

Step-by-step guide to produce a 3-D visualization of NAME particle simulations using ParaView

This document provides a step-by-step guide for producing a minimal example of a 3-D visualization of desert locust swarm flight simulations conducted with the UKMO's Lagrangian Particle Dispersion Model NAME. It aims at providing an introductory example that may serve as a starting point for automated production of 3-D visualizations of NAME simulations using the open-source visualization framework ParaView.

Contents

Test-Data.....	1
Step-by-step guide	2
(1) Download, Install & Start ParaView.....	2
(2) Conduct NAME simulation & data-conversion or copy the test data for this minimal example	2
(3) Visualization of NAME topography on 3-D sphere.....	2
(4) Save Configuration of Visualization to ParaView state-file	7
(5) 3-D visualization of particle trajectory data.....	7
(6) Visualize country borders by adding administrative boundaries	8
(7) Define a title for your animation	8
(8) Animate through time	9
(9) Save the animation to file	12
(10) Automate the 3-D Movie generation	13

Test-Data

The folder /3D_NAME_DesertLocust_ParaView contains all required data & files for this minimal example:

- /Paraview State-File
 - Configuration file for ParaView session for reproducing the 3-D visualization in Movie 8
- /Python script for 3-D movie generation
 - ParaViews provides a “tracing” feature that allows automated generation of Python scripts to reproduce a set of GUI commands. The Python script provided here was obtained by tracing the GUI ParaView commands to configure the 3-D scene as visualized in Movie 8.
- /Particle Trajectory Data
 - Sample NAME trajectories from a test simulation (converted to .csv as required for processing in ParaView)
- /Topography
 - Topography_global_NAME.nc: NAME Topography data (converted to netCDF4 as required for processing with ParaView)
 - Topography_color.json: colorbar for topography data
- /Camera Position
 - Files storing the camera positions as used for Movie xy
- /Administrative Boundaries
 - Sample data for plotting administrative boundaries on 3-D sphere
- Movie Files
 - Contains the movies as .ogv (default ParaView output) and .mp4 files

Step-by-step guide

(1) Download, install & start ParaView

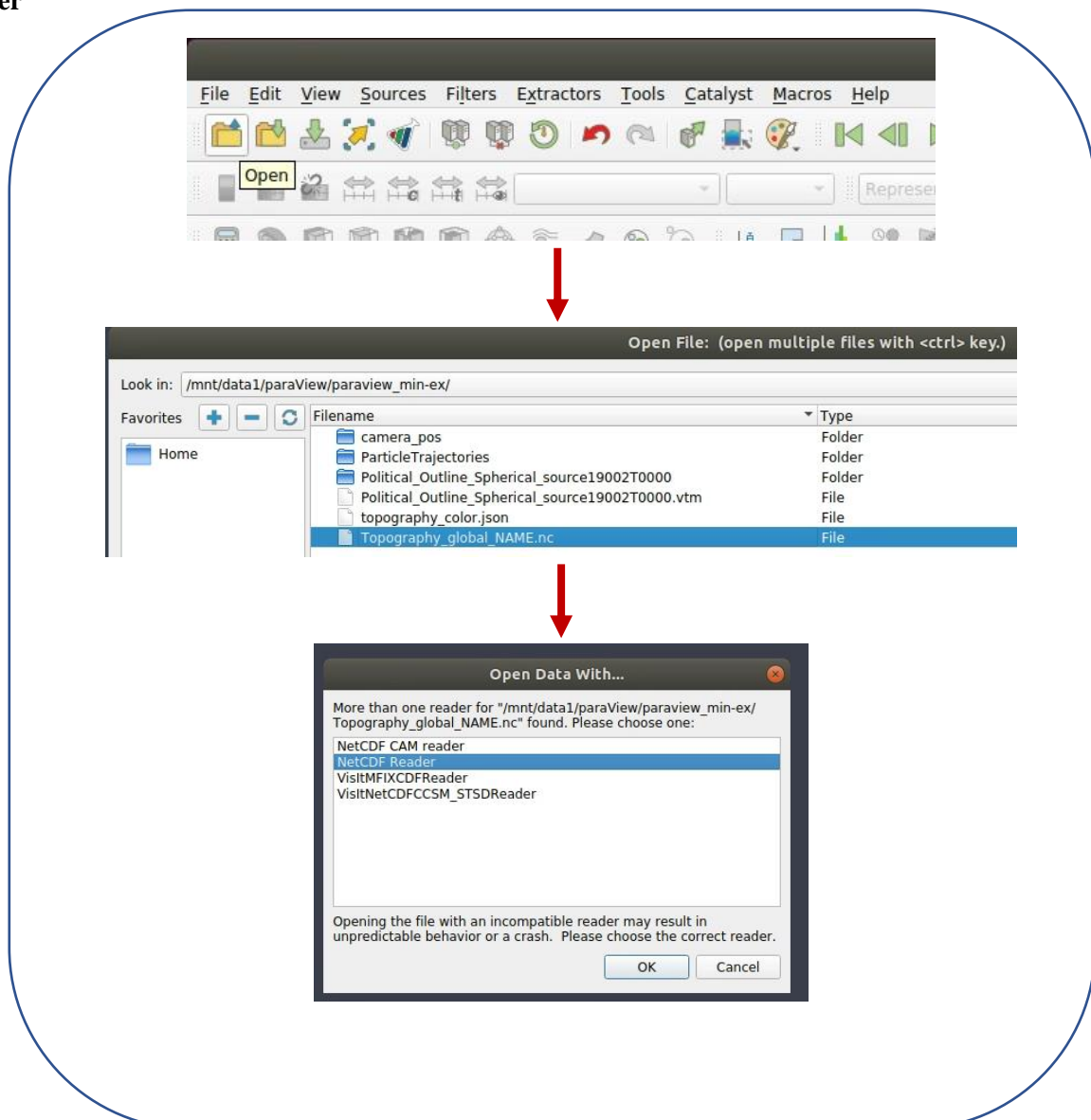
- Please consult the ParaView documentation material for installation, configuration & basics.
 - o See: <https://www.paraview.org/tutorials/>
- We used ParaView version 5.10.0.

