Journal of
open research software

SOFTWARE METAPAPER

Particle Data Management Software for 3DParticle Tracking Velocimetry and Related Applications – The Flowtracks Package

Yosef Meller[1] and Alex Liberzon[2]

[1] Ph.D. candidate, School of Mechanical Engineering, Faculty of Engineering, Tel Aviv University, IL

[2] Head of Laboratory, School of Mechanical Engineering, Faculty of Engineering, Tel Aviv University, IL

Corresponding author: Yosef Meller (yosefmel@post.tau.ac.il)

The Particle Tracking Velocimetry (PTV) community employs several formats of particle information such as position and velocity as function of time, i.e. trajectory data, as a result of diverging needs unmet by existing formats, and a number of different, mostly home-grown, codes for handling the data. Flowtracks is a Python package that provides a single code base for accessing different formats as a database, i.e. storing data and programmatically manipulating them using format-agnostic data structures. Furthermore, it offers an HDF5-based format that is fast and extensible, obviating the need for other formats. The package may be obtained from https://github.com/OpenPTV/postptv and used as-is by many fluid-dynamics labs, or with minor extensions adhering to a common interface, by researchers from other fields, such as biology and population tracking.

Keywords: 3D-PTV; hdf5; trajectory database; particle tracking; velocimetry

(1) Overview

Introduction

The three-dimensional Particle Tracking Velocimetry (3D-PTV) is a well established method of experimental fluid dynamics research, rapidly growing in interest due to technological advances on 3D imaging and increasing accessible computing resources. The method in its basic form uses volumetric photography techniques to obtain 3D positions of particles suspended in a flow from synchronised video streams created by several cameras (2–4 are the common cases) [1]. The 3D position data is obtained by a combination of standard image processing techniques and domain specific algorithms for finding inter-image correspondences, trajectory identification and calibration. These techniques are implemented in a number of research software packages, most are custom-made in every laboratory doing 3D-PTV research [2,3,4,5]. There is a large group of users in the consortium of several research institutions (organized by TU Eindhoven, Tel Aviv University, ETH Zurich) that jointly develops the OpenPTV software library, which is a derivative of the 30 year old *3D-PTV* code from ETH Zurich [2,3,4].

Each of these software packages generates some form of trajectory data – a collection of 3D positions (and other properties) of particles in space, conjoined with a position in the time series (termed *frame* as it relates to one of the camera frames) and time-connectivity information (termed *trajectory* physically related to Lagrangian trajectory, describing the object movement in space and time). This data set, in order to be useful for post analysis, requires some transformations. Example methods are smoothing, vetting trajectories and controlling their quality; some methods are research-specific, such as computing correlation functions, various time and space derivatives of kinematic significance, or dynamic properties.

The generated set of related data, therefore, should be organized for rapid search and retrieval, which is to say it should be placed in a *database*. A database, as defined by the Merriam-Webster dictionary, is "a usually large collection of data organized especially for rapid search and retrieval". However, existing tools neglect the organization for search and retrieval, and only focus on storage.

The field thus faces two problems. The first is that tools for these post-processing tasks have so far been home-grown, with little sharing or review. The usual consequences of such a state are evident in the field: little emphasis on quality control as the software is discarded after publication, duplicated effort and little progress on technical issues such as processing speed and efficient algorithms and data structures.

The second problem is the existence of several formats for particle data, mostly duplicating the same functionality, and eschewing best-practices of research software and accumulated features offered by modern formats. This is discussed further in section **Implementation and architecture**. Additionally, in the present form of multiple

ASCII table-type files per each frame, a typical problem is that there are slight differences between series of text files produced by the 3D-PTV code and that of some post-processing tool, but the lack of content metadata and unifying structure makes sharing code among the formats a somewhat involved task.

Many database formats and database management programs exist today, such as MySQL, PostgreSQL, and the recently popular MongoDB. They are, however, designed for general structured business data or unstructured network data, and poorly suit the specific needs of storing and querying scientific data generally, and trajectory data in particular.

The Flowtracks package described herein intends to provide the particle tracking community with a flexible, efficient and modern database management solution tailored to its special needs. It incorporates a data format based on the HDF5 format, a binary format developed and extensively used for scientific data, with a modular framework for querying the database in the patterns typical to the field, and an extensive I/O library for working with legacy formats and converting them to other database formats.

