Exploiting napari for big data exploration

Vanessa Dao, Dave Barry
vanessa.dao@crick.ac.uk | david.barry@crick.ac.uk
CALM, The Francis Crick Institute, London, UK

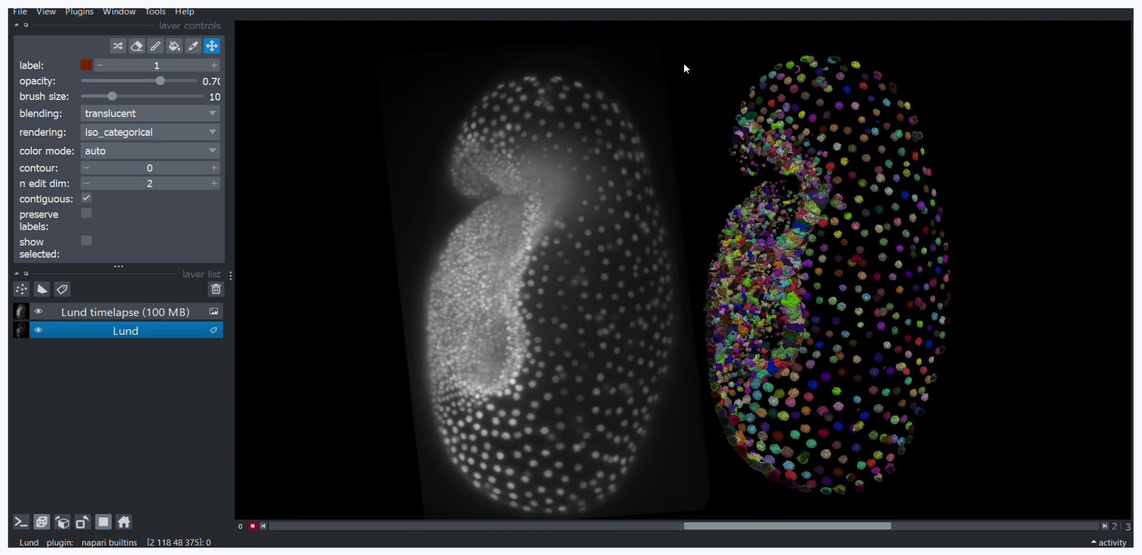
One of the most pressing issues facing the bioimaging community is the rise in data size, with biologists routinely acquiring large image volumes, while lacking an easy-to-use, open-source, freely-available multi-dimensional viewer for big data. Finding the best software and file format will aid in bridging the gap between biologists and image analysts, streamlining workflows with efficient reading of big data and user-friendly interaction with the data. Commercial software such as Imaris and Arivis address the issue, but are unable to support multiple OS platforms and are not freely available to the microscopy community. Open-source solutions, despite being more accessible, typically have inferior features such as the lack of 3D visualisation in BigDataViewer and limited support for next-generation file formats such as OME-Zarr in napari. We are developing a napari-based workflow prototype to optimise the process of big image visualisation with an emphasis on accessibility. By taking advantage of standard python libraries, open-source plugins and next generation file formats we are exploring and comparing different approaches to user-friendly big data interaction. This will ultimately support coordination between data acquisition and processing communities by providing a smooth and responsive means of viewing and interacting with big microscopy data in napari.