(2) Conduct NAME simulation & data-conversion or copy the test data for this minimal example

- If you run your own NAME simulation (or use a different atmospheric dispersion model such as HYSPLIT), then you need to convert the format of the output data such that it can be processed in ParaView. For this minimal example you need 3-D particle trajectories and 3-D topography data.
- The following Python scripts in the folder /Data Conversion provide examples that can be adapted to convert NAME output particle trajectory output and NAME topography data to the format required for processing in ParaView. The test data in /Particle Trajectory Data and /Topography provide examples of converted NAME data.

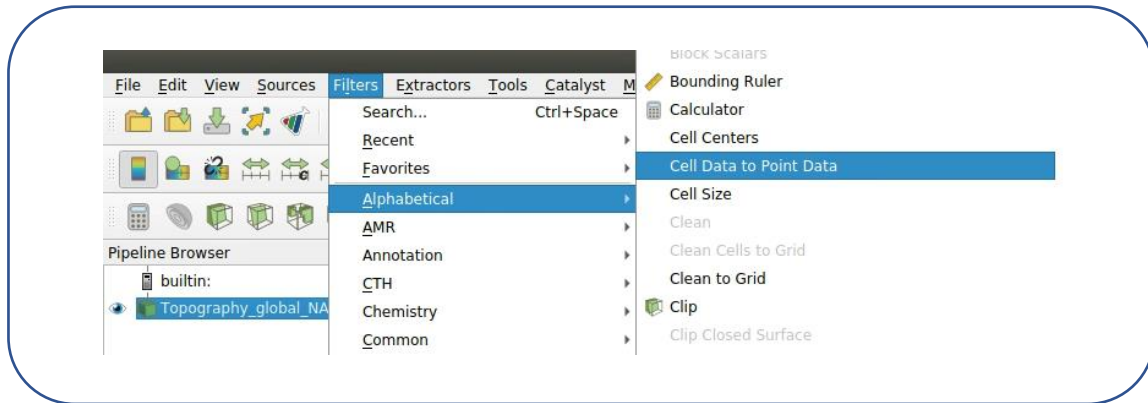
(3) Visualization of NAME topography on 3-D sphere

(3.1) Load the NAME topography (Topography_global_NAME.nc) into ParaView using the built-in netCDF file reader



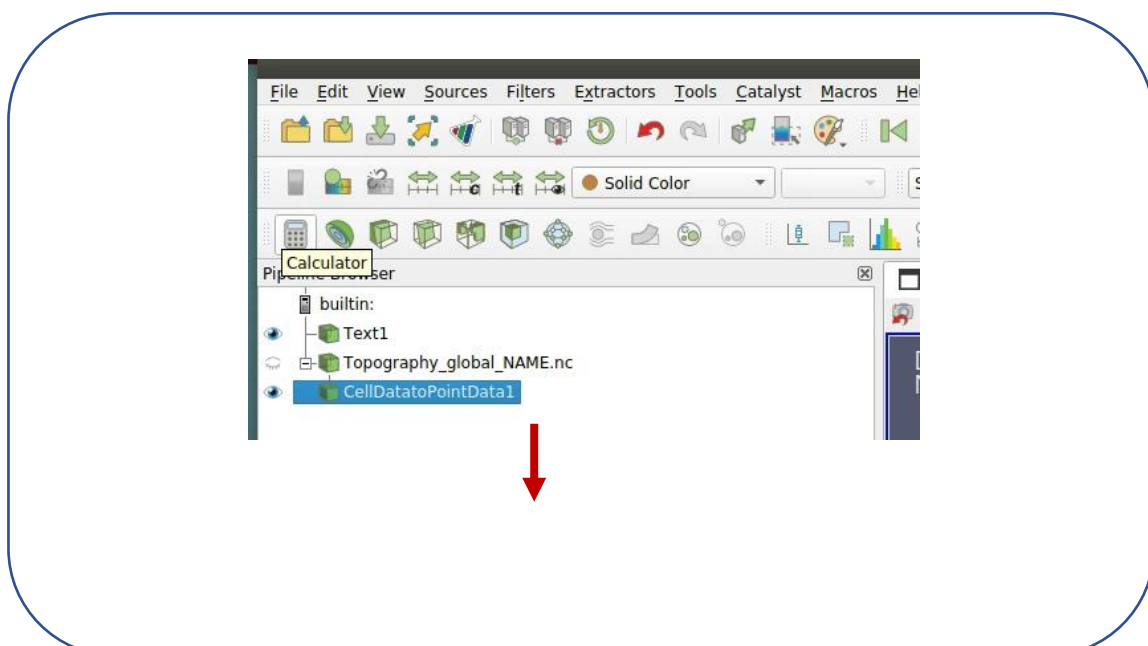
(3.2) Convert the topography data from cell data to point data. This is a necessary preparation for subsequent calculations for scaling the topography along the z-Axis for visualization on a 3-D sphere.

- Use the Filter “CellDataToPointData”
- Check “process all arrays”
- Click “Apply”

