Note: Tormenta: An open source Python-powered control software for camera based optical microscopy

Federico M. Barabas,1,2,a) Luciano A. Masullo,1,2 and Fernando D. Stefan1,2
1Centro de Investigaciones en Biomanoecientes (CIBION), Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Godoy Cruz 2390, C1425FQD Buenos Aires, Argentina
2Departamento de Física, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Pabellón 1 Ciudad Universitaria, C1428EHA Buenos Aires, Argentina

(Received 5 September 2016; accepted 4 December 2016; published online 16 December 2016)

Until recently, PC control and synchronization of scientific instruments was only possible through closed-source expensive frameworks like National Instruments’ LabVIEW. Nowadays, efficient cost-free alternatives are available in the context of a continuously growing community of open-source software developers. Here, we report on Tormenta, a modular open-source software for the control of camera-based optical microscopes. Tormenta is built on Python, works on multiple operating systems, and includes some key features for fluorescence nanoscopy based on single molecule localization. Published by AIP Publishing. [http://dx.doi.org/10.1063/1.4972392]

The development of digital cameras has expanded tremendously the field of application of wide-field optical microscopy.1–4 Digital imaging and analysis have enabled quantitative optical microscopy measurements on inorganic materials as well as real-time fluorescence microscopy of living cells in their natural environment.5,6 Fluorescence microscopy is nowadays the method of choice for the visualization of biological systems. Modern photodetectors and cameras are sufficiently sensitive to detect the light emitted by single molecules with high signal-to-noise ratio (SNR), enabling the tracking of individual biomolecules7 as well as imaging with spatial resolution beyond the diffraction limit with methods called super-resolution microscopy or fluorescence nanoscopy.8

State-of-the-art optical microscopy requires coordinated control of several instruments: cameras, photodetectors, lasers, shutters, etc., which is typically achieved via custom routines running on a PC and AD/DA converters. National Instruments’ proprietary and closed software LabVIEW has become the most popular environment for instrument control. In fact, many instrument manufacturers offer device drivers exclusively for it. However, in the last years, new open-source alternatives like ImageJ’s plug-in µManager9 are becoming widely used. There is also a constantly growing interest in Python within the scientific community.10 Python is a widely used high-level, general-purpose programming language conceived to improve code readability while maintaining good efficiency. Its syntax allows programmers to express concepts in fewer lines of code than possible in other languages such as C++ or Java. It is suitable for non-experts and a powerful tool for scientists who need to program their own scripts in order to meet the particular needs of their research.

The interest in using Python for research instrumentation stems not only from the fact that it is cross-platform, free, and open-source but also because there is a massive set of ready-to-use high-level libraries, supported by a vibrant community of users and developers. Pyqtgraph, for instance, is a graphical user interface (GUI) library optimized for live applications.11 Another interesting library is Lantz,12 a toolkit that provides functionalities for building applications that communicate with scientific instruments, including a set of ready-to-use Python drivers and detailed instructions for coding new ones from their C counterparts or serial communication commands. There are also user efforts dedicated to provide Python drivers for specific devices. PyAPT13 is an example for the widely used Thorlabs’ stepper motors.

As for the input/output of analog/digital signals, essential for many experimental setups including scanning microscopy systems, vendors like LabJack14 already offer Python drivers for its data acquisition hardware. For National Instruments’ devices, user-developed PyDAQmx15 gives complete Python support.

These tools, alongside with PyQt,16 a Python binding of the GUI toolkit Qt, compose an efficient framework for developing instrument control and data analysis software.

Within this context we present Tormenta, an open-source, cross-platform software written in Python for the control of a wide-field optical microscope and adaptable to any kind of imaging experiment. It was primarily developed for super-resolution fluorescence imaging based on single molecule localization. To this end, Tormenta includes features typically required for this application such as large data set handling, a focus stabilization module, and routines for background subtraction (specific for single-molecule data) and for the precise (nanometric) spatial overlap of multicolor acquisitions.

Tormenta consists of a number of routines written in Python 3.4 for instrument control through a GUI. Instrument drivers and library wrappers were taken from free online repositories,13,14 when available. Otherwise, wrappers of the serial commands or DLL instructions were written using Python packages pyserial and ctypes, respectively, depending on the driver supplied by the manufacturer. Tormenta is maintained

a)Author to whom correspondence should be addressed. Electronic mail: federico.barabas@cibion.conicet.gov.ar
at a GitHub repository, including a readme file with detailed instructions for running it under Windows and Linux.