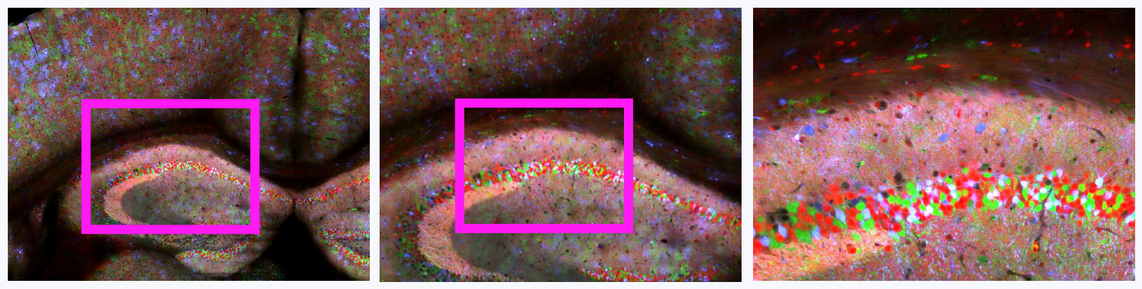
Open source viewers

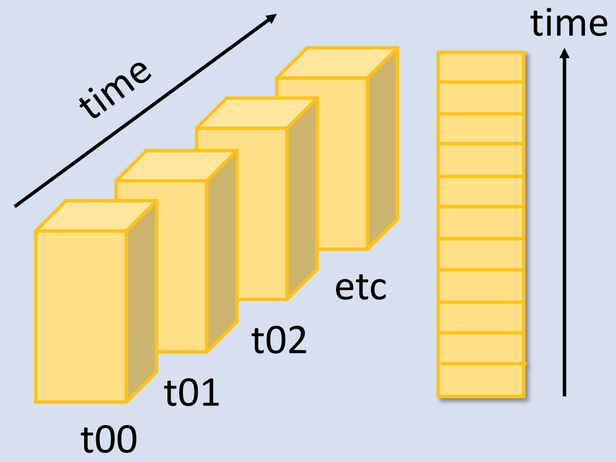
✖ No
● Yes
— Debateable

Viewer	Multi - dimensional	NGFF Support	Orthogonal View	Labels Overlay
ImarisViewer	✖	●	●	✖
BigVolumeViewer	✖	✖	✖	✖
3DViewer	✖	✖	●	✖
VVDViewer	●	✖	✖	●
Agave	—	●	●	✖
Vaa3D	●	✖	●	●
Drishti	—	✖	✖	✖
Napari	●	●	—	●

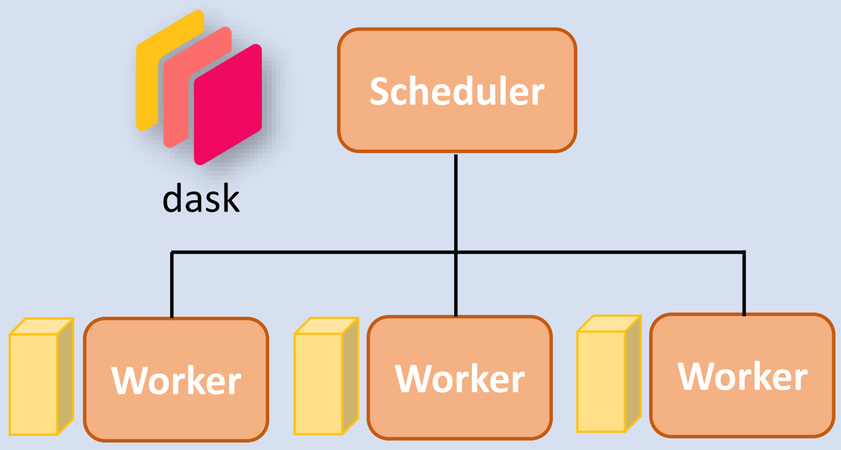
Napari



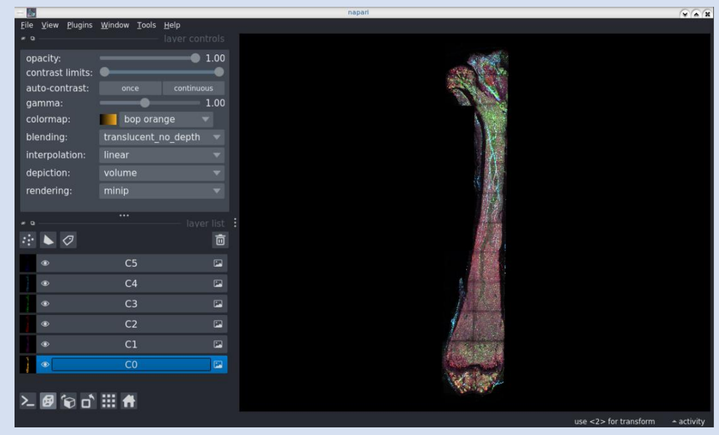




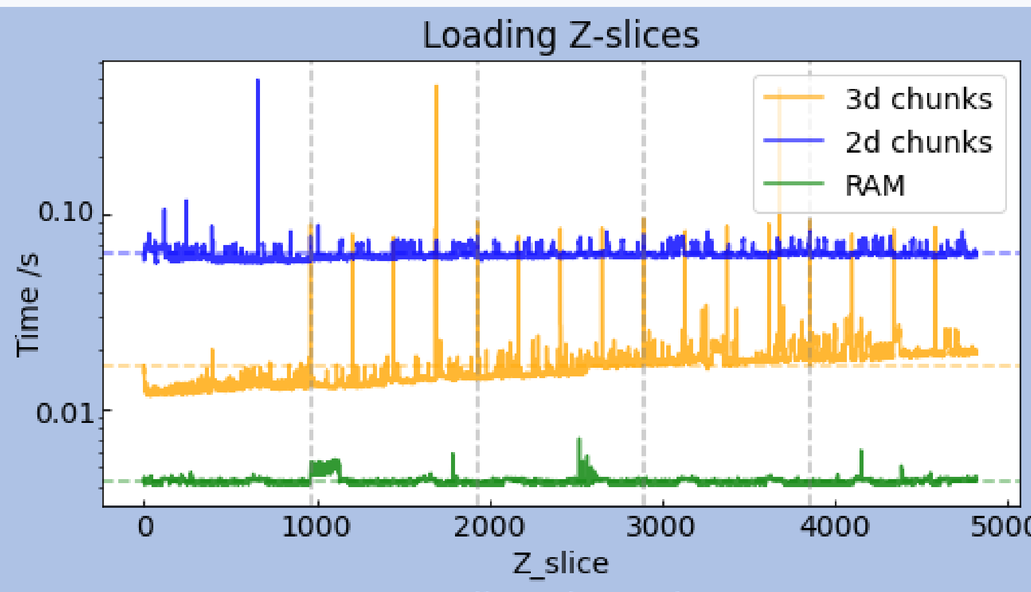
Multidimensional data organised into a dask stack then split into chunks