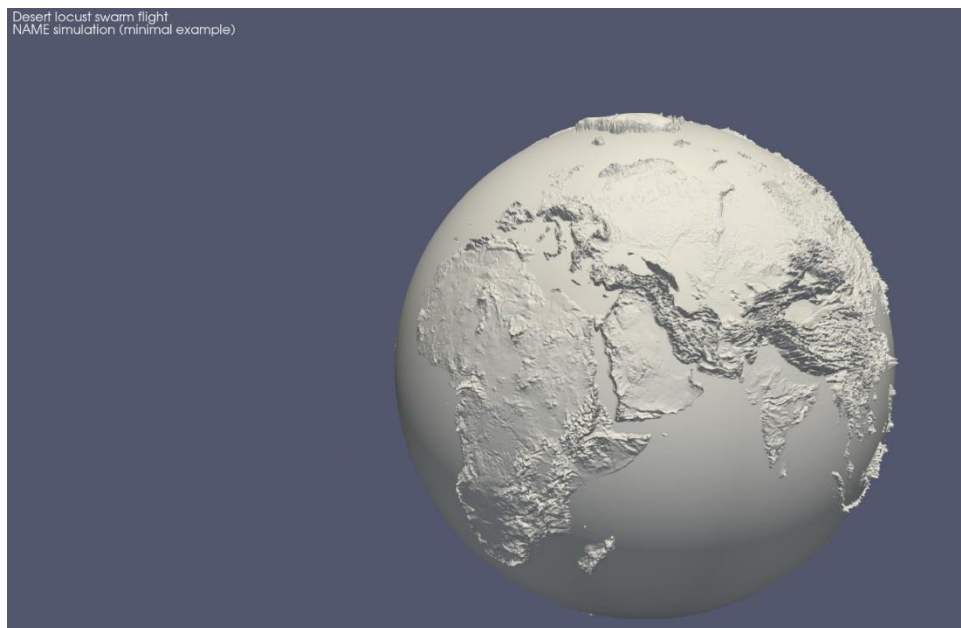
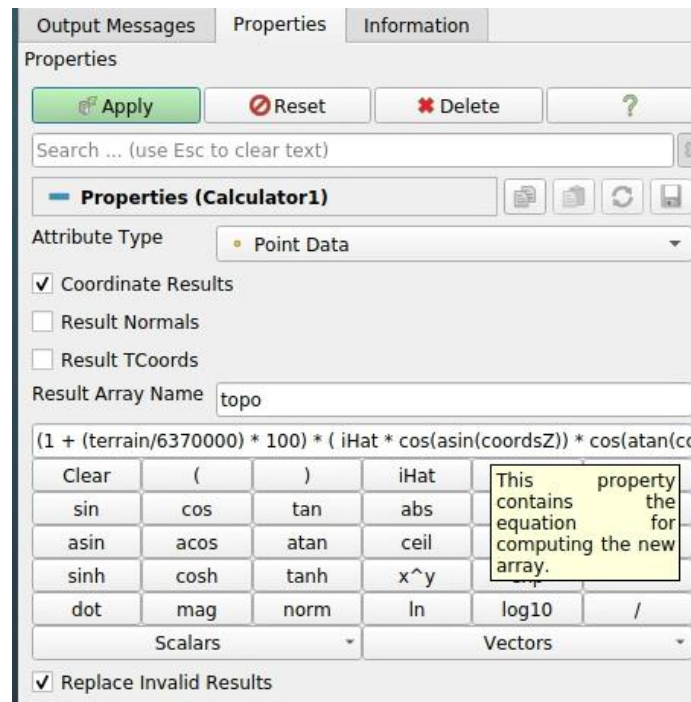


(3.3) Use the Calculator Tool to scale the altitude component (z-Axis) of the topography data for visualization on 3-D sphere representing the Earth

- Use the Filter “Calculator”
- Choose Attribute Type: “Point Data”
- Check “Coordinate Results”
- Choose a name for the array that holds the results of the calculator filter
- Enter the following extrusion formula into the calculator
 - o $(1 + (\text{terrain}/6370000) * 50) * (i\text{Hat} * \cos(\text{asin}(\text{coordsZ})) * \cos(\text{atan}(\text{coordsY}/\text{coordsX})) * \text{coordsX}/\text{abs}(\text{coordsX}) + j\text{Hat} * \cos(\text{asin}(\text{coordsZ})) * \sin(\text{atan}(\text{coordsY}/\text{coordsX})) * \text{coordsX}/\text{abs}(\text{coordsX}) + k\text{Hat} * \text{coordsZ})$
 - o For details, see: <https://docs.dkrz.de/doc/visualization/sw/paraview/Filters/extrusion/index.html>
- Click “Apply”

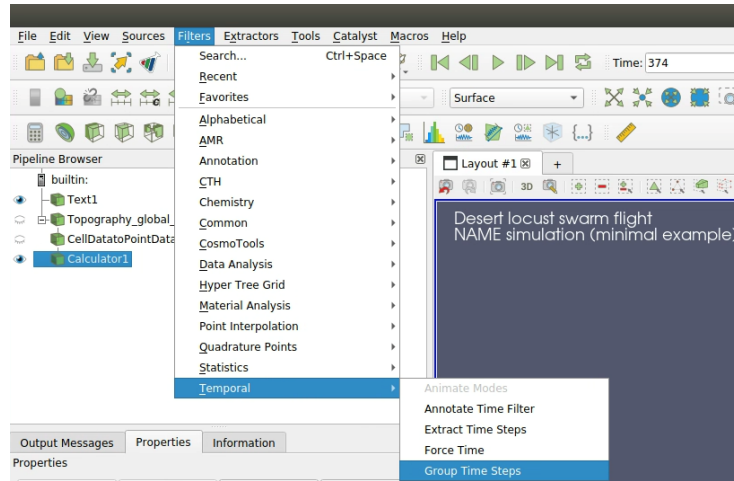


Step-by-step guide to produce a 3-D visualization of NAME particle simulations using ParaView



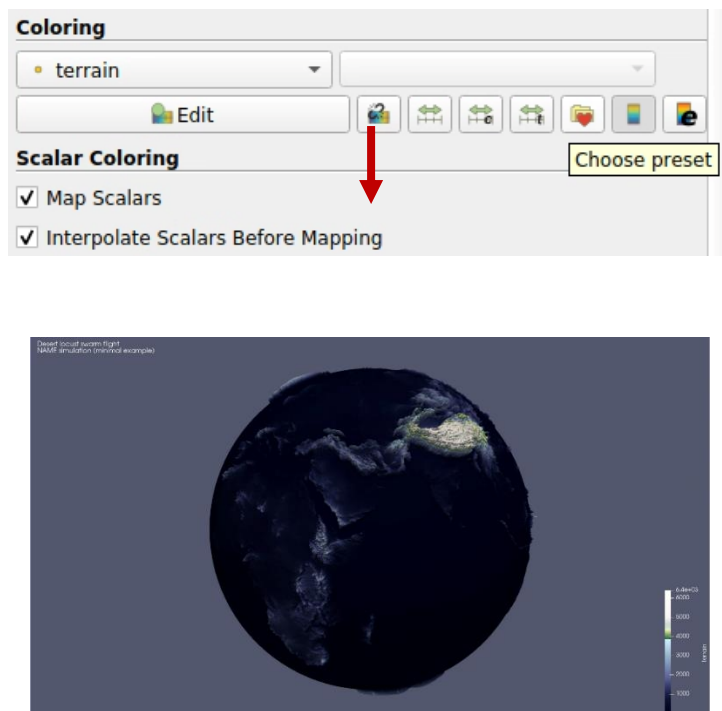
(3.4) Ensure that your topography data does not contain a time-dimension or group the time-steps

- Use the Filter “GroupTimeStep”
- Click “Apply”



