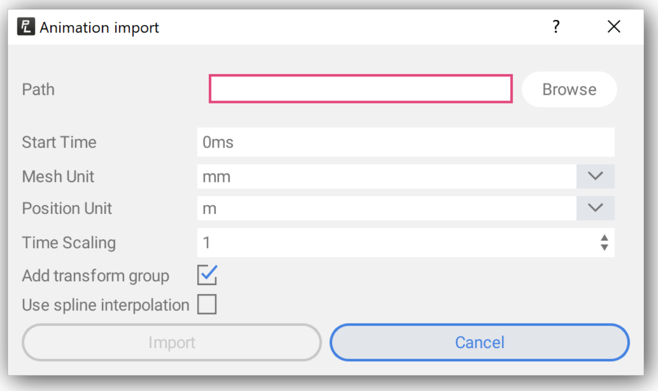
is then set in the created resource objects or as scale factor in the mesh objects, if no resource object should be created.

For STEP files, PreonLab needs to tessellate the geometry. To guide the tessellation, you get to choose the tessellation settings under the **Tessellation** group of the dialog. Please refer to the Table 103 for detailed description of the tessellation settings we provide. Please note that we recommend using a mesh resource object when importing STEP files, as this will enable you to verify the tessellation settings used after the import, and give you the chance to re-tessellate an imported file, if required.

The mesh import dialog gives you the option to convert meshes into the PRMESH file format (which is an optionally compressible binary file format for storing mesh data). During the import, please refer to Section 13.6 for more details about the format and the benefits it provides you. The settings to enable conversion can be found under the property group **File Handling**. The option *convert to 'prmesh'* will convert all the meshes to be imported into the PRMESH file format. Make sure you have a mesh resource object for each of the meshes, as this is a requirement for the conversion to PRMESH. Additionally, you have to provide an *output path* where the converted files will be placed.

17.3 Import animation data

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



Path

Start Time 0ms

Mesh Unit mm

Position Unit m

Time Scaling 1

Add transform group ☑

Use spline interpolation ☐

Figure 77: Animation import dialog.

File→*Import*→*Import Animation* which opens a dialog as shown in Figure 77. 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 6.1.4).

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 139: 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 13.7.

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.

17.6 Import Tensor Field

Importing air flow, acceleration, velocity fields as well as all fields concerning heat transfer, such as heat flux, heat transfer coefficient and temperature, i.e., vector fields and scalar fields, can be done through the unified **Import Tensor Field** dialog found in *File→Import→Import Tensor Field*. See Section 12.3 for information on how to import airflow and acceleration fields.

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 predefined temperatures onto 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 field type can be chosen under **Samples→quantity type** under the *Property Editor* and then right-click on the point cloud resource on the scene inspector to find the respective import option. In the connection editor of the **Point cloud resource**, the slot for the tensor field is renamed as per the sample quantity type selected in the *Property Editor*

17.6.1 Importing temperature or heat flux samples from CSV

To import a CSV file with temperatures or heat flux, choose **Temperature** or **Heat Flux** in the **Tensor** drop-down menu of the *Tensor Field Import* dialog. The necessary inputs are described in Table 140 and Table 141.

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 points' x-positions.
Y-Position Name	Choose the respective column name from the drop-down list to specify the sample points' y-positions.
Z-Position Name	Choose the respective column name from the drop-down list to specify the sample points' z-positions.
Temperature Name	Choose the respective column name from the drop-down list to specify the sample points' temperatures.
Distance Unit	Provides the unit in which the sample point positions are specified.
Temperature Unit	Provides the unit in which the temperatures are specified.
Temperature Offset	Adds an offset to all the imported temperatures.

Table 140: Import dialog input fields for importing temperatures.

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 points' x-positions.
Y-Position Name	Choose the respective column name from the drop-down list to specify the sample points' y-positions.
Z-Position Name	Choose the respective column name from the drop-down list to specify the sample points' z-positions.
Heat flux Name	Choose the respective column name from the drop-down list to specify the sample points' heat flux.
Distance Unit	Provides the unit in which the sample point positions are specified.
Heat Flux Unit	Provides the unit in which the heat flux values are specified.
Heat flux Offset	Adds an offset to all the imported heat flux.