The package started as an in-house effort at another post-processing code. It has been used extensively for research on particle behavior in turbulence [6]. However, due to our work with several collaborators' data sets, the need arose to understand several of the different formats. This resulted in an extensive input and output library. After some discussion with other scientists who have tried the code, its general appeal became clear and resulted in this submission.

Implementation and architecture

The program is written in pure Python [7], although it relies on some widely available extension modules: SciPy [8] provides array input, output and manipulation, and an interpolation module; PyTables [9] provides raw access to HDF5-formatted files, with some structure added by PyTables.

The underlying Flowtracks format provides advantages that will hopefully lead to its useful implementation and to encourage its adoption in the community. Unlike other dataset formats, it was designed to underly database management code, and facilitate fast querying and retrieval. Therefore, it is a binary format, allowing much reduced data size and much increased processing speed; it is based on the HDF5 standard, thus opening up a range of existing libraries for handling it. Usage of plain HDF5 files is common both in the scientific community in general, and in the particle tracking community. Mostly, one encounters unwitting and ad-hoc use of the format through Matlab files (which are based on the format), but also seen in some publicly available code in the particle tracking field, such as the TrackPy package [15].

HDF5 files are preferred over the use, typical to the field, of sets of ASCII table files. Advantages include efficient use of storage space, fast access through memory mapping, and openness. HDF5 is a binary format, which usually requires less bytes per floating point number and far less space for data format information compared to text files. Fast access results from the reduced storage size, but also from memory-mapping of parts of the on-disk data. Memory-mapping is a facility provided by modern operating systems, in which on-disk data is copied to memory quickly and held there only when needed, then transparently dropped from memory. This facility is not directly usable by sets of ASCII text files. Finally, the format is served by a free-software library, open and unencumbered by usage restrictions and license fees.

Flowtracks adds field-specific structure to the basic HDF5 format, while staying within the format's definitions. That is, all Flowtracks databases are still HDF5 files. The package ensures that required fields are present, and ensures size constraints on fields added by the user. It maintains indices by trajectory ID and by frame. A set of classes representing basic concepts in particle tracking are provided as the user's handle on the database.

The modules provided by the package belong to three categories: basic data structures shared by all modules; general facilities agnostic of data format; and scene manipulation and analysis classes, which differ between the HDF5-based format and the legacy formats.

The basic data structures hierarchy is outlined in **Figure 1**. It begins with the ParticleSet class, which represents a general slice of the particle database. The class holds a varying number of particle properties, from the mandatory 2 (position and velocity) and up, and ensures a common length to all property arrays. Adding a new property is supported and creates the necessary attribute query and modification methods from it automatically. In this way, the format allows expanding the set of properties represented in a file, without having to invent a new format or changing the toolset for handling the data. In many usage scenarios at least one more field, for particle acceleration, will be added by the user. Other fields that have been added in practice refer to fluid properties near the particle, as well as per-particle quality indicators.

ParticleSet has two derivatives, Trajectory and ParticleSnapshot. They both add the time and trajectory ID (a number unique across the scene's trajectories) properties to the mandatory set, but treat them differently: for Trajectory, the ID is a scalar, unchanging across particles, while time increases from particle to particle; for ParticleSnapshot, time is scalar and the trajectory ID changes.

Scene manipulation for the HDF5-based format can be done using the *Scene* class, or for scenes involving two types of particles (e.g. tracers and inertial particles), the *DualScene* class. The *AnalysedScene* class offers simultaneous management and querying of the *DualScene* and the analysis results file, also in HDF5-based format. *AnalysedScene* offers the most powerful querying mechanism in the package – the *collect()* method which transparently handles collection of data based on flexible user-defined filters, without the user having to explicitly iterate over frames or trajectories. The analysis of HDF5-based formatted data may be performed using the *analysis* function, which applies a set of user-defined analyser classes on the data; the interface for these classes is defined in the package documentation.

