MyPTV: A Python Package for 3D Particle Tracking

Ron Shnapp[1]

1 Swiss Federal Institute for Forest, Snow and Landscape Research WSL

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 14 April 2022
Published: unpublished

Summary

Three dimensional particle tracking velocimetry (3D-PTV) is a method that is widely used to study the dynamics of objects moving in space and to sample velocity fields at the location of tracer particles. Applications of 3D-PTV abound in various fields, such as, fluid mechanics, biology, animal behavior and crowd control (Arnèodo et al., 2008; Attanasi et al., 2015; Bagøien & Kiørboe, 2005; Brizzolara et al., 2021; China et al., 2017; Holzner et al., 2008; Lüthi et al., 2005; Mass et al., 1993; Michalec et al., 2017; Ott & Mann, 2000; Ouellette et al., 2006; Pouw et al., 2020; Shnapp et al., 2019; Sinhuber et al., 2019; Toschi & Bodenschatz, 2009; Virant & Dracos, 1997). A common methodology of 3D-PTV uses synchronized photography from several locations (e.g., by using several calibrated cameras, see Figure 1). From such camera images, the 3D positions of the particles can be estimated using photogrammetry methods. Then, particle locations are linked in time to generate 3D trajectories that can be analyzed. Furthermore, time differentiation yields measurements of the objects' velocity and acceleration, thus yielding the 3D particle tracking velocimetry method (3D-PTV) (Virant & Dracos, 1997). In this work, we present an open source, Python-based software package, dedicated to making 3D-PTV more accessible to the scientific community.

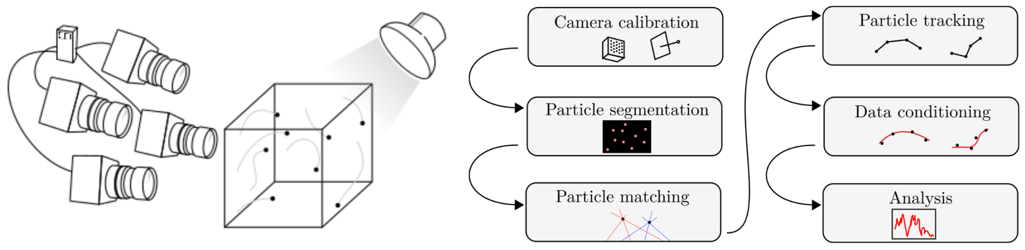

Figure 1: Left - A schematic sketch of a 3D-PTV experiment with a four-camera system. Right - the 6 steps of the post-processing and analysis of common 3D-PTV experiments.

Statement of need

The application of 3D-PTV relies heavily on computation during the several steps of post-processing the experimental results (see Figure 1). In particular, in many applications researchers study objects that appear in high density in the recorded images, and the images are recorded over extended periods of time at high rates, yielding high volumes of raw data (Shnapp et al., 2019). Thus, 3D-PTV experiments are inevitably post-processed using specialized computer codes that often need to be tailored to the specific characteristics of the system under investigation. Indeed, the effort the programming of a functioning 3D-PTV software requires might deter inexperienced researchers from employing 3D-PTV, and hinder further development of the method.

MyPTV is a Python package designed to make 3D-PTV accessible throughout the scientific community, building on the foundation of a previous project. The first open source 3D-PTV software is the *OpenPTV* project, that was initiated at the early 2000's (OpenPTV consortium, 2014). The developers of *OpenPTV* relied on coding in the C language in order to leverage it's high speed of computation for implementing the complex algorithms involved. However, the C language is not accessible to many of the scientists working in the field, so debugging and installation on modern computers can often be challenging, and algorithms from recent years meant to advance the method (e.g. Ouellette et al. (2006); Xu (2008); Schröder et al. (2015); Bourgoin & Huisman (2020); Brizzolara et al. (2021)) have not yet been implemented in *OpenPTV*. Furthermore, modern computers enable using 3D-PTV with higher level programming tools, while maintaining computational times at a reasonable level. MyPTV_ solves these issues and extends *OpenPTV* through three principles. First, *MyPTV* is written exclusively in Python, which is accessible to a wider range of practitioners and widely used in scientific research. This feature allows rapid prototyping and development of the 3D-PTV method, which is crucial for its further development. Second, the dependency on external packages is kept to the bare minimum and includes only an essential set of widely used and properly maintained packages (currently *Numpy*, *Scipy*, *Matplotlib*, *Pandas*, and *Pyyaml*), thus facilitating the maintenance and cross-platform usability without the need for complex deployment phases. Third, *MyPTV* extends *OpenPTV* by including new algorithms for camera calibration, particle tracking, particle segmentation, and trajectory smoothing, that were never implemented in *OpenPTV*. In particular, *MyPTV* includes a novel algorithm for the crucial stereo-matching step that uses time information to prioritize the correspondence of 3D-trackable trajectories. Indeed, now that the code is more accessible, we envision that in the future *MyPTV* will be further extended by its users to include more developments as they come.