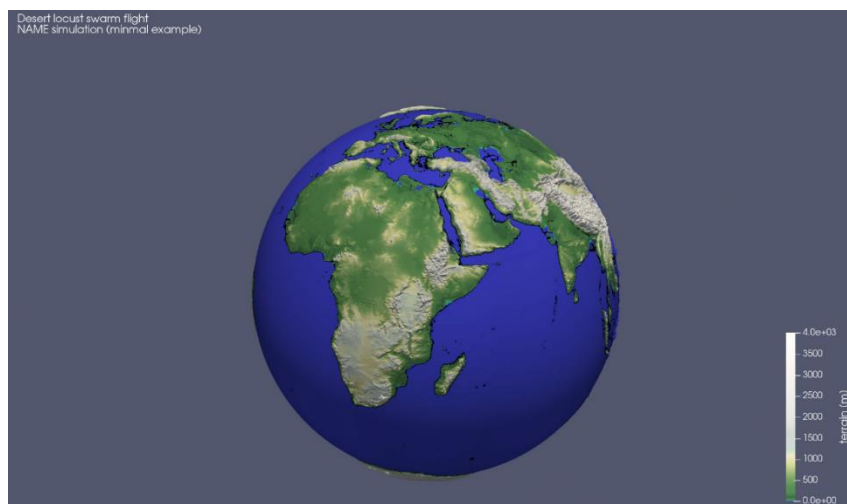
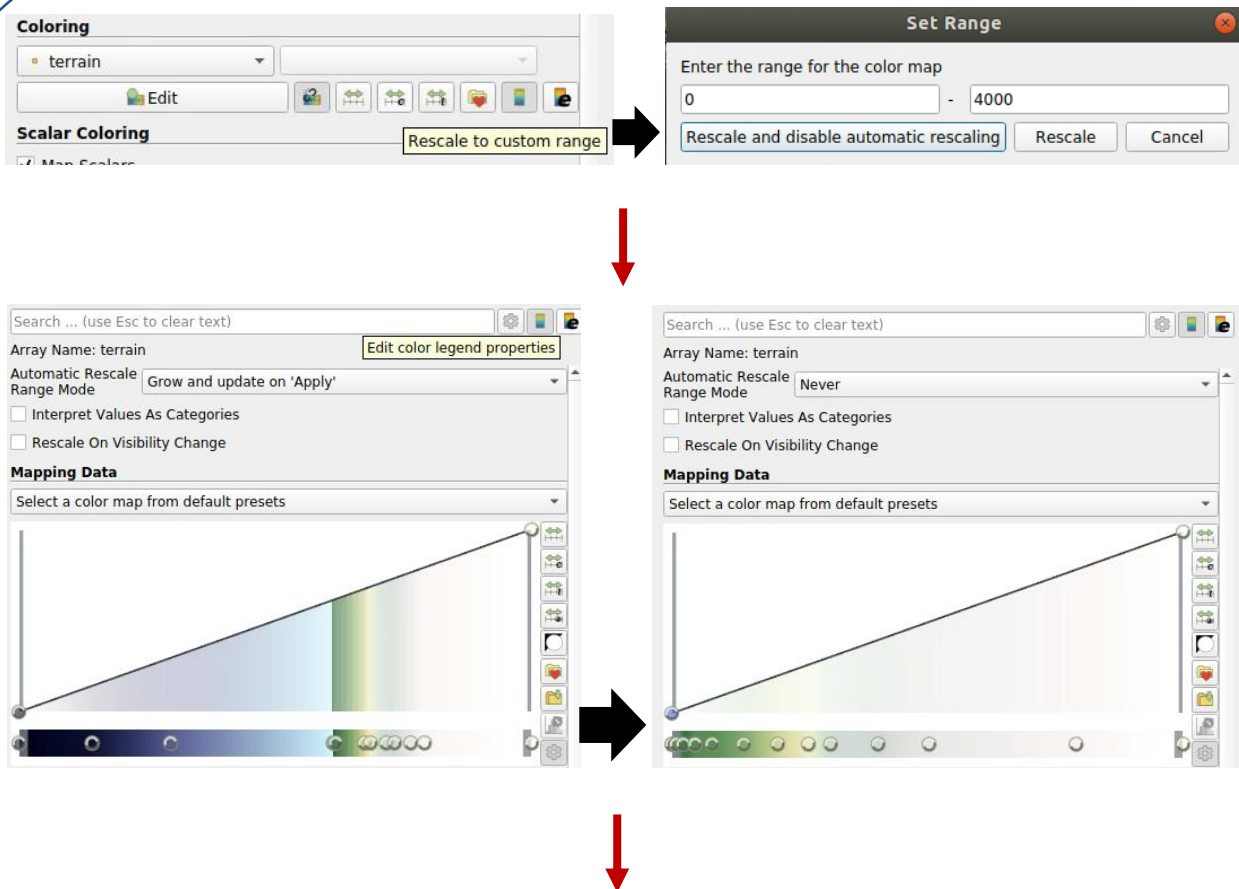
(3.5) Color the terrain

- In Properties → Coloring, select the topography data (name here “terrain”)
- Choose Preset colorbar
- Import the “Topography_color.json” as provided
- Apply HSV color map



Step-by-step guide to produce a 3-D visualization of NAME particle simulations using ParaView

- Adjust the colormap to your data-values such it matches your visual preferences, e.g.:
 - o rescale the custom range from 0-4000 and disable automatic rescaling
 - o adjust color transfer function values
- Edit color legend properties according to your preferences, e.g.:
 - o Change the name of the colorbar to “terrain [m]”
- More details on colormaps in ParaView
- <https://www.kitware.com/using-the-color-map-editor-in-paraview-the-basics/>
- <https://docs.dkrz.de/doc/visualization/sw/paraview/Colormap/index.html>



