24.2 Domain creation

The solver is capable of generating samples that involve multiple lattice structures (e.g., polycrystals). The entire simulation domain (i.e., the material sample) may consist of multiple lattice domains. The solver first constructs the individual lattice domains, then erases the "conflicting" sites in between based on $d_{\min}$.

When constructing a lattice domain, we first go over all the user-specified geometric regions (i.e. S_i) to find a bounding box that is aligned with the lattice vectors $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$. Here, it is noteworthy that all the geometric objects supported by the code (except for planes) are convex. Therefore, any box that contains the vertices of the object contains the entire object. For spheroids, cylinder-cones, and cylinder-with-spherical-caps, we first create an intermediate bounding box aligned with their own axes, as shown in Figure 49. Then, the final bounding box is defined to be the smallest box that is aligned with the lattice vectors, and contains all the vertices of the intermediate bounding box. This solution is not optimal, but easy to implement and widely applicable.

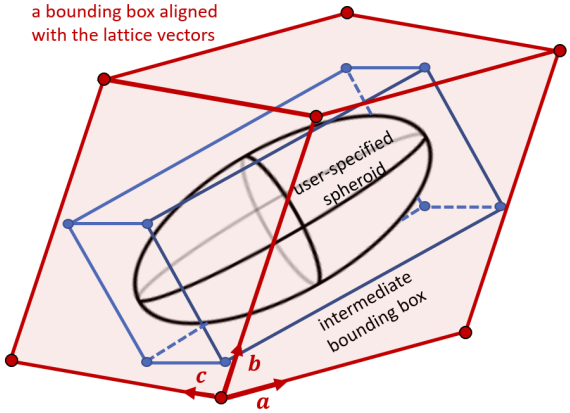

Figure 49: Generation of a bounding box aligned with lattice vectors for a spheroid.

25 Parallel visualization on computer clusters using Paraview

The results of M2C can be visualized on a comptuer cluster using Paraview, without transferring the data back to the user's local computer. Also, the results of M2C can be visualized in real-time, while the simulation is still running.

As an example, we assume that the simulation is performed on the Tinkercliffs cluster at Virginia Tech, and the local computer is a Windows PC with Ubuntu installed as a *Windows Linux Subsystem*. For other configurations, the general method is the same.

25.1 Preparation (One-time effort)

(1) Ensure the local computer and the cluster have the same version of Paraview. At the time of writing, Tinkercliffs has both Paraview/5.8.0 and Paraview/5.9.1. The user should install either one on their local computer.

(2) On the local computer, launch Paraview. Then, click `[File]→[Connect]→[Add Server]`. Give this server a name. Set port to `11111`. Click `Configure` (Fig. 50). Then, set *Startup Type* to *Manual*, and click `Save`. Now, a configuration with the given name should appear in the window. Close Paraview.

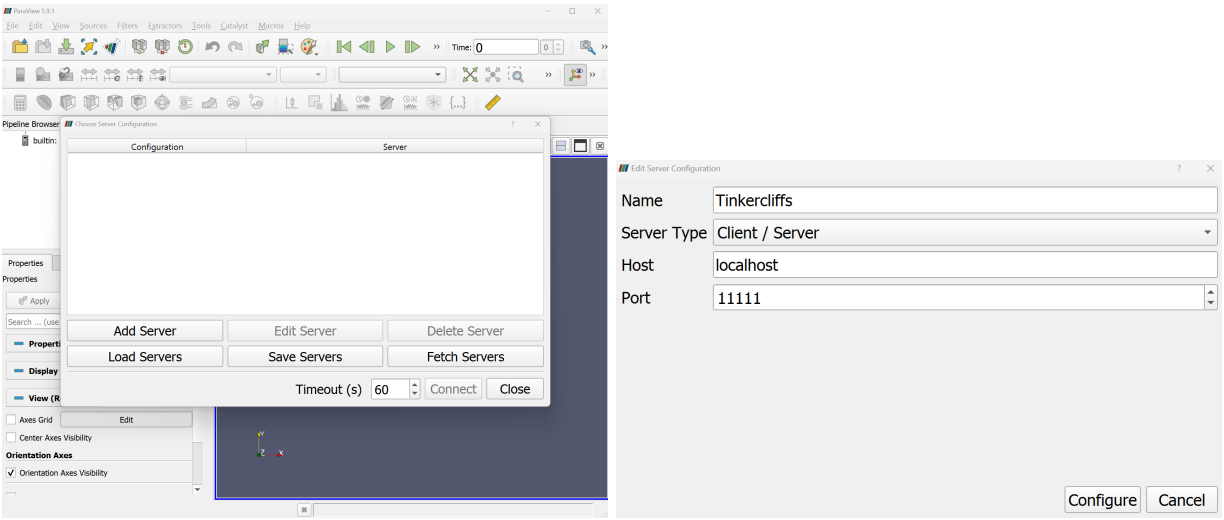

Figure 50: Visualization preparation: Adding a server.