Current capabilities

MyPTV, currently in version 0.2, contains all the necessary code needed to obtain three dimensional particle trajectories from a set of raw image data. In particular, this includes camera calibration, particle segmentation, stereo-matching, particle tracking, smoothing and stitching of broken trajectories. Each of these steps is built as a separate module of *MyPTV*, and generally contains a Python class or two used to perform a particular task. The code is written in an object-oriented style which is suited for the step-by-step structure of 3D-PTV.

Tests

MyPTV had been tested in a series of laboratory experiments. For example, in one of the experiments, seeding particles were tracked in moderate Reynolds numbers turbulent flows generated through an 8-rotating wheels device (Hoyer et al., 2005). Images were taken at 50 frames per second per camera, for a duration of 11.68 seconds using a three camera system. The camera resolution was 1280×1024 pixels^2. The calibration, obtained through MyPTV's calibration module, had a static calibration error of 84 microns, estimated through stereo-matching the 437 points of the calibration target. The particles in our experiment, $50\ \mu$m in diameter, were tracked over a volume of $70 \times 70 \times 40$ mm^3. In each time step, about 850 particles were successfully linked in space and time. A 3D rendered image of particle trajectories obtained in the experiment is shown in Figure 2, showing a subset of 718 particle trajectories recorded during 3 seconds of the measurement.