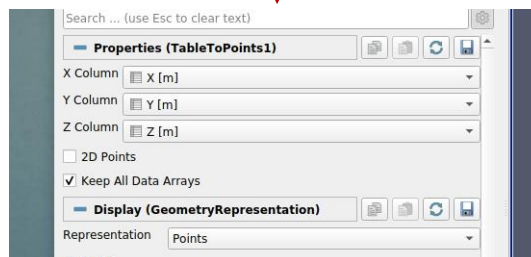
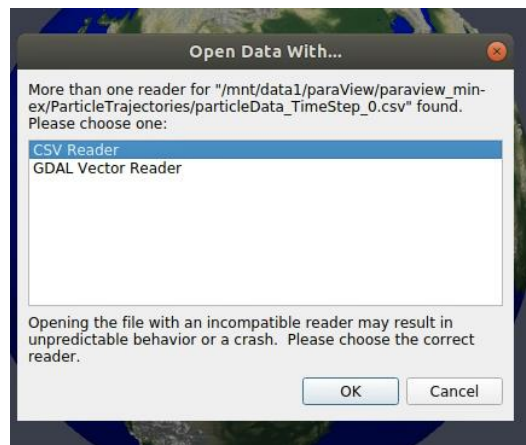
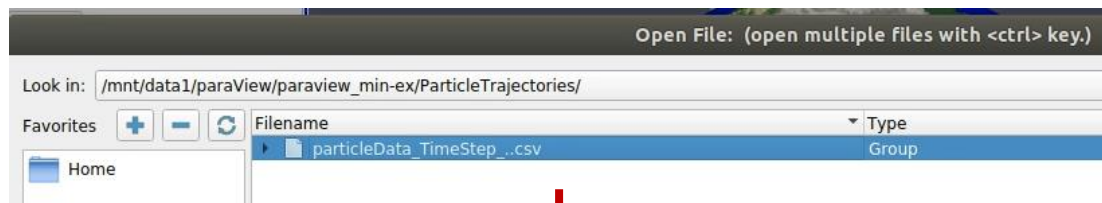
(4) Save configuration of visualization to ParaView state-file

- Save the current state of your visualization to a ParaView “State-file”. The state-file saves all configuration settings of your current visualization. For example, you could load the state-file to reproduce the above visualization of the topography. Save the current configurations as you go along, similar to how you would save any other file as you work on it.

(5) 3-D visualization of particle trajectory data

(5.1) Load the particle trajectory data

- Ensure that particle positions are provided in Cartesian coordinates (see example file)
- Use the ParaView CSV reader to open all .csv files with particle trajectory data. If the filenames contain a regularly order sequence of time-steps, then ParaView should load all files into a single dataset if you point the CSV reader to the folder containing all CSV files
- Click “Apply”
- Details on loading CSV files in ParaView: <https://www.olcf.org/using-paraview-to-view-csv-files/>

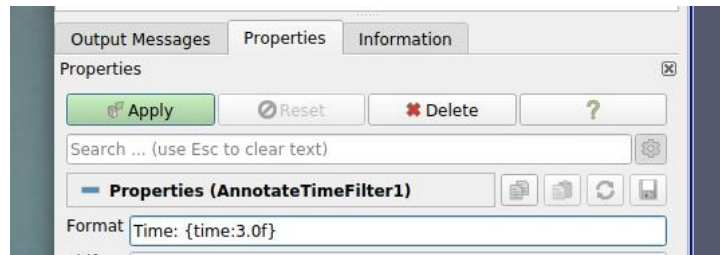


(5.2) Convert particle trajectory data from table to point data

- Use the filter “TableToPoint”
- Choose column in input data that match components in Cartesian coordinates. For the sample data provided choose “x[m], y[m], z[m]” for the X, Y, Z column, respectively
- Check “keep all data arrays”
- Choose Representation “Points”
- Choose configuration for visualization according to your preferences (e.g.: point size, render points as spheres, coloring)
- Click “Apply”

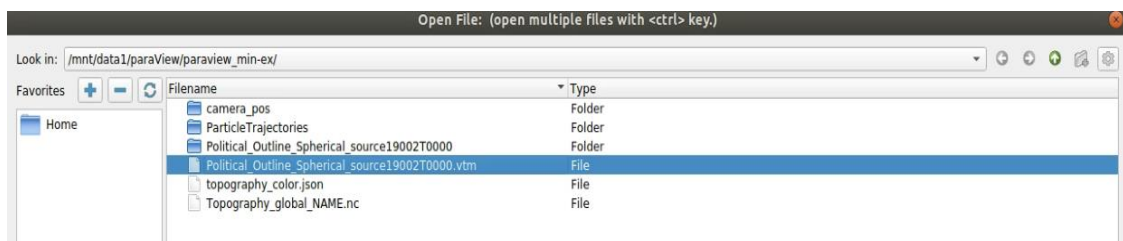
(5.3) Annotate time

- Use the filter “AnnotateTime”
- Format is in fmt styles. Default for time is float; change according to your data and preferences (e.g., for display of decimals)
- Choose text position. E.g.: right upper corner



(6) Visualize country borders by adding administrative boundaries

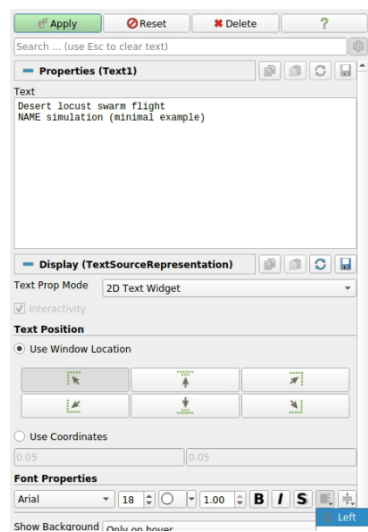
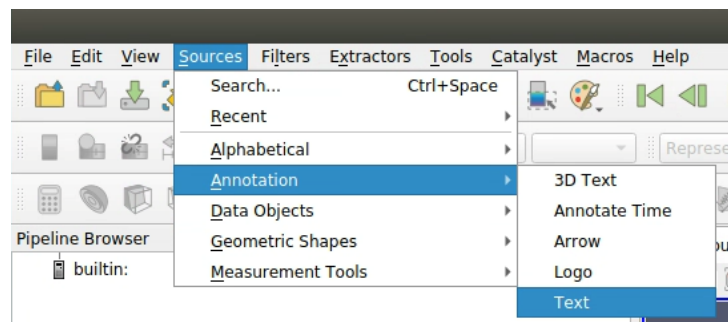
- Load the sample data provided in /Administrative Boundaries (vtkMultiBlockDataSet)



(7) Define a title for your animation

- Under “Sources” choose “Text”
- Enter title, e.g.: “Desert Locust Swarm Flight – Minimal Example”
- Set font properties
- Click “Apply”