(3) On the local computer, launch a Ubuntu terminal. In an appropriate repository (e.g., the home repository), create two bash script files: `paraview_step1` and `paraview_step2`. The first file should contain (with the correct user name):

```
ssh -t kevinw3@tinkercliffs2.arc.vt.edu ". /etc/bashrc;
module load slurm;
sbatch --output=/dev/null --error=/dev/null paraview.sh;
bash -l"
```

The second file should contain (with the correct user name)

```
ssh -L 11111:$1:11111 kevinw3@tinkercliffs2.arc.vt.edu
```

Here, it is assumed that the Paraview server on the cluster opens a port with code `11111`. This is true for Tinkercliffs at the time of writing. For a different cluster, one needs to find the correct port number. This can be done by running Paraview on the cluster in an interactive session.

These two files can also be found on Tinkercliffs in

```
/projects/wang_aoe_lab/KevinWang/ParaviewScripts
```

(4) On the computer cluster, create a sbatch file in the home directory for launching Paraview. For example, it may be named `paraview.sh`, with the following content.

```
#!/bin/bash

#SBATCH --nodes=1
#SBATCH --ntasks-per-node=32
#SBATCH -t 3:00:00
#SBATCH -p dev_q
#SBATCH -A wang_aoe_lab

# Add any modules you might require.
```

```
module load ParaView/5.9.1-foss-2021a-mpi
cd $SLURM_SUBMIT_DIR
mpirun -n 32 pvserver
exit;
```

Again, an example can be found on Tinkercliffs in the same folder mentioned above. This example will launch Paraview on 32 CPU cores for a period of 3 hours. These numbers can be changed. Also, this example launches `ParaView/5.9.1` on the cluster. To launch a different version (that has been installed on the cluster), the `module` in the script needs to be modified.

25.2 Visualization

(1) On the local computer, launch a Ubuntu terminal. Run the first script, i.e., `paraview_step1`. In a few seconds, the log-in node of the cluster should be reached (Fig. 51).

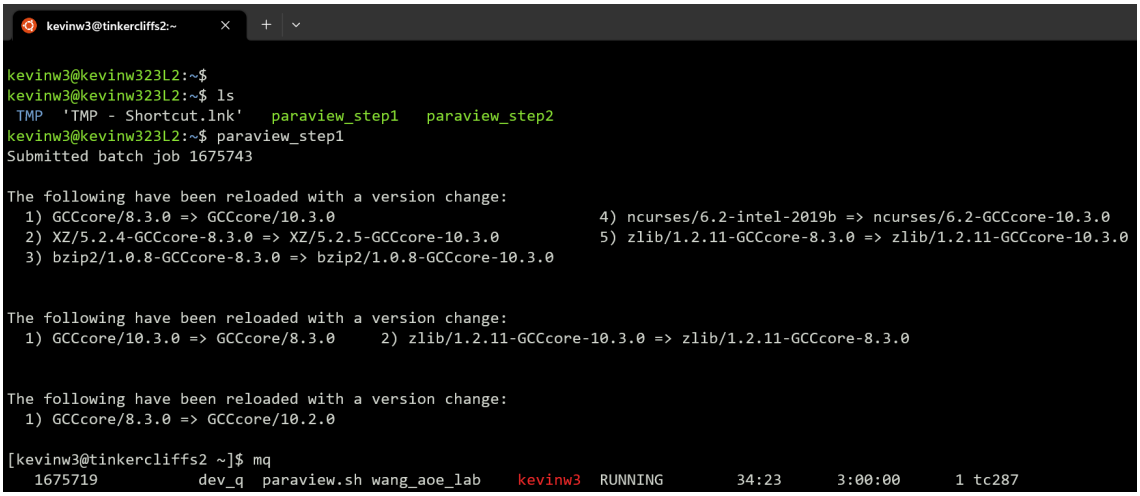

Figure 51: Remote visualization: Launching Paraview on the cluster.

Give it a few seconds, then check the job queue. It may take some time for the job to start. On Tinkercliffs, it usually takes just a few seconds. Once the job starts, make a note of the compute node number (e.g., `tc287` in Fig. 51).

Type `exit` to get back to the Ubuntu terminal on the local computer. Then, run the second script **with the compute node number as an input**. That is, run

```
./paraview_step2 tc287
```

(Do NOT forget to provide the compute node number.)

Again, it should log onto the cluster, as shown in Fig. 52. Now, the connection between the local computer and the cluster has been established. Do not log out of the cluster from this terminal. Otherwise, the connection will be lost.