Table 141: Import dialog input fields for importing heat flux.

After the **Point cloud resource** has been added to the scene, it can be used to map the stored temperatures or heat flux on solid objects. To do so, create a connection between the resource and the receivers of the mapping i.e. the solid objects. For a temperature field mapping, use the *TemperatureField* slot of the resource and for a heat flux mapping, use the *HeatFluxField* slot of the resource to connect to the *TensorField* slot of the solid object. Finally, the mapping has to be triggered via right-clicking on the solid object in the scene inspector and selecting *Apply connected tensor field*. Note that this action becomes available only after:

a) a point cloud resource is connected to the solid object
b) thermodynamics are enabled for the solid object
c) the scene contains at least one solver (to determine the particle size of the surface sampling of the solid)

It is also possible to apply temperature, heat flux and heat transfer coefficient map on a thermal sensor to replace the undefined heat transfer values of dry areas. Please refer to Section 16.8 for details.

Best practices

In the following, we recall important steps to consider in order to 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 **boundary type** 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 size 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.
- If 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.

17.6.2 Importing power and volumetric power samples from CSV

To import a CSV file with power or volumetric power, choose **Power** or **VolumetricPower** in the **Tensor** drop-down menu of the *Tensor Field Import* dialog. The necessary inputs are described in Table 142 and Table 143.

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 points' x-positions.
Y-Position Name	Choose the respective column name from the drop-down list to specify the sample points' y-positions.
Z-Position Name	Choose the respective column name from the drop-down list to specify the sample points' z-positions.
Power Name	Choose the respective column name from the drop-down list to specify the sample points' power.
Distance Unit	Provides the unit in which the sample point positions are specified.
Power Unit	Specifies the unit of the power as defined for the sample points.
Power Offset	Adds an offset to the imported power values.

Table 142: Import dialog input fields for importing power values.

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 points' x-positions.
Y-Position Name	Choose the respective column name from the drop-down list to specify the sample points' y-positions.
Z-Position Name	Choose the respective column name from the drop-down list to specify the sample points' z-positions.
Volumetric Power Name	Choose the respective column name from the drop-down list to specify the sample points' volumetric power.
Distance Unit	Provides the unit in which the sample point positions are specified.
Volumetric Power Unit	Specifies the unit of the volumetric power as defined for the sample points.
Volumetric Power Offset	Adds an offset to the imported volumetric power values.

Table 143: Import dialog input fields for importing volumetric power values.

The imported power field or volumetric power field can be used to control the spatial distribution of the heat field, see Section 12.6 for more information.

17.6.3 Importing velocity samples from CSV

To import a CSV file with velocities, choose **Velocity** in the **Tensor** drop-down menu of the *Tensor Field Import* dialog. The necessary inputs are described in Table 144.

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 points' x-positions.
Y-Position Name	Choose the respective column name from the drop-down list to specify the sample points' y-positions.
Z-Position Name	Choose the respective column name from the drop-down list to specify the sample points' z-positions.
X-Velocity Name	Choose the respective column name from the drop-down list to specify the sample points' x-velocities.
Y-Velocity Name	Choose the respective column name from the drop-down list to specify the sample points' y-velocities.
Z-Velocity Name	Choose the respective column name from the drop-down list to specify the sample points' z-velocities.
Distance Unit	Provides the unit in which the sample point positions are specified.
Velocity Unit	Specifies the unit of the velocity as defined for the sample points.
Velocity Offset	Adds a per-component (xyz-direction) offset to all the imported velocities.

Table 144: Import dialog input fields for importing velocities.

The imported velocity field can be used to define the initial velocities of particles emitted by a **Volume Source**, see Section 10.2.4 for more information.

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 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.