Tormenta’s GUI was created with PyQt and is flexibly adaptable to different control routines according to user needs. Fig. 1 shows a possible configuration consisting of the following modules: (a) camera control and live view, (b) illumination control, (c) active focus stabilization system, and (d) a Python console. The last three are arranged in tabs and can be interactively moved, resized, or detached from the main window during runtime. A standard menu bar at the top gives access to custom file and analysis tools and a bottom status bar provides online information about the camera temperature and frame rate.

The camera control module is divided in four sections (Figure 1(a)). The one on the right is the camera image view which displays either the live-view streaming or the last frame acquired. In this example the image is displayed with the cube-helix colormap, chosen for its colorblind compatibility and conservation of intensity gradients when printed in grayscale but there is a large variety of colormaps available. On the right hand side, an image histogram allows for live tuning of the minimum and maximum brightness levels for optimal image contrast.

The section on the upper-left side groups all camera settings in a pyqtgraph parameter tree. All operating parameters relevant to optical microscopy of an EM-CCD can be controlled: electron-multiplying gain, field of view, exposure time, etc. In particular, the current version of Tormenta includes a complete driver for the EMCCD Andor iXon 897. The heading box gives loading/saving functionality of imaging camera parameters.

The field of view menu includes options suited for dual-color imaging. Differences in magnification, shear, and image rotation between the two channels must be taken into account to obtain an accurate overlay. This is usually done by imaging fluorescent markers visible on both channels and then finding the affine transformation matrix that minimizes the distance between the same markers as detected in each detection channel. Tormenta includes a routine for this kind of calibration and correction. It finds the optimum affine matrix and uses it to directly save the corrected dual-color imaging data. During two-color acquisitions, the image view can be set to dual-view mode that enables independent histograms for each channel, which is especially useful when the intensity and background of each channel are different.

A third section gathers buttons to run control routines such as switching on and off the live-view streaming, superimposing grids or cross-hairs to the image (e.g., useful for optical alignment), and changing in 1000-fold the illumination intensity by moving a motorized filter wheel (e.g., useful for switching from wide-field visualization to acquisition of super-resolution data).

Finally, a section labeled Recording is dedicated to the configuration and triggering of single frame (snap) and multiple frame acquisitions. Localization-based super-resolution microscopy methods often need thousands of frames for the reconstruction of a super-resolved image, hence the need of a saving format capable of handling gigabyte-sized files, such as TIFF and HDF5 formats, both implemented in Tormenta. An additional file containing the entire system configuration is saved alongside the single frame or multiple frame recording.

The illumination control module (Figure 1(b)) is meant to control the wavelength, intensity, and mode of illumination. It is able to drive three continuous wave lasers from different manufacturers. Each laser control is instantiated from the same Python class, so it is straightforward to change laser units or add a new one. Besides setting the output power of each one, there is a checkbox for setting the laser’s shutter state, i.e., switching on and off any of the illumination sources. The button “Get Intensities” triggers the power measurement of each laser with a photodiode and reports the intensity at the sample using a previous calibration. This module also controls a motorized linear stage (Thorlabs APT Motor) that permits to...
In LabVIEW or Java's
and analysis. Developing custom solutions from the ground-up
developing your own. Compared to commercial software like
alternative to employing commercial imaging software or
resolution fluorescence microscopy, Tormenta stands as an
efficiency, and ease of use.

In conclusion, we have described and made available Tormenta, a free and open-source control software for video optical microscopy. The software is capable of driving simultaneous control of an EMCCD camera, several illumination lasers, motorized filter wheels, and shutters. Also, it provides useful image analysis, calibration routines, and key features for single molecule localization super-resolution techniques, such as handling of large size data sets, active focus stabilization, and two-color view and calibration routines.

We believe Tormenta constitutes a clear demonstration that Python-written software is a practical and reliable alternative to commercial frameworks for experiment control, with the additional advantages of being free, more flexible, and supported by a large developer community.

This work was funded with the support of CONICET, ANCYPT Project Nos. PICT 2009-0110 and PICT 2010-2511, and a Partner Group of the Max-Planck-Society.

13 M. Leung, PyAPT: Controlling Thorlabs APT Using Python Code Using APT.dll and APTlib, Bypassing the ActiveX Controls, https://github.com/mleung/PyAPT.