(2) On the local computer, launch Paraview. Click `[File]→[Connect]`. Choose the configuration created in Preparation Step (2). Click `Connect`. Now, a warning about not being able to run remote rendering may appear. This is fine in most cases (Correct?). It means the cluster will process the (big) data, while the local computer will render the graphics. Despite this warning, Paraview should have access to the data on the computer cluster (Fig. 53).

```
[kevinw3@tinkercliffs2 ~]$ mq
   1675719          dev_q  paraview.sh wang_aoe_lab    kevinw3  RUNNING       34:23      3:00:00      1 tc287
[kevinw3@tinkercliffs2 ~]$ exit
logout
Connection to tinkercliffs2.arc.vt.edu closed.
kevinw3@kevinw323L2:~$ paraview_step
paraview_step1  paraview_step2
kevinw3@kevinw323L2:~$ paraview_step2
Last login: Tue Sep 26 20:03:06 2023 from 172.21.188.160
+-------------------------------------------------------------------------+
| This computer is the property of Virginia Polytechnic Institute and State |
| University. Use of this equipment implies agreement to the university's   |
| Acceptable Use Policy (Policy 7000). For more information, please visit:  |
| https://vt.edu/acceptable-use.html                                      |
+-------------------------------------------------------------------------+
+-------------------------------------------------------------------------+
| NOTE: VT Enterprise Directory Password authentication requires a DUO      |
|       second factor challenge.  After your password is provided, you      |
|       will receive a DUO challenge.                                       |
+-------------------------------------------------------------------------+


The following have been reloaded with a version change:
  1) GCCcore/8.3.0 => GCCcore/10.3.0                      4) ncurses/6.2-intel-2019b => ncurses/6.2-GCCcore-10.3.0
  2) XZ/5.2.4-GCCcore-8.3.0 => XZ/5.2.5-GCCcore-10.3.0    5) zlib/1.2.11-GCCcore-8.3.0 => zlib/1.2.11-GCCcore-10.3.0
  3) bzip2/1.0.8-GCCcore-8.3.0 => bzip2/1.0.8-GCCcore-10.3.0


The following have been reloaded with a version change:
  1) GCCcore/10.3.0 => GCCcore/8.3.0     2) zlib/1.2.11-GCCcore-10.3.0 => zlib/1.2.11-GCCcore-8.3.0


The following have been reloaded with a version change:
  1) GCCcore/8.3.0 => GCCcore/10.2.0

[kevinw3@tinkercliffs2 ~]$
```

Figure 52: Remote visualization: Setting up a connection channel.

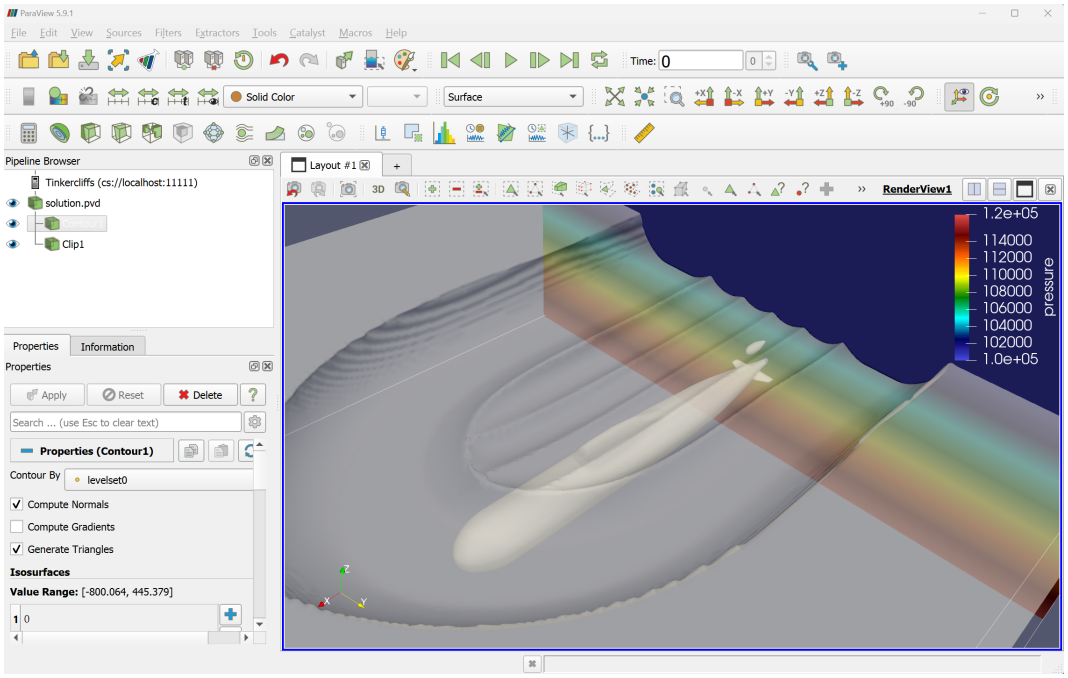

Figure 53: Remote visualization: Launching Paraview client on the local computer.

(3) Keep in mind that the Paraview server (on the cluster) will stop running and release the allocated processors when

- the requested job time has elapsed, or
- the Paraview client (on the local computer) is closed.