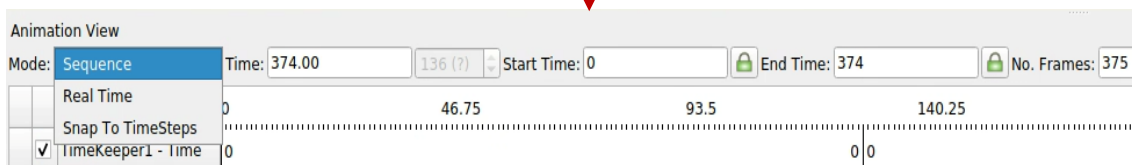
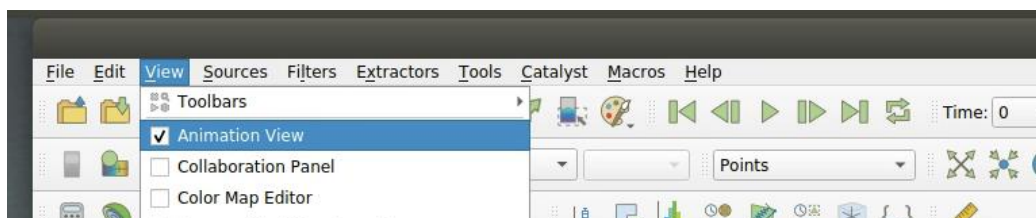
Step-by-step guide to produce a 3-D visualization of NAME particle simulations using ParaView



(8) Animate through time

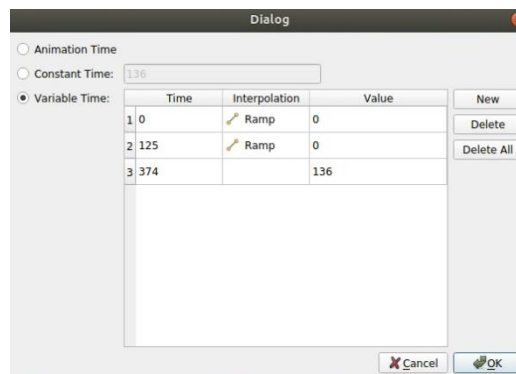
(8.1) Choose time sequence and number of frames

- Under “View” choose “Animation View” and then “Sequence Mode”
- Define the total number of frames per second for the time-animation. Adapt this to your data.



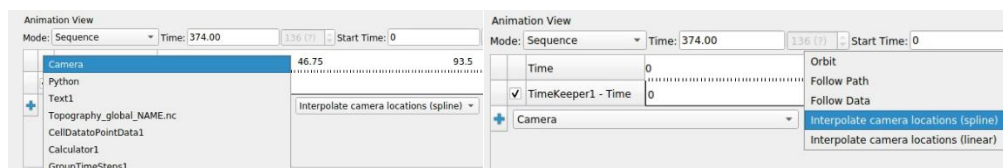
(8.2) Choose how to animate through time (data time-steps, animation time, variable time)

- Double-click to open the Dialog window.
- Choose “Variable time” (or adapt to your choice)

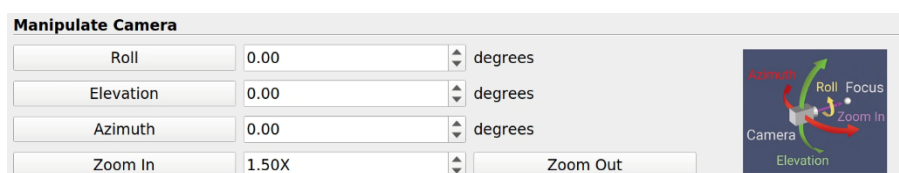
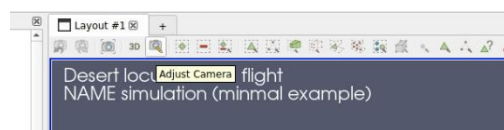


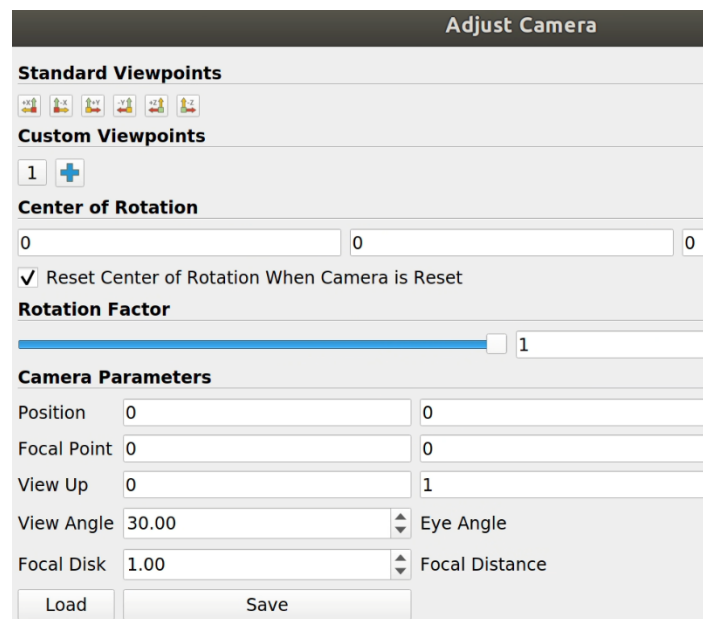
(8.3) Set the Camera Path

- Camera must be added in the “animation view” below the time-keeper. A camera can be added in one of three configurations: “Orbit”, “Follow Path”, and “Interpolate Camera Locations”
- Choose Interpolate Camera Location

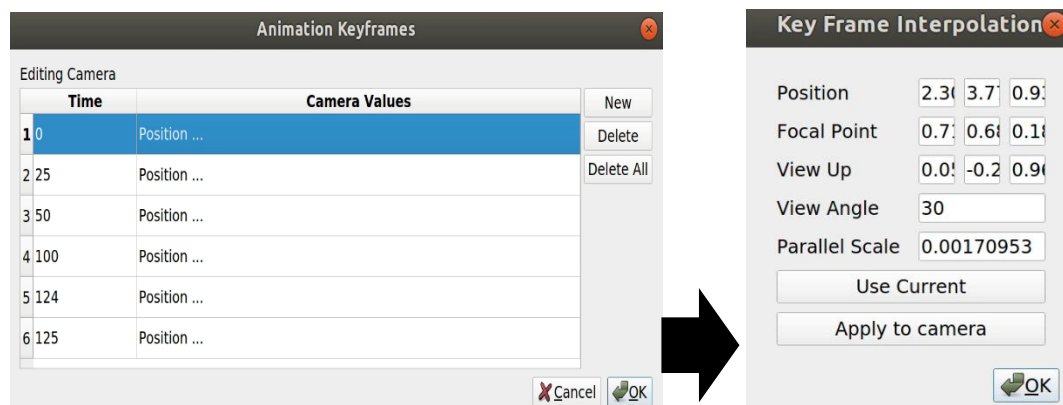


- Double-click to open the Dialogue window “Animation Keyframes”
- Define where the Camera Position should be at a set of fixed time-steps throughout the animation and choose interpolation type for determining the camera position in between these fixed time-steps.
- You can either define the Camera Position and Viewing angle by using the Scene View in the ParaView GUI (click on “adjust camera at in the toolbar at the top of the 3D Scene view)