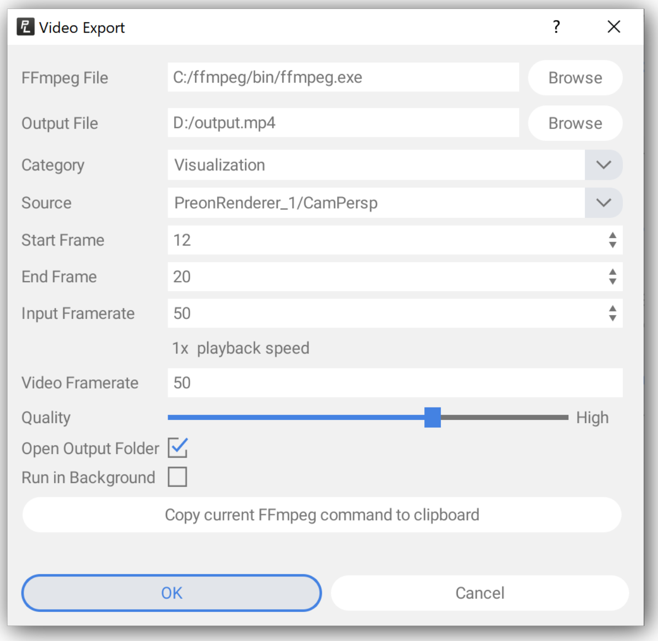
17.9 Export to EnSight

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 9.1.16). Note that changing the caching settings after the simulation won't change the simulation data which is already stored on disk.

17.10 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*[1] which has to be downloaded and extracted onto the computer running PreonLab as a prerequisite. Please note that we require a version greater or equal to 2.6.8. for matching command line arguments.

[1] https://www.ffmpeg.org/



FFmpeg File: C:/ffmpeg/bin/ffmpeg.exe
Output File: D:/output.mp4
Category: Visualization
Source: PreonRenderer_1/CamPersp
Start Frame: 12
End Frame: 20
Input Framerate: 50
1x playback speed
Video Framerate: 50
Quality: High
Open Output Folder ☑
Run in Background □

Figure 78: 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 78). 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.

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 1*x* 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 $1/2$x 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*.[2] For lower settings, the file sizes of the

[2] 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

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.[3] 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. Note that if the path to the FFMpeg executable is encapsulated in quotes (i.e., ***"<path>" <arguments>***) and you want to paste the command into the Windows PowerShell, you have to add a preceding call operator (i.e., ***& "<path>" <arguments>***).

17.10.1 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 due to the chosen particle size and sample size of the sensor. This might lead to several video players not playing back the video.

lossless mode sets the CRF parameter to 0.

[3] 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.

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 Status file

When you run a simulation on PreonNode or PreonCLI, you can be informed of a running simulation or postprocessing run via a status file. This is enabled via the command-line option `--statusFile` and specifying the path to a JSON file.

Example: `./PreonCLI ~/exampleScene.prscene --statusFile ~/status.json`

With the option enabled, the specified JSON file will contain the current state of the simulation at each frame. The file includes the following pieces of information: the version of the file, the mode in which PreonCLI or PreonNode is currently running (*simulation* or *postprocessing*), the state of the simulation (*started*, *processing*, *finished*, *canceled* and *error*), the start frame, the end frame and the current frame which is being simulated. The contents of the file are *overwritten* at each frame with the current information. An example file can be seen below. Please note that the example file is split with line breaks only for visual purposes, the content of the file outputted from PreonCLI or PreonNode is contained on a single line.

```
{
"version": "1.1",
"mode": "simulate", "state": "processing",
"frame": 62, "start_frame": 0, "end_frame": 500
}
```

Please note that only errors which occur after the scene has been loaded will be reflected in the file. For more information, please refer to the command-line option description of PreonNode or PreonCLI.

18.5 Abort file

If you need to stop a simulation or post-processing run before the predefined end frame is reached, you could place a file named `ABORT.txt` in the simulation's working directory. The application will detect the file, and abort the run after finishing processing of the current frame. The created abort file will then be automatically removed.

18.6 Simulation logging

When PreonCLI or PreonNode is running, a more complete view of the simulation is now possible with the help of the new simulation logging feature. The scene statistics which are usually shown in the OSD during a simulation run in PreonLab are now logged at each frame for PreonNode and PreonCLI.

The logging is turned on by default and can be switched off using the `--no-progress` command-line option. Furthermore, using the `--logProgressEveryNthFrame` option you can log only every *n* frames.

18.7 Optional start parameters

You can use some optional parameters when starting PreonCLI. These are listed in the table below.