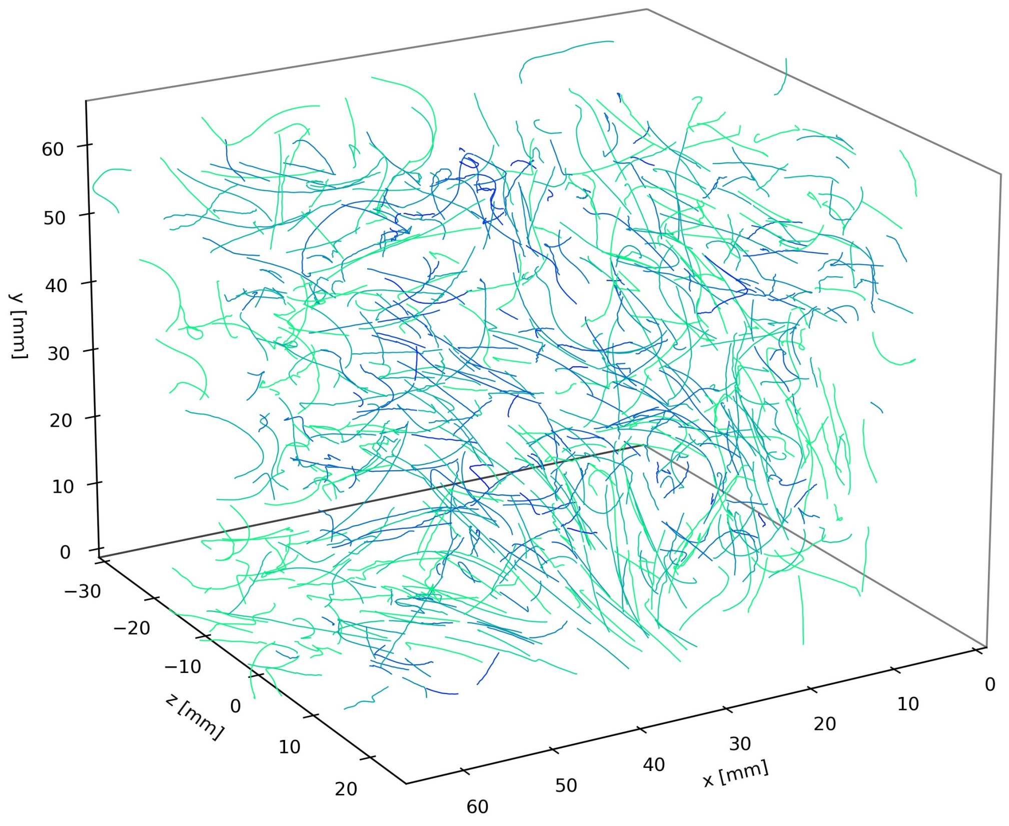

Figure 2: A 3D-rendered image, showing particle trajectories obtained in an experiment. The data shown corresponds to three seconds of measurement and shows 718 trajectories.

Documentation and usability

MyPTV includes several helpful tools to ensure the software user friendly. In particular, MyPTV comes with a detailed user manual which outlines the instructions on how to use the software to achieve the desired results, and all of the functionalities of the various modules, including figures that demonstrate the various file formats used for saving the results of each module. In addition, the software includes an example data set that demonstrates the use of MyPTV on real data.

Furthermore, to enable users who are not experienced with Python to use the software, MyPTV includes a dedicated "workflow" script used to run the various processing steps through a command line interface. Specifically, parameters for each particular experiment can be inserted by the users into a dedicated YAML file, and the workflow script can then be used to automatically perform any particular task. The results of the computations are then saved as text files following a tab-separated value format, which guarantees that the data can be analyzed with other softwares chosen by the users.

Acknowledgements

The author is grateful for help in structuring the package and for numerous suggestions by Alex Liberzon, for fruitful discussions and suggestions by Markus Holzner, and Gal Schnapp, for help in conducting the test experiments and fruitful discussions with Stefano Brizzolara, and for the help in writing this paper from Dana Omer-Shnapp. Furthermore, the author wishes to

acknowledge the significant contribution of the developers and the community of the openPTV project to the development of the current software and the 3D-PTV method in general.

References

Arnèodo, A., Benzi, R., Berg, J., Biferale, L., Bodenschatz, E., Busse, A., Calzavarini, E., Castaing, B., Cencini, M., Chevillard, L., Fisher, R. T., Grauer, R., Homann, H., Lamb, D., Lanotte, A. S., Lévèque, E., Lüthi, B., Mann, J., Mordant, N., … Yeung, P. K. (2008). Universal intermittent properties of particle trajectories in highly turbulent flows. *Physical Revew Letters*, *100*, 254504. https://doi.org/10.1103/PhysRevLett.100.254504

Attanasi, A., Cavagna, A., Del Castello, L., Giardina, I., Jelić, A., Melillo, S., Parisi, L., Pellacini, F., Shen, E., Silvestri, E., & Viale, M. (2015). Greta-a novel global and recursive tracking algorithm in three dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *37*(12), 2451–2463. https://doi.org/10.1109/TPAMI.2015.2414427

Bagøien, E., & Kiørboe, T. (2005). Blind dating—mate finding in planktonic copepods. I. Tracking the pheromone trail of Centropages typicus. *Marine Ecology Progress Series*, *300*, 105–115. https://doi.org/10.3354/meps300105

Bourgoin, M., & Huisman, S. G. (2020). Using ray-traversal for 3D particle matching in the context of particle tracking velocimetry in fluid mechanics. *Review of Scientific Instruments*, *91*(8), 085105. https://doi.org/10.1063/5.0009357

Brizzolara, S., Rosti, M. E., Olivieri, S., Brandt, L., Holzner, M., & Mazzino, A. (2021). Fiber Tracking Velocimetry for two-point statistics of turbulence. *Physical Revew X*, *11*, 031060. https://doi.org/10.1103/PhysRevX.11.031060

China, V., Levy, L., Liberzon, A., Elmaliach, T., & Holzman, R. (2017). Hydrodynamic regime determines the feeding success of larval fish through the modulation of strike kinematics. *Proceedings of the Royal Society B: Biological Sciences*, *284*(1853), 20170235. https://doi.org/10.1098/rspb.2017.0235