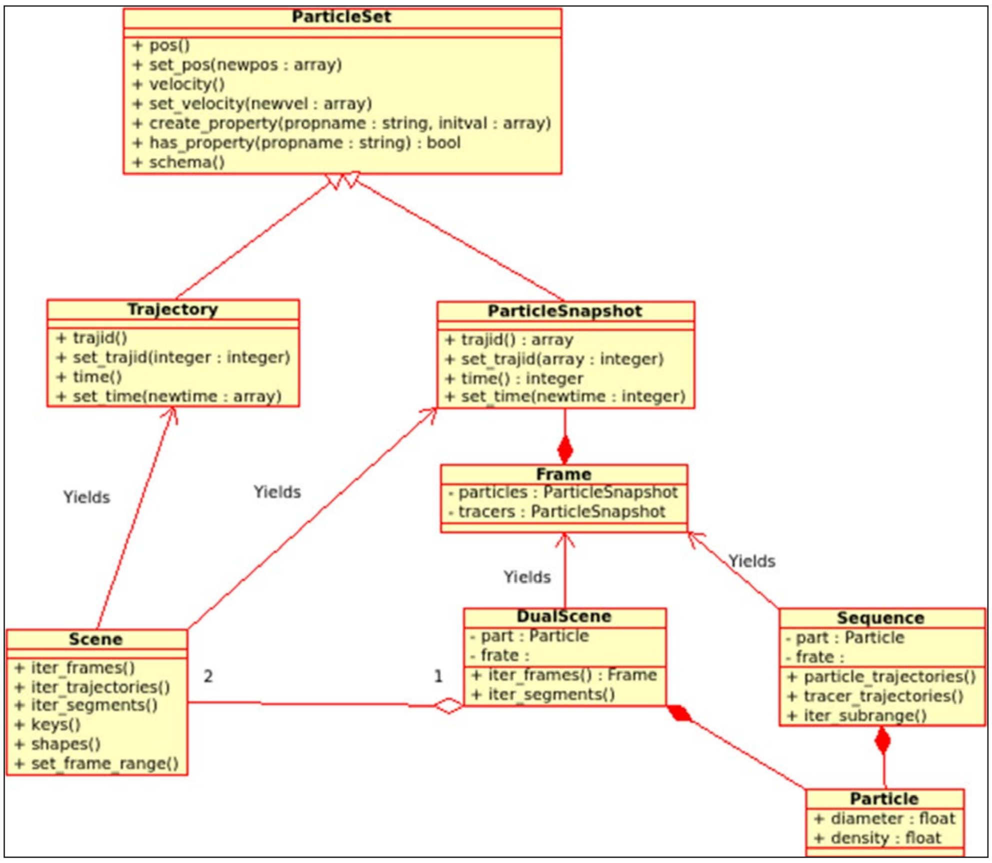


Figure 1: Class diagram. Only key operations are shown – more are available, as detailed by the package's documentation.

The parallel for legacy formats is the *Sequence* class, serving as both a dual-scene manager and analyser. The particle database itself is accessed using the *io* module, with the above-mentioned limitations on memory size and speed. Methods of this class provide iteration over the data and querying the data and analysis results.

The general facilities offered by Flowtracks include first of all the *io* module, which handles reading and/or writing of the different formats the package understands. Although the package provides higher-level interfaces to the data through the *Scene, DualScene* and *Sequence* classes, this package may be used for a more direct and fine-grained access to the different storage formats that are recognized. It provides the data read from databases as lists or Python iterators – the last option allows in many cases to reduce the memory resident data size, but only to varying amounts depending on the format. Other facilities include: an interpolation module that allows repeated interpolation at selected points (the selection of interpolation methods is detailed in the module documentation), caching intermediate results for speed; generation of probability density functions from the data and plotting them in the visual style of the fluid-dynamics community; and calculations on particles such as trajectory smoothing and nearest-neighbour searches.

Example workflow

In the common case, a user would start by generating trajectory data in a particle tracking software. Although Flowtracks understands a few formats produced by these tools, post processing would be easier after converting the dataset to the Flowtracks database format. This may be as simple as two lines of code. For example, a series of 100 text files, one per frame, numbered from 1 to 100, and representing one second of capture, is converted using the following Python code:

```
from flowtracks.io import trajectories,
    save_particles_table
trajects = trajectories("frame%d", 1, 100,
    frate=100, iter_allowed=True)
save_particles_table("trajects.h5", trajects)
```

However, at this point the user may want to apply one-time smoothing or filtering per trajectory before saving, using the facilities provided by Flowtracks or custom code.

Given the database in the Flowtracks format, the data can now be efficiently iterated using the *flowtracks.analysis* module or any of the *Scene* or *DualScene* classes as needed.