Start Parameter	What it does
--help	Displays help information. These are also displayed when you start PreonCLI without any parameters.
--version	Shows the version of PreonCLI.
--no-progress	Disables the progress bar and the simulation logging during simulation.
--licenseMain, *--lMain*	The main license that should be used. Possible values are *full* and *prepost*. If not specified, PreonCLI will try to checkout a suitable license.
--licenseBoost, *--lBoost*	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 *mpi_unlimited*, *sim_threads_max* or a number that specifies the number of additional licensed simulation threads that should be checked out. *sim_threads_max* will attempt to checkout the number of threads that are required to achieve the best performance on the machine.
--licenseModel, *--lModel*	The license model that should be used for the main license. Possible values are *node-locked* or *n* for node-locked licenses and *floating* or *f* for floating licenses. If omitted, both models can be checked out.
--licenseList, *--lList*	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 *--licenseMain*).
--postprocess	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.
--logDir	Sets a log directory other than the default.
--logProgressEveryNthFrame	Logs the progess of a simulation only every *n* frames.
--statusFile	Specifies the path to a JSON file, where the current status of a simulation or post-processing run will be reported.

Table 145: 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 from version 3.6 to 3.10. These versions (except 3.6) are the ones officially supported by the Python developers. We plan to continue supporting at least the Python versions, which are officially supported, at the time we release a particular PreonLab version. Hence, the support of Python 3.6 may be dropped at any time.

PreonLab and PreonCLI currently ships with a Python 3.8 distribution. Note, that we might update it in future versions, not later than fall 2024 when 3.8 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 3.8 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. Using an older Python version might not support features, that you can use on 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 on some Linux distributions you may have to enable the extra packages (EPEL) in order to be able to install *python3* and *python3-pip*.

- yum install -y epel-release

If your Linux distribution does not include an appropriate version of Python in its repositories, the safest way to install it is to compile it yourself. Please refer to Section 19.2.2 for this.

19.2.2 Installing Python from source

Python can usually be built very easily on many Linux distributions. This way you can have a more recent Python version and have it installed alongside the default system version. This can also make sense on systems where no other repositories should be added. The following instructions show how to compile Python 3.8 on a RHEL operating system 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/3.8.5/Python-3.8.5.tgz`
- tar xzf Python-3.8.5.tgz

Then compile and install it. Replace the */home/username/Python38* 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-3.8.5
- ./configure --prefix=/home/username/Python38
- make
- make altinstall

19.2.3 Installing Python on Windows

On Windows, simply download the Python 3.8 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` 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/Python38/bin/pip install ...

On Windows, you can refer to your *pip* executable in one of the following two manners:

- C:\Python38\python.exe -m pip install ...
- py -3 -m pip install ...

