# EFFICIENT MICROSCOPY IMAGE VISUALIZATION AND CELL TRACKING ANALYSIS OF MULTI-GIGABYTE DATASETS.

**M. Jones[1], C. Birnbaum[1], B. Graff[1], S. McElroy[1]**, H. Lai[1], V.T. Chou[2], J.B. Long[2], M. Arnes[3], K. Obbad[2], S.V. Alworth[4], C.C. Huang[1], **L.A.G. Lucas**[1], D. Van Vactor[2], and J.S.J. Lee[1]

[1]DRVISION Technologies LLC, Bellevue, WA; [2]Department of Cell Biology and Program in Neuroscience, Harvard Medical School, Boston, MA; [3]Columbia University, New York, NY; [4]AcuraStem Inc., Los Angeles, CA

DRVISION Technologies LLC

**Introduction**

Characterizing the molecular and cellular migration behavior is critical for understanding the complex cellular and sub-cellular events that drive physiological and pathological states. In recent years, light sheet microscopy coupled with modern sample clearing techniques have enabled the visualization of multi-millimeter samples (e.g. a mouse brain) with high spatiotemporal resolution.

While there have been major advances in the development of imaging solutions that allow long term inspection of living samples, the efficient display and accurate analysis of multi-gigabyte datasets remain a major bottleneck. Traditionally, the workflow (visualization followed by detection and tracking) is accomplished using open-source tools (such as ImageJ/FIJI) with an off-the-shelf algorithm (such as Brownian motion) or a custom algorithm tailored to a specific application (such as plusTipTracker plug-in for Matlab). While custom algorithms can be very efficient for specific data sets, they can rarely be applied successfully on other types of data. Moreover, both the custom and the off-the-shelf algorithms available to the community are rarely deployed within a software package that offers superior performance when it comes to handling multi-gigabyte (per time point) 3D data sets.

Our objective with the present work is to introduce and discuss the merit of a general purpose 3D object tracking approach that is deployed within a fully supported software package that also is able to handle very large 3D data sets efficiently.

## Materials and Methods

| Test Data set | File size (GBs) | XYZ dimensions | Number of Voxels | Bit depth | Channels | Number of time points |
|---|---|---|---|---|---|---|
| A | 1 | 2048x2048x250 | 1.05E+09 | 8 | 1 | 1 |
| B | 1.4 | 1272x603x125 | 9.59E+07 | 16 | 1 | 7 |
| C | 2.6 | 2048x2048x632 | 2.65E+09 | 8 | 1 | 1 |
| D | 3.9 | 2060x2048x450 | 1.90E+09 | 16 | 1 | 1 |
| E | 6.1 | 2160x2560x594 | 3.28E+09 | 16 | 1 | 1 |
| F | 7.6 | 1024x1024x300 | 3.15E+08 | 16 | 1 | 11 |
| G | 11 | 1272x603x125 | 9.59E+07 | 16 | 1 | 50 |
| H | 16.1 | 2160x2560x1565 | 8.65E+09 | 16 | 1 | 1 |
| I | 17.8 | 2691x1809x1800 | 8.76E+09 | 8 | 2 | 1 |
| J | | 7835x3943x143 | 4.42E+09 | 16 | 4 | 1 |
| K | 239 | 9534x11869x721 | 8.16E+10 | 8 | 3 | 1 |

**Table 1. Description of test data sets used:** We sourced 11 test data sets covering a wide range of XYZ dimensions as well as typical number of channels and bit depth.

**Computer used for testing**: Razor Blade 2016, Intel® Core™ i7-6700HQ CPU @ 2.6 GHz 2.59 GHz, 16 GB of RAM, 64 bit Windows 10 Pro, NVIDIA GTX 1060 (6 GB of onboard memory), 1 TB SSD internal, 4 TB SSD external (USB 3.1).

