VIS 2023

# Installing TTK

**Christoph Garth**

University of Kaiserslautern-Landau

# TTK Architecture

TTK has several layers:

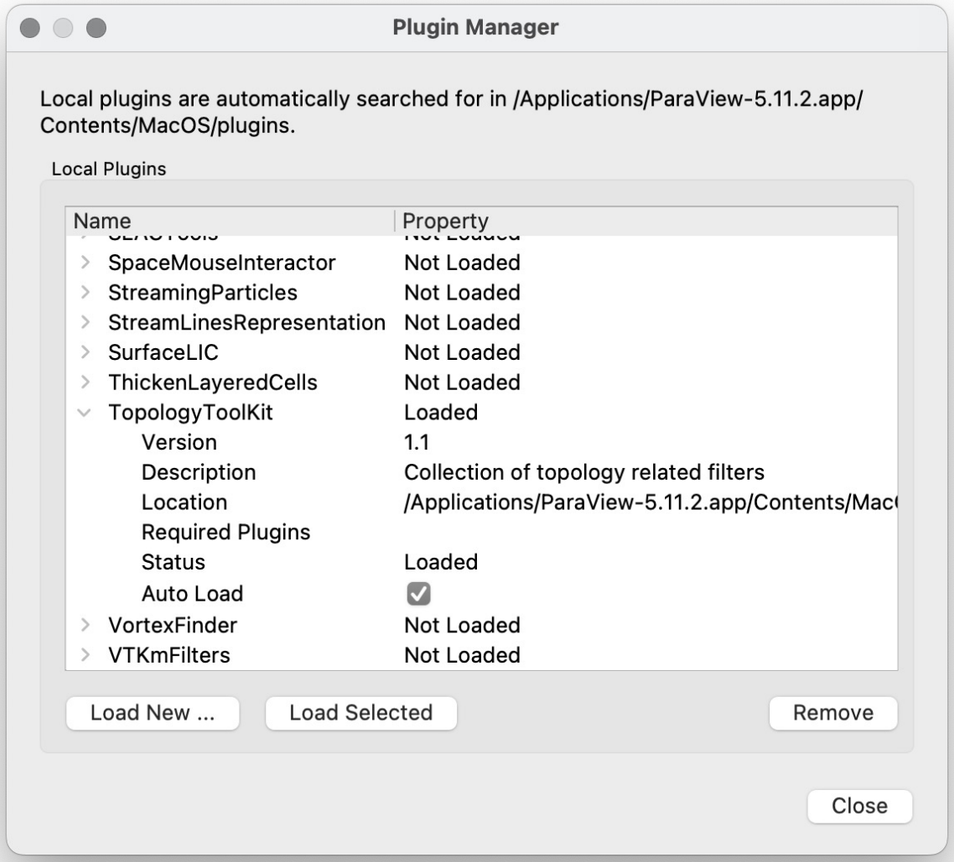
- A base layer for core algorithms and data structures
  - these are implemented in C++
- A set of VTK wrappers for TTK's algorithms
  - implemented in C++
  - with optional Python wrappers
- A plugin for the ParaView visualization system
  - provides TTK algorithms in ParaView's GUI
  - user-friendly access to topological data analysis

# Using the integrated ParaView plugin

Easiest installation option:

- Since version 5.10, ParaView has contained TTK as a built-in plugin.
- Not enabled by default!
  - Select Tools/Manage Plugins in menu
  - Expand the on Topology Toolkit section
  - Click Auto Load and Load Selected

Not always the latest released TTK version (e.g. ParaView 5.11.2 contains TTK 1.1)