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 (*venv*s). 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-5.2/
```

19.3.1 Licensing

If you have installed PreonPy into your Python environment and import it from a Python script, it is not possible to select the license via command line arguments. Starting with version 5.0, the `import preonpy` statement does not automatically check-out a license. Consequently, you have the possibility to choose an appropriate license first via `preonpy.checkout_license(...)` before you start working with PreonPy. If you omit this, a default license is checked out as soon as a scene is opened.

19.3.2 Error Handling

In case of unexpected behavior, please always check the Preon log for warnings and errors. The log is printed in the Preonlab 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 π 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_v5_2_3.zip`
- `cd PreonNode_Linux_v5_2_3`
- `./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 parameters similar to PreonCLI, with the difference, that Python is not supported. Note that status file, abort file and simulation logging functionality is also available for PreonNode. Please refer to the respective sections in Chapter 18 for more information on these.

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_v5_2_3`
- `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
--*help*	Displays help information. These are also displayed when you start PreonNode without any parameters.
--*version*	Shows the version of PreonNode.
--*licenseMain*, --*lMain*	The main license that should be used. Possible values are *full* and *prepost*. If not specified, PreonNode will try to checkout a suitable license.
--*licenseBoost*, --*lBoost*	The license extension that should be used in addition to the main license. Possible values are *mpi_unlimited*, *sim_threads_max* or a number that specifies the number of additional licensed simulation threads that should be checked out. *sim_threads_max* 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.
--*licenseModel*, --*lModel*	The license model that should be used for the main license. Possible values are *node-locked* or *n* for node-locked licenses and *floating* or *f* for floating licenses. If omitted, both models can be checked out.
--*licenseNodeIndividual*, --*lNode*	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.
--*licenseList*, --*lList*	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 --*licenseMain*).

--timeout	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.
--busyWaiting	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.
--noPerformanceChecks	Disables performance checks executed at startup that detect typical bad configurations like running more than four PreonNode processes concurrently on the same machine, low network speed or using not enough threads.
--allowSceneSpecificThreads	Unless this option is specified, the **individual threads** property in the scene settings is ignored.
--testThreads	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.
--debugLogging	Enables more log messages intended for debugging.
--timingBasedLoadBalancing	Enables load-balancing based on timings reported by worker nodes. This can improve performance if the worker nodes operate at different speeds.
--postprocess	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.
--logDir	Sets a log directory other than the default.
--logProgressEveryNthFrame	Logs the progess of a simulation only every *n* frames.
--statusFile	Specifies the path to a JSON file, where the current status of a simulation or post-processing run will be reported.
--continue	Continues a previously stopped or canceled simulation. If passed, the scene statistics are considered and the simulation is continued from where it was left off.

Table 146: 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