**Aivia versions used**: Aivia 6.0 was used for file load and render benchmark tests. Aivia 6.1 beta 1 was used for all 3D object tracking tests.

**Rendering multi-gigabyte data sets**

As input we used individual tiff files containing either 3D (single time point) or 3D+time data sets. We have developed an algorithm that takes into account multiple key factors (e.g. available RAM, GPU memory, data set dimensions and characteristics, storage availability and speed) in order to determine the size and shape of each of the data blocks and resolution levels that will compose the high performance Aivia-tiff file. This algorithm can run offline (Aivia's batch file converter) or online when the file is opened directly in Aivia. Once the file of interest exists in the high-performance format it is rendered in real-time (see results below) using a novel rendering engine (which includes portions of VTK (1) and OpenGL (2)) and highly optimized memory management system.

The Aivia-tiff file contains a full resolution version of the original file as well as several down sampled versions. In addition, the data at each resolution level is stored in 3D blocks. Aivia decides, on the fly, which resolution level and which 3D blocks it needs to display to the user. As the user moves about the 3D or 4D data set Aivia quickly swaps blocks and resolution levels in order to always display the best possible representation of the data while using the lowest possible amount of hardware resources.

**Cell tracking (including cell detection)**

Input images where preprocessed to create a confidence image via bright spot enhancement, thresholding and softmatching (a supervised machine learning method for pixel classification). The local-maxima were detected on the output images and subsequently filtered based on their distance to other nearby local-maxima. This was followed by marker-based watershed to separate objects that could be touching. Finally, small or over segmented partitions were merged into neighbors to fit into the user defined object size range. To track 3D objects motion and morphology scores were determined. The motion score is based on how well the location of the new object conforms to the trajectory it had in previous timepoints. The morphology score is based on the similarity in intensity and volume between the last object in the trajectory and the new object. The best trajectory-object pairs were selected through matchmaking using the Hungarian algorithm (3) or a greedy match.

## Results and Discussion

### Rendering multi-gigabyte data sets

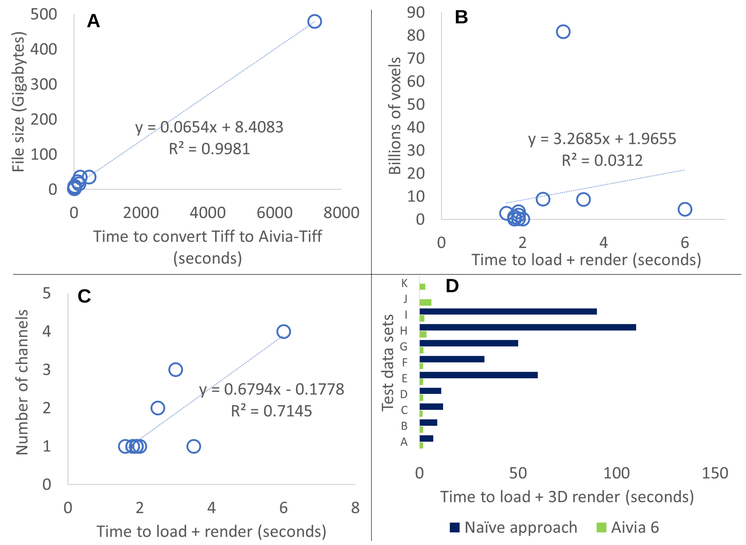


**Fig 1. Load and render speed benchmark results.** A) Off line conversion of original tiff files to Aivia-tiff could be achieved at a rate of 5.7 ± 2.7 GB/minute. We did not see any evidence for this rate being different (i.e. lower) for very large files (e.g. 470 GBs). B) file size (in GB or voxels) does not significantly affect the time taken to load and render images saved as Aivia-tiff. Most files completed the test in 2 or less seconds and the average time was 2.5 ± 1.2 seconds. C) However, the number of channels in an image does seem to affect loading and rendering speeds. Each channel took an average of 1.8 ± 0.6 seconds. As a consequence data set J (with 4 channels) took the longest to complete the test (6 seconds) even though it was not the biggest file. D) Real time loading and rendering of multi-gigabyte Aivia-tiff files. The blue bars represent the time it took to load and render each of the test data sets using the original tiff files. Each test data set was loaded from a single tiff file containing a 3D or 3D+time information. The green bars represent the time needed to load and render the test images when using the Aivia-tiff file.