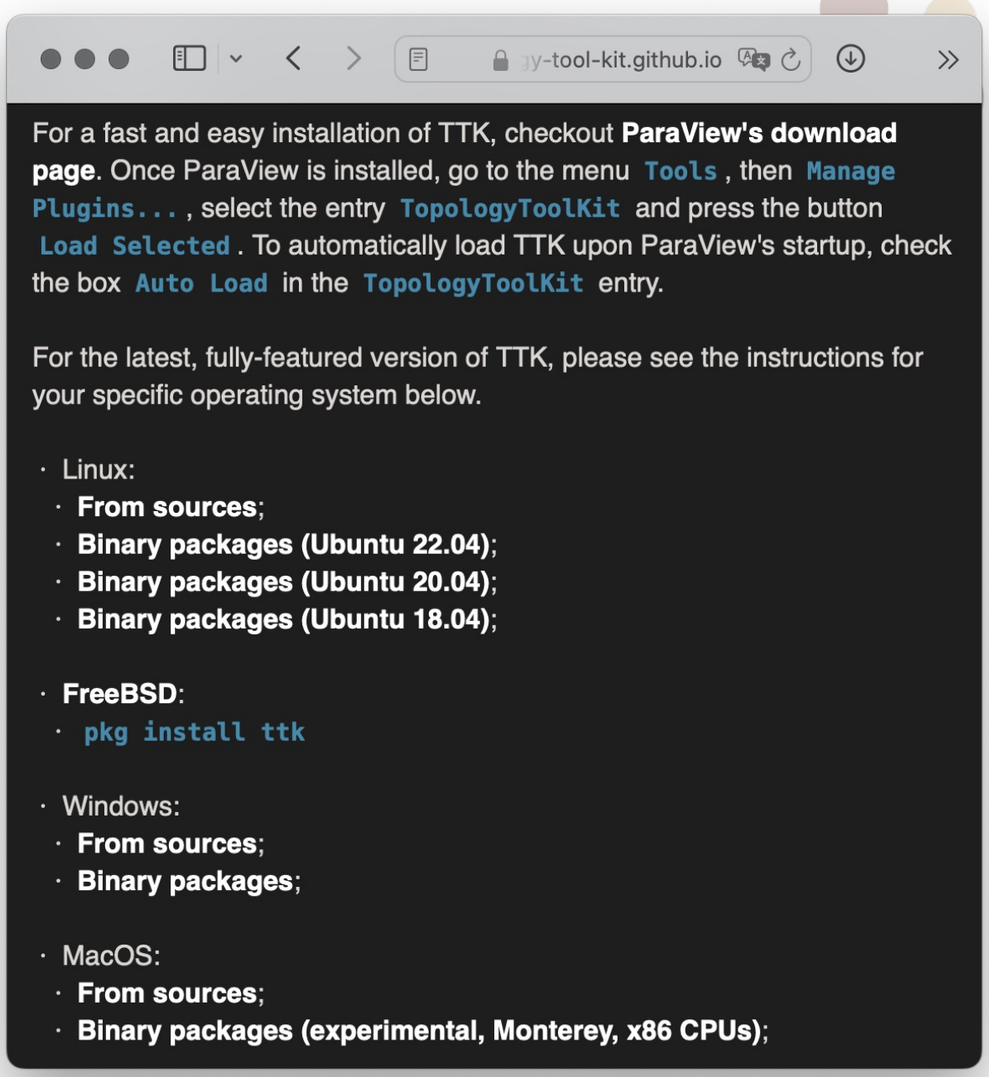


# Using a pre-built package

There are various binary packages for...

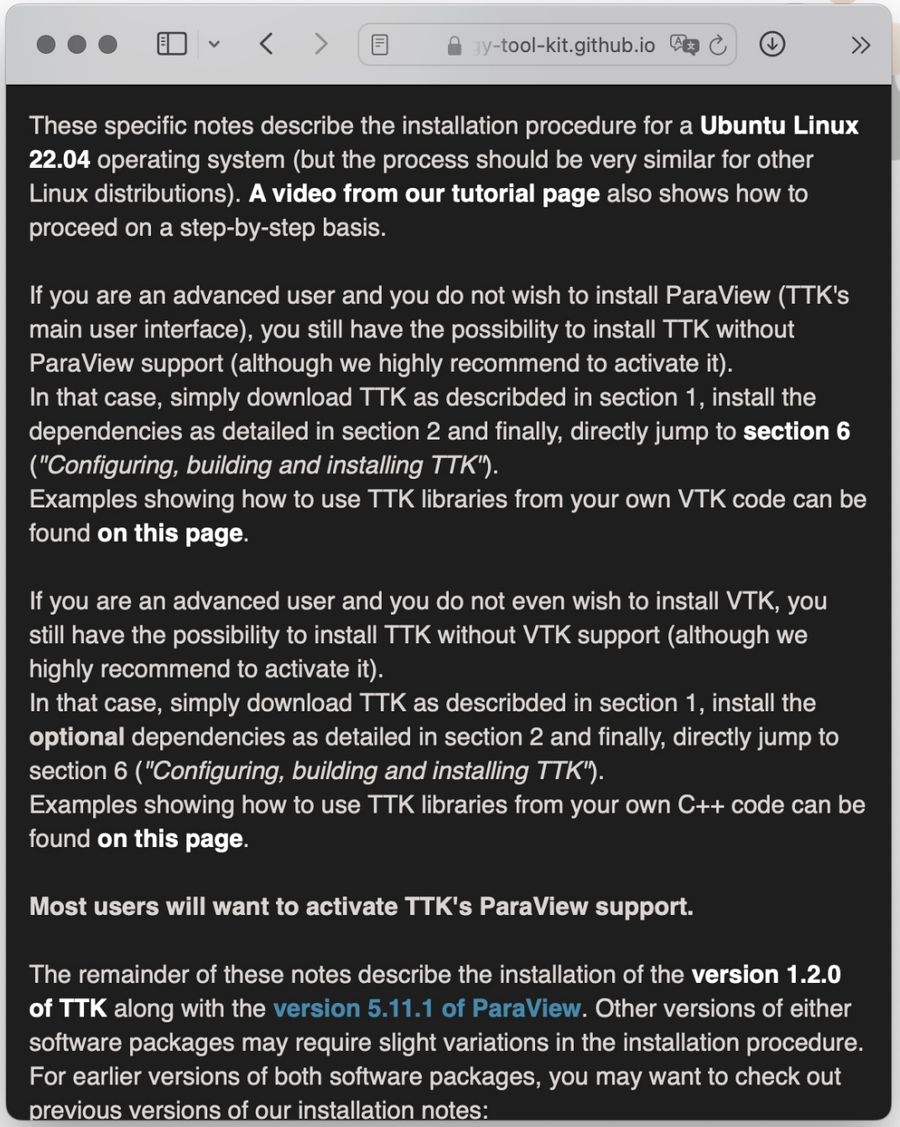
- Paraview with latest TTK (1.2.0)
  - x86 / x64: Linux, Mac, Windows
- TTK library (without ParaView)
  - x86 / x64: Linux, Mac, Windows
  - with Python and VTK support
- Anaconda
  - with / without ParaView
  - only x86 for the moment