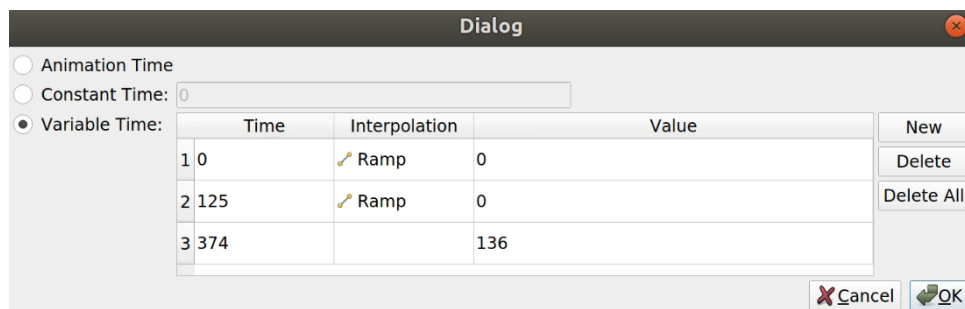


- or you can load the pre-defined Camera Position as provided in the sample data folder /Camera Position
- Every line in the “Animation Keyframes” dialogue window corresponds to one keyframe of your animation defined via a time and a camera position. You can load the pre-defined camera positions by using the load button at the bottom in the adjust camera window, while keep the Animation Keyframes window open. To modify keyframes, double-click on the entry. You can check if it is the desired camera position by clicking on “Apply to Camera”. Once finished, press “Use Current” followed by “ok”



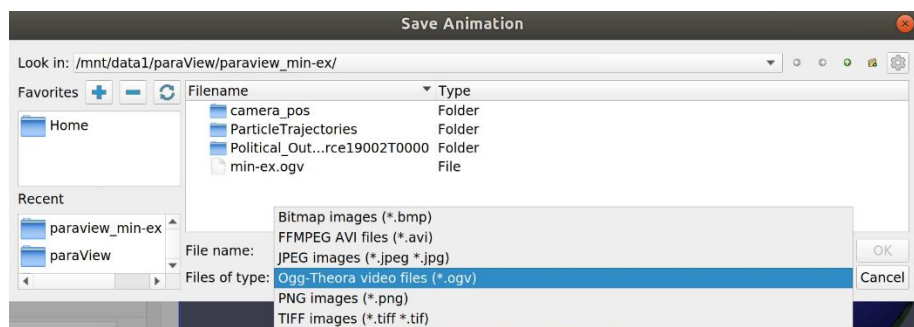
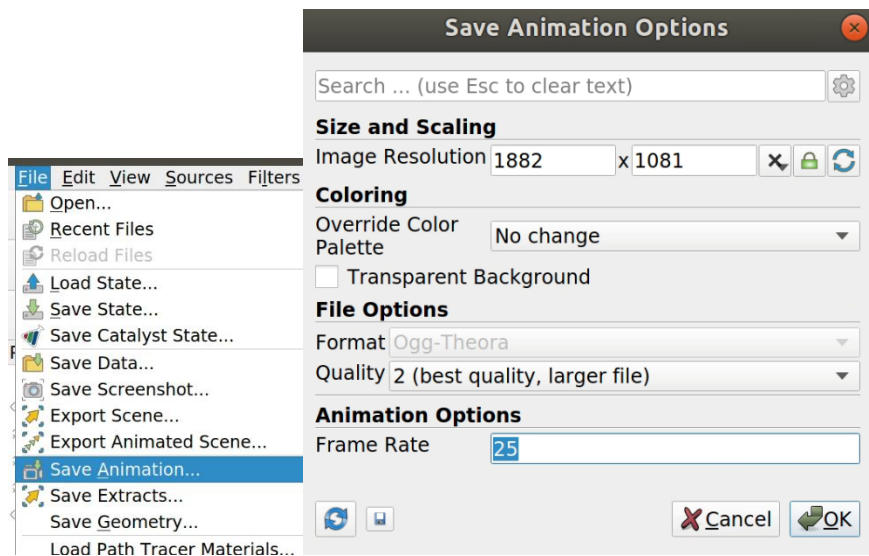
(8.4) Link the camera path to the time-animation

- Double-click the time-keeper
- Choose “Variable time”
- Keep the default interpolation method (linear ramp)
- For details, see: https://docs.dkrz.de/doc/visualization/sw/paraview/Camera_and_perspective/interpolate-camera-locations/index.html



(9) Save the animation to file

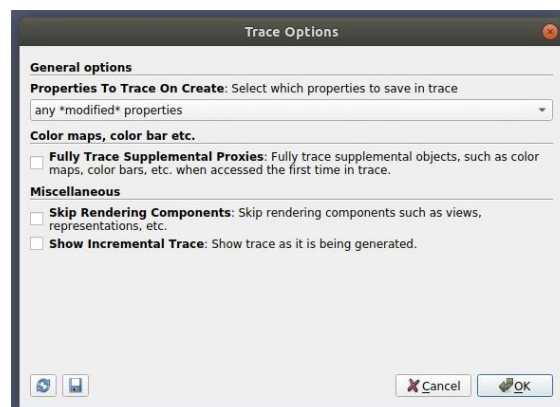
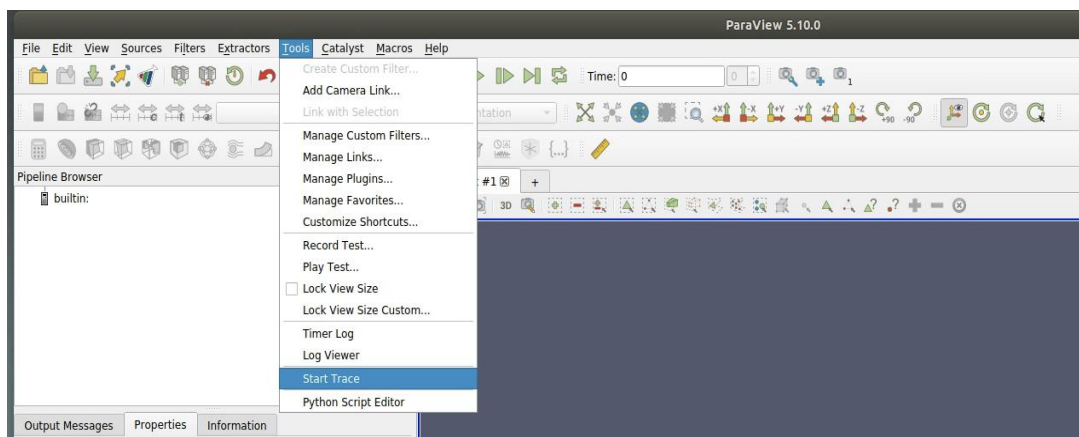
- Note the size of your ParaView GUI window is used as the default size of your animation view
- You can set custom values under “View” (e.g., aspect ratio, etc.)
- Under “File” and “Save Animation” choose where to save the animation file (default output file format is .ogv). The “Save Animation Options Window” can be used to modify key settings, e.g., frame rate.
- For details, see: <https://docs.dkrz.de/doc/visualization/sw/paraview/Export/saving-animations-screenshots/index.html>