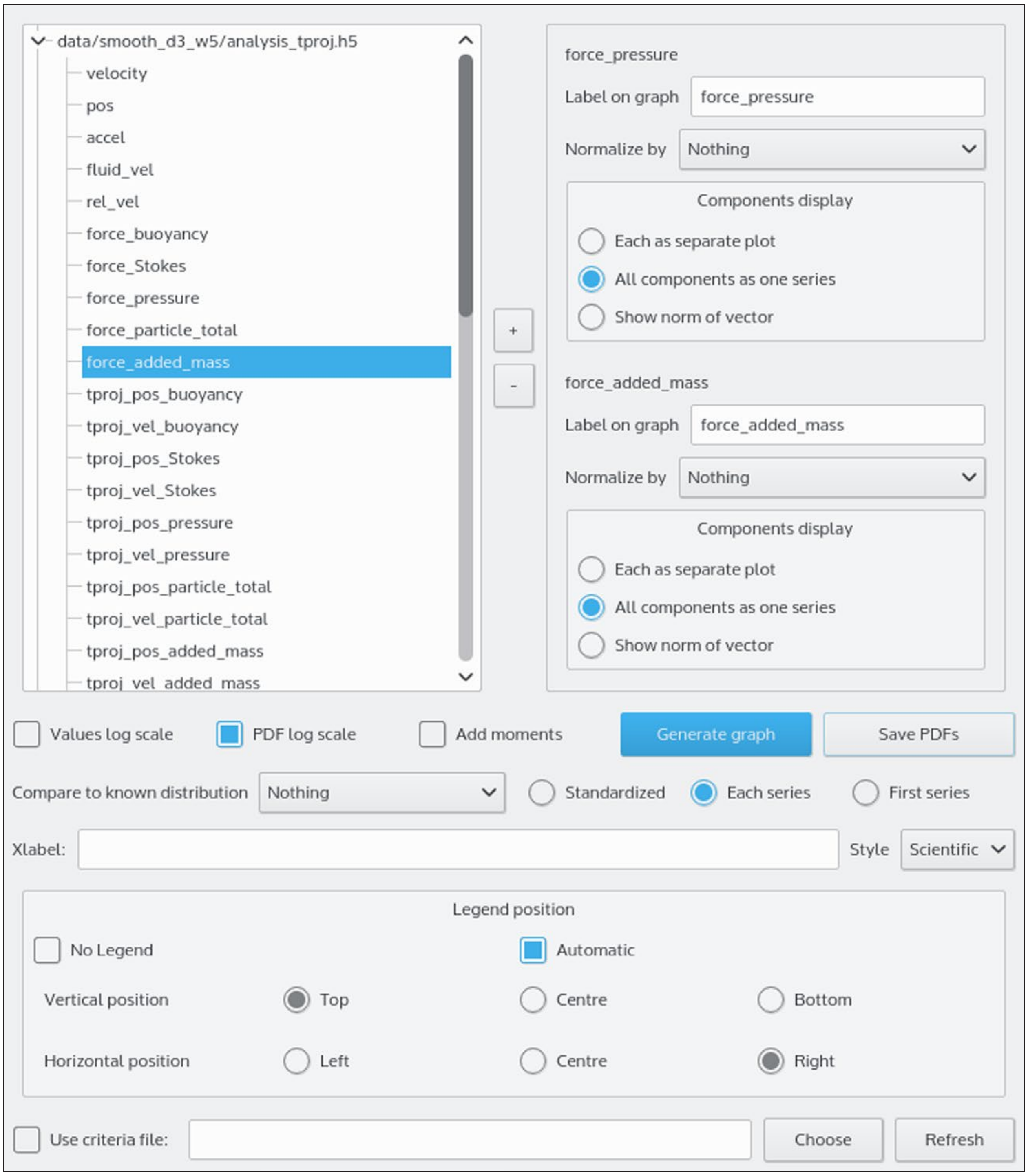
After producing several analyses, each exploring different aspects of the data, the introspection properties offered by the database code become useful. The researcher needs not expand effort keeping track of the different analyses found in each file, as the database schema is reported by Flowtracks on request. In **Figure 2** we see a program listing, on the left side of the interface, all analysis fields available in an analysed database. The figure also demonstrates the possibilities offered by using Flowtracks in conjunction with the wide range of free software available in the Python language for building user interfaces and performing other useful functions not directly related to the science.

With the information on the database schema, it is now possible to query certain fields or parts thereof. A common operation, querying an analysis key and generating a Probability Density Function of the result (example is shown in **Figure 3**) are generated and displayed using the *flowtracks.graphics* module.