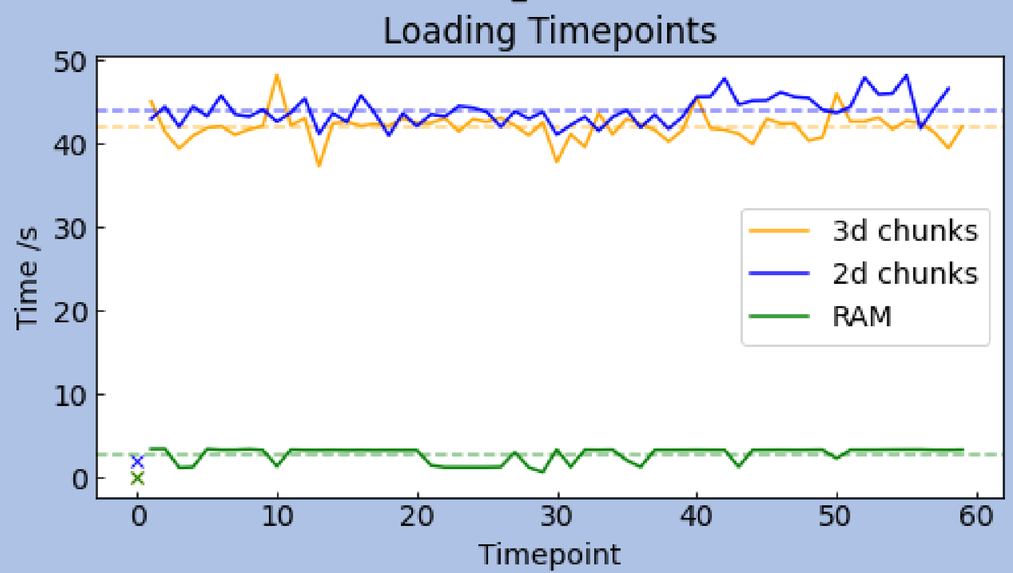


Dask scheduler manages data transfer between CPU workers and assigns chunks for computation



Data chunks are read into RAM as and when needed by the viewer



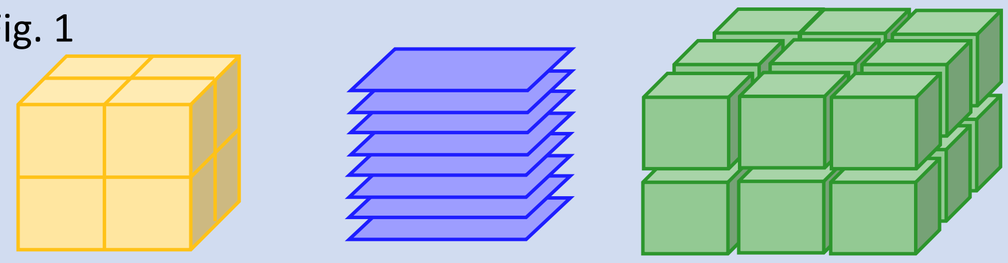


Quantifying interactivity within napari

We compare interactivity performance of dask with different chunk sizes against loading the following dataset into RAM within napari.

Fluo-N3DL-TRIF (60, 991, 1875, 965) *TZYX* ~ 180 GB
celltrackingchallenge.net/3d-datasets/

Fig. 1



3D chunks + dask	2D chunks + dask	Whole + RAM
(1, 247, 469, 241)	(1, 1, 1875, 965)	(60, 991, 1875, 965)
~ 165 seconds	~ 160 seconds	~ 40 minutes

The graphs display the time taken for napari to load the field of view upon changing z-slice and timepoint, respectively. The legend corresponds to Figure 1.

GitHub Repo

Survey