Overview: go to TTK homepage → Installation



# Compile from source

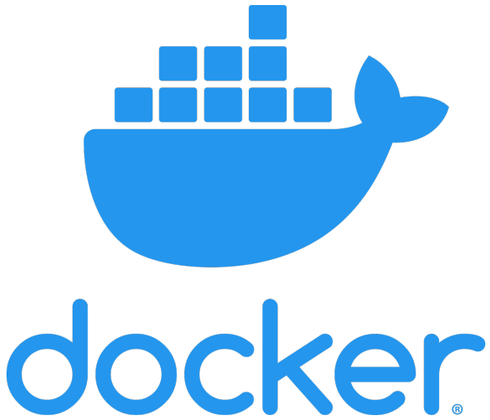
- Instructions for Ubuntu 22.04:
  TTK homepage → Installation
- Requires good comfort with
  command line, build tools, ...
- Dependencies must be installed, esp:
  - To compile the ParaView plugin,
    you will also have to compile ParaView
- Advanced option,
  high complexity



# What is Docker?

Docker is a tool designed to make it easier to create, deploy, and run applications by using containers. Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies, and ship it all out as one package.

– from opensource.com



# Why use Docker?

- Software dependency problems are a constant challenge encountered in scientific workflows
  - conflicting dependencies for installed binaries
  - frequent recompilation needed
  - platform-specific problems
- Docker containers allow packaging of TTK with all dependencies
  - Use TTK from container instead of directly from native install.
  - No installation, no dependency problems, no recompiles.

# Docker Essential Terminology

Size:
1GB (Docker image) vs 14GB (VM)

Startup time:
< 1s (Docker image) vs 20s (VM)

- A Docker container is a lightweight, encapsulated environment
  - almost completely isolated from the hosting operating system; similar to a "thin" virtual machine (no hardware emulation)
  - executed on the host system with OS support (Linux) or in a Linux VM (Windows, macOS)
- A Docker image represents a snapshot of a Docker container
  - Running a container initializes it from an image.
- The Docker engine transparently takes care of executing containers.

# Getting Docker

Docker is not open source, but the Desktop Edition is free for personal use.