(10) Automate the 3-D Movie generation

(10.1) Start a ParaView “Trace” session

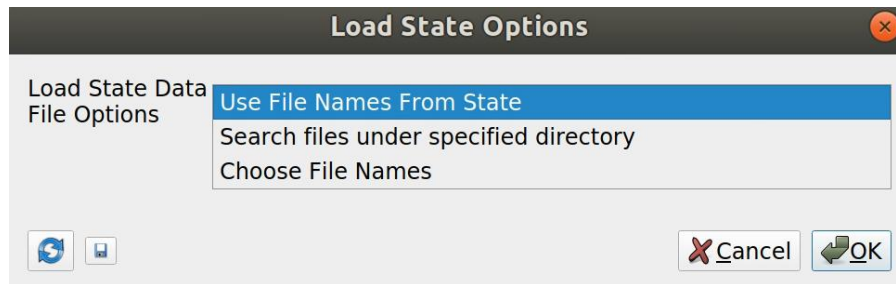
- ParaView provides the user with an option for automating the generation of 3-D visualizations with Python
- You can either code your visualization from scratch following ParaView Guidelines (see: <https://kitware.github.io/paraview-docs/latest/python/>). Or you can use ParaViews “trace” functionality to automatically generate a Python script that reproduces all commands that you input into the Graphical User-Interface.
- To start tracing your commands into a Python script go to “Tools” and then “Trace Start” and then click “Ok”



(10.2) Create visualization

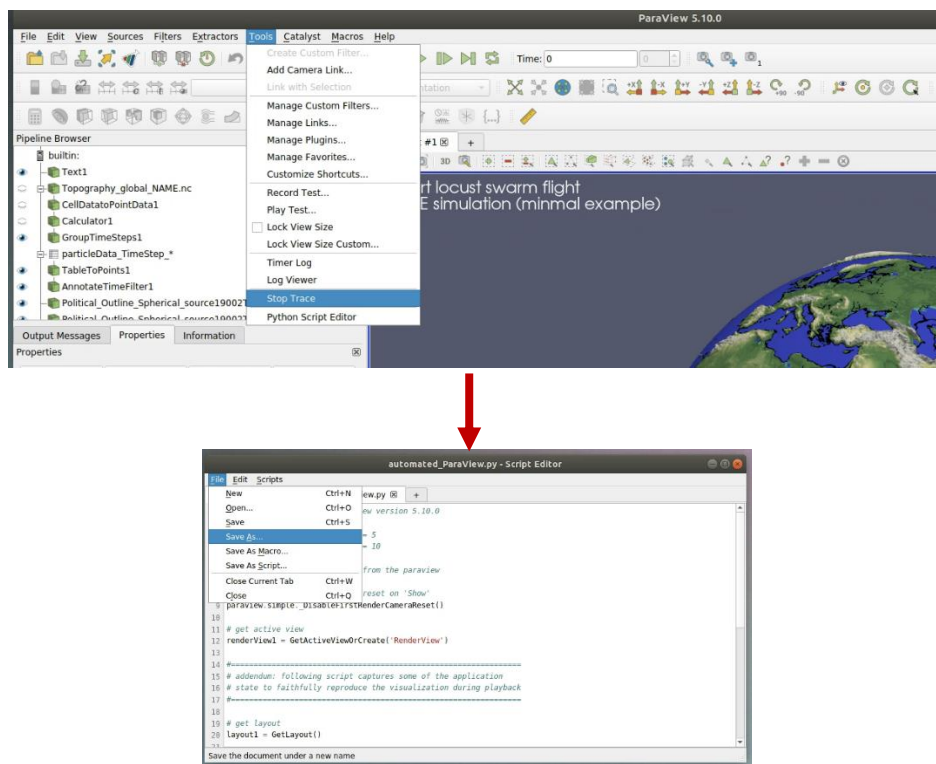
- Create your visualization pipeline whilst the tracer is running. For example, you could conduct all the steps described above (1-6) for loading a topography file and some particle trajectory data and coloring these appropriately. Or you can load a ParaView “State-File” (see above for details) to load all data and visualization configurations at once.
- When loading a state-file during a tracing session, there are two different options “Use File Names from State-File” or “Search Files under specified directory”. Use “Choose File Names from State File”

Step-by-step guide to produce a 3-D visualization of NAME particle simulations using ParaView



(10.3) Stop the trace session & save it

- Under “Tools” choose “Stop Trace”
- Click “Save As”
- For details, see: <https://www.youtube.com/watch?v=FTUBpqkC3Ss>



(10.4) Run Python script to generate 3-D visualization

- Use the ParaView-Python interface “PvPython” or “PvBatch” to run your visualization script
- For details, see:
 - o <https://kitware.github.io/paraview-docs/latest/python/quick-start.html#getting-started>
 - o https://www.paraview.org/Wiki/PvPython_and_PvBatch
 - o <https://docs.dkrz.de/doc/visualization/sw/paraview/technical/pvbatch/index.html>

```
~/ParaView_Versions/ParaView-5.10.0-MPI-Linux-Python3.9-x86_64/bin$ ./pvpython your_paraview_script.py
```