Holzner, M., Liberzon, A., Nikitin, N., Lüthi, B., Kinzelbach, W., & Tsinober, A. (2008). A Lagrangian investigation of the small-scale features of turbulent entrainment through particle tracking and direct numerical simulation. *Journal of Fluid Mechanics*, *598*, 465–475. https://doi.org/10.1017/S0022112008000141

Hoyer, K., Holzner, M., Lüthi, B., Guala, M., Liberzon, A., & Kinzelbach, W. (2005). 3D scanning particle tracking velocimetry. *Experiments in Fluids*, *39*(5), 923–934. https://doi.org/10.1007/s00348-005-0031-7

Lüthi, B., Tsinober, A., & Kinzelbach, W. (2005). Lagrangian measurement of vorticity dynamics in turbulent flow. *Journal of Fluid Mechanics*, *528*, 87–118. https://doi.org/10.1017/S0022112004003283

Mass, H. G., Gruen, D., & Papantoniou, D. (1993). Particle tracking velocimetry in three-dimensional flows part i: Photogrammetric determination of particle coordinates. *Experiments in Fluid*, *15*, 133–146. https://doi.org/10.1007/BF00190953

Michalec, F.-G., Fouxon, I., Souissi, S., & Holzner, M. (2017). Zooplankton can actively adjust their motility to turbulent flow. *Proceedings of the National Academy of Sciences*, *114*(52), E11199–E11207. https://doi.org/10.1073/pnas.1708888114

OpenPTV consortium. (2014). *Open Source Particle Tracking Velocimetry*. http://www.openptv.net/

Ott, S., & Mann, J. (2000). An experimental investigation of the relative diffusion of particle pairs in three-dimensional turbulent flow. *Journal of Fluid Mechanics*, *422*, 207–223. https://doi.org/10.1017/S0022112000001658

Ouellette, N. T., Xu, H., & Bodenschatz, E. (2006). A quantitative study of three-dimensional Lagrangian particle tracking algorithms. *Experiments in Fluids*, *40*(2), 301–313. https://doi.org/10.1007/s00348-005-0068-7

Pouw, C. A. S., Toschi, F., Schadewijk, F. van, & Corbetta, A. (2020). Monitoring physical distancing for crowd management: Real-time trajectory and group analysis. *PloS One*, *15*(10), e0240963. https://doi.org/10.1371/journal.pone.0240963

Schröder, A., Schanz, D., Michaelis, D., Cierpka, C., Scharnowski, S., & Kähler, C. J. (2015). Advances of PIV and 4D-PTV "Shake-The-Box" for turbulent flow analysis–the flow over periodic hills. *Flow, Turbulence and Combustion*, *95*(2), 193–209. https://doi.org/10.1007/s10494-015-9616-2

Shnapp, R., Shapira, E., Peri, D., Bohbot-Raviv, Y., Fattal, E., & Liberzon, A. (2019). Extended 3D-PTV for direct measurements of Lagrangian statistics of canopy turbulence in a wind tunnel. *Scientific Reports*, *9*(1), 1–13. https://doi.org/10.1038/s41598-019-43555-2

Sinhuber, M., Van Der Vaart, K., Ni, R., Puckett, J. G., Kelley, D. H., & Ouellette, N. T. (2019). Three-dimensional time-resolved trajectories from laboratory insect swarms. *Scientific Data*, *6*(1), 1–8. https://doi.org/10.1038/sdata.2019.36

Toschi, F., & Bodenschatz, E. (2009). Lagrangian properties of particles in turbulence. *Annual Review of Fluid Mechanics*, *41*, 375–404. https://doi.org/10.1146/annurev.fluid.010908.165210

Virant, M., & Dracos, T. (1997). 3D PTV and its application on Lagrangian motion. *Measurement*, *8*, 1552–1593. https://doi.org/10.1088/0957-0233/8/12/017

Xu, H. (2008). Tracking Lagrangian trajectories in position–velocity space. *Measurement Science and Technology*, *19*. https://doi.org/10.1088/0957-0233/19/7/075105