### Cell Tracking

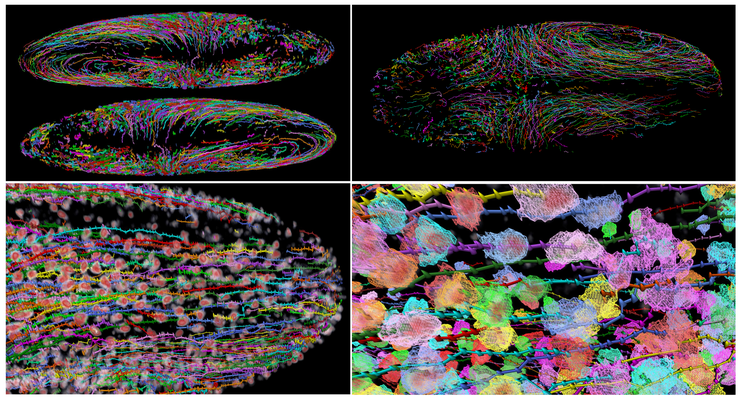

**Fig 2. Cell detection and tracking of multi-gigabyte images.** The benchmark tests (using data set G, table 1) conducted show that each frame took approximately 11 seconds to be fully processed. This includes both the 3D object detection, mesh generation, 3D object tracking and the generation of measurements. Thus the 50 frames of the test data set took nearly 550 seconds (just over 9 minutes) to be completed. In time point 1, 3341 individual cells were detected and 384 could be tracked for at least 40 frames. In total we detected and tracked 15363 cells, with more than 50% of the tracks lasting at least 19 frames. Note that with our current implementation if a cell divides two new tracks are created, i.e. lineages are not tracked. Early tests with an extended version of the presented algorithm indicate we can also use it to track lineages – this will be added in future versions of Aivia. Thus, it is likely that we could not track the same cells in all 50 frames because some underwent mitosis. A total of 384 cells were successfully tracked for an average 40 ± 4 frames.

## Conclusion

The present work introduces a high performance solution for the display and interaction with multi-gigabyte multi-dimensional microscopy data sets in real time. We show that we can load, display and interact with files of hundreds of gigabytes in 3.7 second or less using a standard gaming laptop with a single GPU. Moreover, we discussed a novel 3D object tracking approach which can process an 11 GB 4D data set (50 time points) in less than 10 minutes. Jointly, these two solutions enable the study of dynamic cellular and sub-cellular processes linked to a variety of normal and pathological states (e.g. metastasis formation, immune responses, protein trafficking, stem cell differentiation). Our next step will be to apply this approach to the tracking of EB1 plus-TIPs in live Xenopus neurons and then expand our tracking model to detect cell lineages thus enabling a number of studies in developmental biology as well as cancer research and stem cell biology.

## References

1) Shreiner, Dave; Sellers, Graham; et al. (30 March 2013). OpenGL Programming Guide: The Official Guide to Learning OpenGL. Version 4.3 (8th ed.). Addison-Wesley. ISBN 978-0-321-77303-6.
2) Schroeder, Will; Martin, Ken; Lorensen, Bill (2006), The Visualization Toolkit (4th ed.), Kitware, ISBN 978-1-930934-19-1
3) Kuhn, H.W. 1955. The Hungarian method for the assignment problem. Naval Research Logistics Quarterly, 2:83–87.

## Acknowledgements

Research is supported by NIH Grant number 5R43MH100780-02