In conclusion, the Flowtracks package offers a set of classes and methods for managing and analysing a particle database in the new format, and a set of tools for reading other formats and converting to the new format. It provides a bridge from legacy formats while leveraging more efficient options available to the research software community. It affords flexibility without compromising efficiency, and eases reuse of one dataset by several scientists.

Quality control

The program was subjected to extensive functional testing while it was used for turbulence research in our laboratory. Additionally, a set of unit tests is available, but the current coverage is still limited to major functionality. Our usage of the software was in a Linux



data/smooth_d3_w5/analysis_tproj.h5
- velocity
- pos
- accel
- fluid_vel
- rel_vel
- force_buoyancy
- force_Stokes
- force_pressure
- force_particle_total
- force_added_mass
- tproj_pos_buoyancy
- tproj_vel_buoyancy
- tproj_pos_Stokes
- tproj_vel_Stokes
- tproj_pos_pressure
- tproj_vel_pressure
- tproj_pos_particle_total
- tproj_vel_particle_total
- tproj_pos_added_mass
- tproj_vel_added_mass

force_pressure
Label on graph: force_pressure
Normalize by: Nothing
Components display: Each as separate plot [✓ All components as one series] Show norm of vector

force_added_mass
Label on graph: force_added_mass
Normalize by: Nothing
Components display: Each as separate plot [✓ All components as one series] Show norm of vector

Values log scale [✓ PDF log scale] Add moments

Compare to known distribution: Nothing — Standardized [✓ Each series] First series

Xlabel:
Style: Scientific

Legend position
No Legend [✓ Automatic]
Vertical position: [✓ Top] Centre Bottom
Horizontal position: Left Centre [✓ Right]

Use criteria file:

Figure 2: In-house software developed using Flowtracks' format and analysis tools. Database schema introspection provided by Flowtracks is used for building the list of available analyses on the lefthand side.

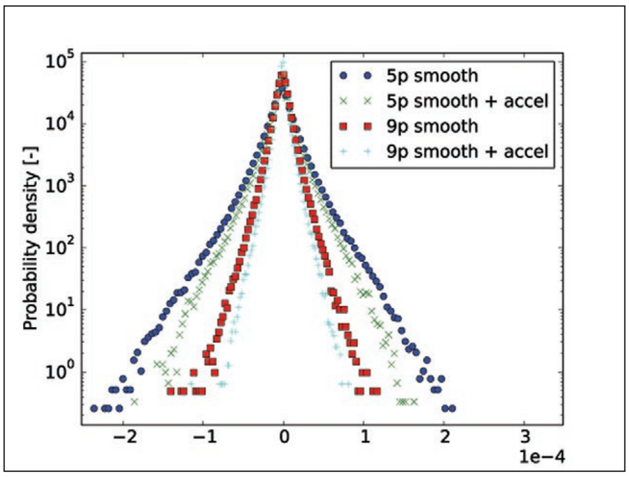


Figure 3: Probability Density Functions created using the flowtracks.graphics module. Each series is queried, binned and graphed using Flowtracks, shielding the researcher from having to handle the database manually.

environment (Kubuntu, several latest releases) and MacOS X (10.1).

(2) Availability

Operating system

The software should be able to run on any system that can run the necessary version of Python and has the dependency modules available. It was tested on Linux (Kubuntu) and MacOS X but should also work on Windows and other systems.

Programming language

Python of the 2.x series is needed. The library requires version 2.6 or higher, but the tests and building the documentation require features of version 2.7

Additional system requirements

N/A

Dependencies

- Scipy (>= 0.13)
- PyTables 3.x
- Sphinx – optional, for building documentation in HTML or other formats.
- Matplotlib – optional, for generating graphs.

List of contributors

N/A

Software location

Archive (e.g. institutional repository, general repository) (required)

Name: Figshare

Persistent identifier: DOI: http://dx.doi.org/10.6084/m9.figshare.1572986

Licence: GNU General Public License V3

Publisher: Yosef Meller

Date published: 13/10/15

Code repository (e.g. SourceForge, GitHub etc.) (optional)

Name: GitHub

Identifier: https://github.com/OpenPTV/postptv

Licence: GNU General Public License V3

Date published: 13/10/15

Emulation environment

N/A

Language

Package: Python; documentation: Sphinx reStruturedText; some examples provided as Jupyter notebooks.