In most cases, you should only create one process for each machine. Each process will utilize available CPU cores on the machine (or socket) using OpenMP, which reduces synchronization effort between the cores. Spawning too many processes per node can seriously harm the performance. By default, PreonNode will refuse to start if it detects that more than four processes are running on the same machine (this number was chosen because there are few systems with more than four sockets). Pass the argument --*noPerformanceChecks*if you want to allow it anyway.

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 *Cum. Update* and *Cum. Update Simulation*. *Cum. Update* is the overall computation time including post-processing, while *Cum. Update Simulation* 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 Heterogeneous clusters

It is possible to use machines with different hardware configurations for a single simulation. By default PreonNode will evenly distribute particles to all machines. In order adjust it to the capabilities of each machine, the `--timingBasedLoadBalancing` start parameter has to be passed. 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.

20.5.6 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.

20.5.7 Scaling

As mentioned, it is impossible to predict the performance scaling for every kind of scene. As a reference point, Figure 79 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 80. Feel free to download the scene[1] in order to run the benchmark on your cluster.

[1] Download MPI scaling benchmark *mpi_dive_portable.prscene* from `https://transfer.fifty2.eu/index.php/s/bhHEW07DSgTvt8f`

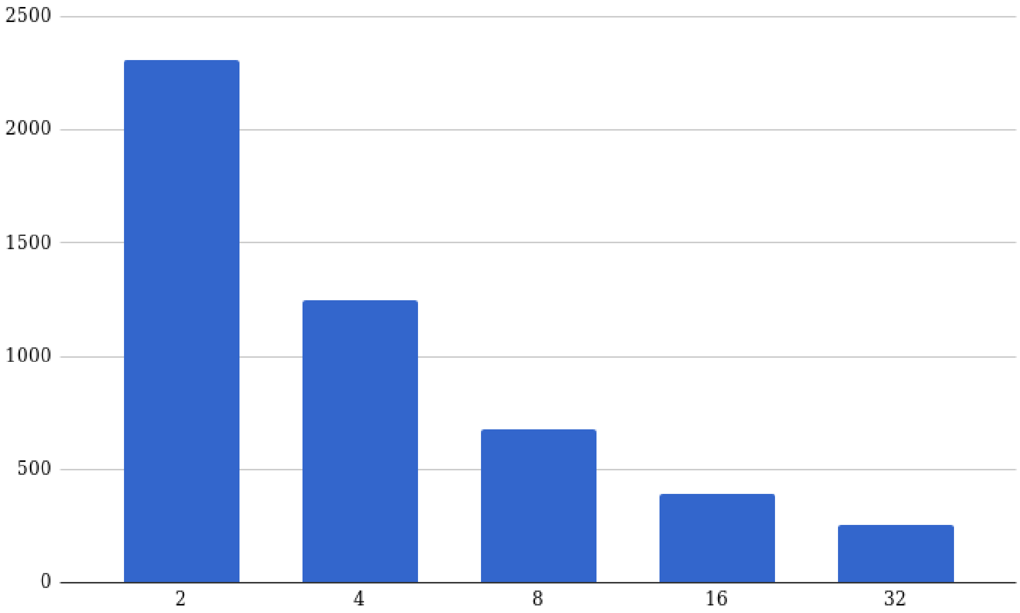


Figure 79: Total update times in seconds for a specified frame range of the dip coating benchmark. The tests were conducted on the High-Performance Computing Center Suttgart for 2, 4, 8, 16 and 32 nodes. Thereby, each node has 16 CPU cores.

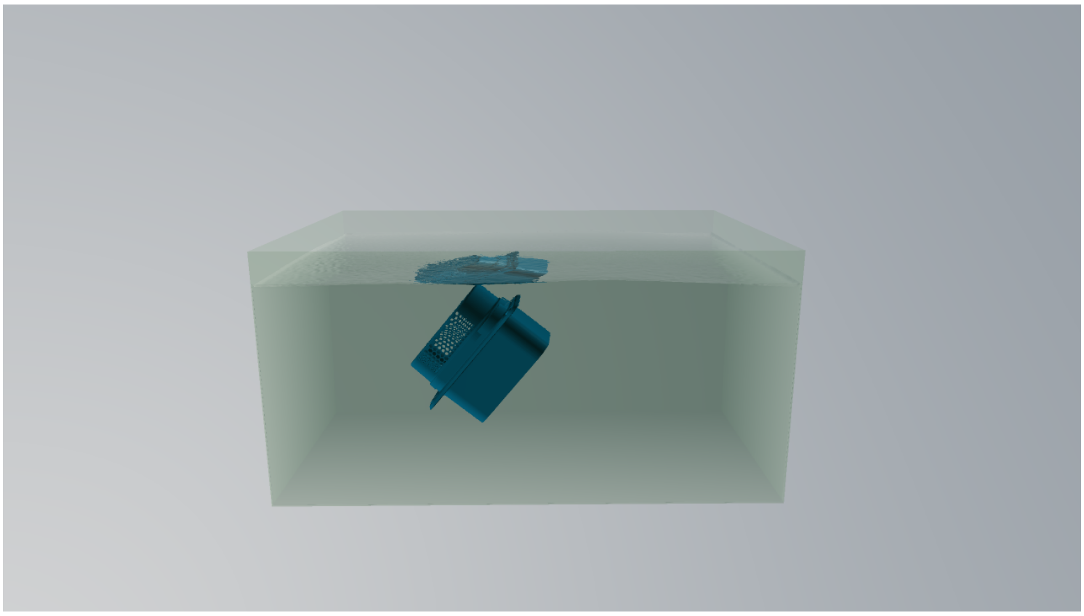


Figure 80: Example frame of the dip coating scene used for the MPI scaling benchmark.

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).

20.6.4 Running PreonNode with Intel MPI

In some situations Intel MPI applies processor binding which results in PreonNode only being able to use one CPU core. Passing `-genv I_MPI_PIN=off` to *mpiexec* disables processor binding entirely.

20.7 Dynamically changing set of nodes

The typical cluster environment performs computations on a fixed set of nodes that is assigned to one specific calculation for a certain amount of time. Some cloud-based providers offer discounted prices for idling nodes. However if you are adding such nodes to you cluster, you are doing this under the caveat that other users might claim these nodes by paying full price. In this case, it will be removed from your cluster on short notice (i.e., within minutes).

Since PreonNode uses MPI for distributed computation and MPI is also built around the concept of a fixed set of nodes defined at startup, it is not trivial to remove or add nodes during a simulation. The principal setup would consist of a loop that starts *mpiexec* with PreonNode in the background and then checks if either *mpiexec* returns or if the cloud provider sends a notification about changes in the node set. In the latter case, we can stop the *mpiexec* background process and start with a new set of computation nodes.

In this context, PreonNode should be started with `--continue` such that it automatically continues the simulation from where it was started. This also works if PreonNode was in the process of writing a frame to disk. Here, it would start one frame earlier. If the scene was simulated before and contains old scene statistics, you have to make sure that the simulation starts from the beginning. This can be done by not passing `--continue` in the first run or by removing the scene statistics file `scene.stats` in the scene directory before.

20.8 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, the force sensor and the Y+ 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.