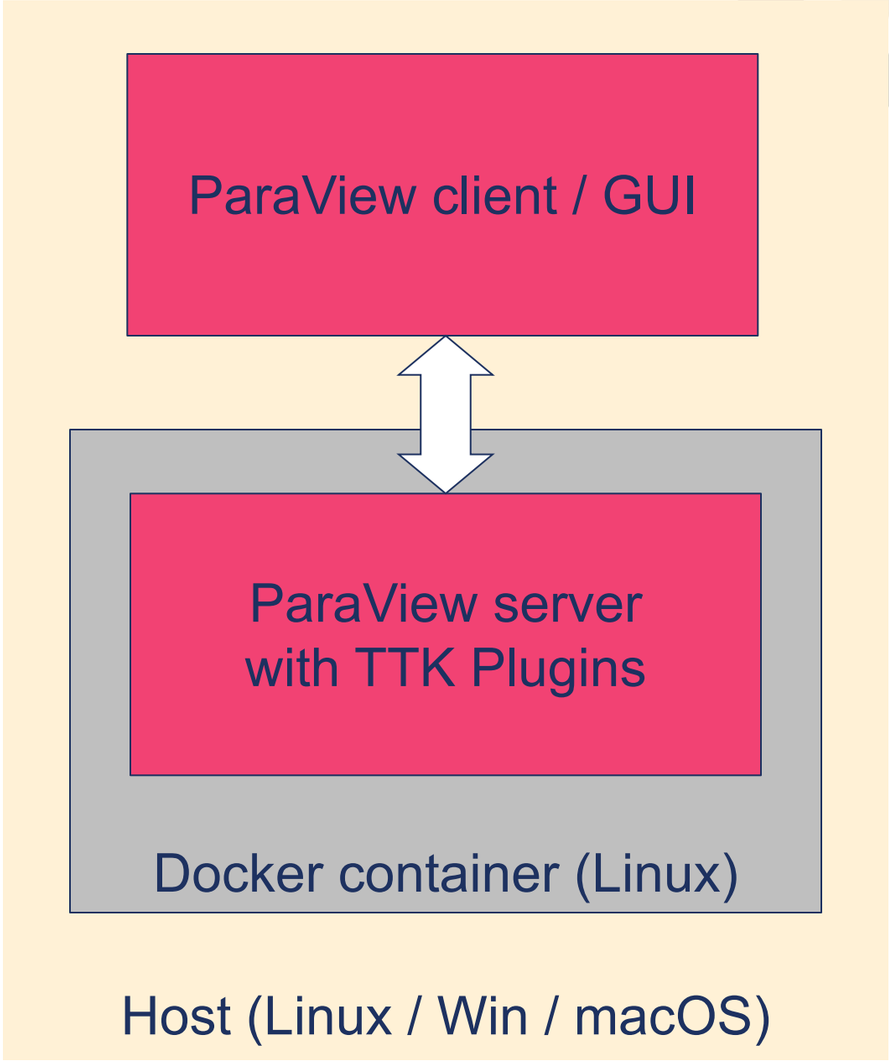
- Requires superuser / administrator privileges on the host system.

Installation instructions for Docker Desktop Edition:

| Linux | Windows | macOS |
| --- | --- | --- |
| Ubuntu, Debian, CentOS, Fedora, other Linux | Windows 10: Docker Desktop for Windows | 10.12 and later: Docker Desktop for Mac |
| QoL improvement: run containers without sudo | Older Windows: Docker Toolbox for Windows | Older: Docker Toolbox for Mac |

# Setup

- TTK+ParaView Docker containers utilize ParaView's built-in client/server mode
  - Server (including TTK plugins) runs in container.
  - Default ParaView client (GUI) runs in host OS.
  - Communication via host↔container networking.
  - No compilation needed at all.
- Caveats
  - Client / container versions must match exactly
  - Only software rendering and OSPRay supported; no hardware acceleration possible



# Running the TTK+Paraview Docker Image

Assume ParaView client 5.11.2 installed; want TTK version 1.2.0

Enter in terminal:

```
docker run -it --rm -p 11111:11111 -v ${HOME}:${HOME} -u ${UID} ghcr.io/scivislab/ttk:1.2.0
```

Remove container after exit

Same user in container as on host.

Image to run.

Run interactively (allow Ctrl-C).

Allow container to receive network connections on port 11111

Map user home directory to same path in container.

# Which Paraview version is used in the image?

Query using pvserver executable contained in image.

```
> docker run ghcr.io/scivislab/ttk:1.2.0 pvserver --version
paraview version 5.11.2
```

# Notes

- File paths (data or other) must made available to container.
  - Host filesystem is not visible to container by default.
  - Must explicitly pass “-v“ flags for needed directories.
  - Convenience script will automatically do this for home directory.

- Windows + macOS: container inside virtual machine
  - Docker Desktop Preferences:
    set up paths in “Shared Files“ or „File Sharing“

# Notes II

- Container can also run on other host
  - Just like ParaView server without container
  - Also with Singularity or Shifter (HPC systems)

- Build your own container
  - All scripts in <TTK>/scripts/docker
  - E.g.

```
cd <TTK>/scripts/docker

docker build -t my-ttk \
     --build-arg paraview=5.10.1 \
     --build-arg ttk=1.1.0
```

# Develop [on | using] TTK in 60 seconds

Visual Studio Code + Development Containers

- clone TTK repository from Github (or your own fork etc.)
- download development container description file (.devcontainer.json) and place it in TTK source

```
> curl -L bit.ly/dev-ttk-now > .devcontainer.json
```

- Open TTK source in Visual Studio Code
- Click „Reload in Container“

- ... profit!

# Conclusion

There are many ways to install TTK on your system.

Recommendations:

- If you are an end user: use the builtin ParaView plugin
- If you are a developer: check out the Docker images[1]

Contact: Ask us here or at `ttk-users@googlegroups.com`

[1]documentation update overdue