(3) Reuse potential

The software is built as a general tool for the kind of data encountered in our field, so that it is possible to replace home-grown solutions with this package. Furthermore, any research which plans to rely on the calculation of new particle properties, may extend both the analysis code (by writing analyser classes) and the saved format (with no extra code).

Further reuse potential lies in the modularity of the package. Legacy formats may be added to the I/O library through a common interface, without changing the rest of the machinery. The data therein may then be directly analysed or converted to the Flowtracks format for easier access.

Outside the fluid dynamics field, trajectory data is employed in other fields such as human crowd tracking [11], biology applications [12,13,14], medicine and other fields. All may benefit from this package, which is agnostic of the source of the particle database.

Acknowledgements

This work has been partially supported by the Israeli Science Foundation.

Competing Interests

The authors declare that they have no competing interests.

References

1. **Dracos, Th** (ed.) 1996 Three dimensional Velocity and Vorticity Measuring and Image Analysis Techniques; Kluwer Academic Publishers. DOI: http://dx.doi.org/10.1007/978-94-015-8727-3
2. **Maas, H-G** 1992 Complexity analysis for the determination of image correspondences in dense spatial target fields. *International Archives of Photogrammetry and Remote Sensing*, Vol. 29, Part B5, pp. 102–107
3. **Willnef, J** 2003 A Spatio-Temporal Matching Algorithm for 3D Particle Tracking Velocimetry; Dissertation, Technische Wissenschaften ETH Zurich, Nr. 15276; Zurich, Switzerland.
4. **Lüthi, B** 2003 Some aspects of strain, vorticity, and material element dynamics as measured with 3D particle tracking velocimetry in a turbulent flow; Dissertation, Technische Wissenschaften ETH Zürich, Nr. 14893; Zurich, Switzerland.

5. **Ouellette, N T, xu, H T** and **Bodenschatz, E** 2006 A quantitative study of three-dimensional Lagrangian particle tracking algorithms. *Experiments in Fluids*, 40 (2): pp. 301–313. DOI: http://dx.doi.org/10.1007/s00348-005-0068-7
6. **Meller, Y** and **Liberzon, A** 2015 Particle–fluid interaction forces as the source of acceleration PDF invariance in particle size. *International Journal of Multiphase Flow*, 76 (11): pp. 22–31. DOI: http://dx.doi.org/10.1016/j.ijmultiphaseflow.2015.04.018
7. **Python.** Python Software Foundation. Available at www.python.org/.
8. **Jones, E, Oliphant, T** et al 2001 SciPy: Open source scientific Tools for Python. Available at http://scipy.org/.
9. **Alfred, F, Vilata, I** et al 2002 PyTables: Hierarchical Datasets in Python. Available at http://www.pytables.org/.
10. **Hunter, J D** et al 2015 Matplotlib. DOI: http://dx.doi.org/10.5281/zenodo.15423
11. **Corbetta, A** et al 2014 High statistics measurements of pedestrian dynamics. *Transportation Research Procedia,* 2: pp. 96–104. DOI: http://dx.doi.org/10.1016/j.trpro.2014.09.013
12. **Zilman, G** et al 2013 The hydrodynamics of contact of a marine larva, *Bugula neritina*, with a cylinder. *Journal of Experimental Biology* (216): pp. 2789–2797. DOI: http://dx.doi.org/10.1242/jeb.083352
13. **Taylor, Z J** et al 2014 Particle Image Velocimetry For Biological Mechanics; in Neu, C P and Genin, G M (ed.), *Handbook of Imaging in Biological Mechanics*, CRC Press. DOI: http://dx.doi.org/10.1201/b17566
14. **Adhikari, D** et al 2015 Simultaneous Measurement of 3D Zooplankton Trajectories and Surrounding Fluid Velocity Field in Complex Flows. *J Exp Biol* (218): pp. 3534–3540. DOI: http://dx.doi.org/10.1242/jeb.121707
15. **Allan, D B** et al 2014 TrackPy. Available at https://github.com/soft-matter/trackpy. DOI: http://dx.doi.org/10.5281/zenodo.9971

How to cite this article: Meller, Y and Liberzon, A 2016 Particle Data Management Software for 3DParticle Tracking Velocimetry and Related Applications – The Flowtracks Package. *Journal of Open Research Software,* 4: e23, DOI: http://dx.doi.org/10.5334/jors.101

Submitted: 13 October 2015 **Aceepted:** 24 May 2016 **Published:** 16 June 2016