Multi-Dimensional Spectrogram Application for Live Visualization and Manipulation of Large Waveforms

B. E. Bolling, A. A. Gorzawski, J. Petersson

The European Spallation Source (ESS) is a research facility under construction aiming to be the world's most powerful pulsed neutron source. It is powered by a complex particle accelerator designed to provide a 2.86 ms long proton pulse at 2 GeV with a repetition rate of 14 Hz. Protons are accelerated via cavity fields through various accelerating structures that are powered by Radio-Frequency power. As the cavity fields may break down due to various reasons, usually post-mortem data of such events contain the information needed regarding the cause. In other events, the underlying cause may have been visible on previous beam pulses before the interlock triggering event.

The Multi-Dimensional Spectrogram Application is designed to be able to collect, manipulate and visualize large waveforms at high repetition rates, with the ESS goal being 14 Hz, for example cavity fields, showing otherwise unnoticed temporary breakdowns that may explain the sometimes-unknown reason for increased power (compensating for those invisible temporary breakdowns). Another physical event that was recorded with the tool was quenching of a superconducting cavity in real time in 3D. This paper describes the application developed using Python and the pure-python graphics and GUI library PyQtGraph and PyQt5 with Python-OpenGL bindings.

Methodology

Multiple Python libraries are used to span the Spectrogram application, which can be split into a data retrieval part (pychiver) and a visualization part.

pychiver

Independently to the use cases highlighted by this paper, a broad use python package *pychiver* is available to interact with EPICS-based (Experimental Physics and Industrial Control System) services at ESS. It focuses on accessing data stored in EPICS Archiver, data available in real-time, and configurations saved in the Save and Restore. One of the features used in described hereafter application is the waveform collector. This particular function allows for the creation of dedicated threads to collect or retrieve the data. An important function of a visualization application is the data size.

For the retrieval of the archived data, a pychiver-native function ensures that big blocks of retrievals are avoided. In the case of real-time data, this thread comes with an internal circular buffer that allows for limiting the overall size of collected waveforms. Any pychiver waveform collector comes with the internally computed average values inside the configurable region of interest (ROI), also, an ROI trend over all collected data is available. All these values, in case of real-time data collection, are updated and recomputed upon the new data arrival.

Visualization Elements

The main graphical user interface for the package utilizes the PyQt5 library, with pyqtgraph library components used for visualization of data with python-opengl bindings to render 3D graphics. To manipulate data arrays, standard NumPy array operators are applied onto the processed waveform.

The standard configuration of the 3D spectrogram launches with the intensity displayed by the vertical axis, the waveform position displayed by the left horizontal axis (WF (waveform) array) and the right horizontal axis showing the time the waveform array was captured.

Combined

Either an event-driven data stream is set up via a connection to a PV or data is fetched from an archiver-instance. For both cases, the data is passed through a pre-processing block (restructuring incoming data to allow it to be manipulated and ensuring correct number of arrays are passed for live data), a data manipulation block, followed by setting up the data correctly for the visualization. The final step is to apply the processed data to the 3D model, which is done immediately for the archived data and also for the live data if the visualizer is not paused. Hence, live data can be collected in the background whilst the user may investigate a single event in the visualizer, resulting in that the connection to a PV is not interrupted for as long as the application is running and connection permits this.

A high-level flowchart depicting this is attached in Fig. 1.

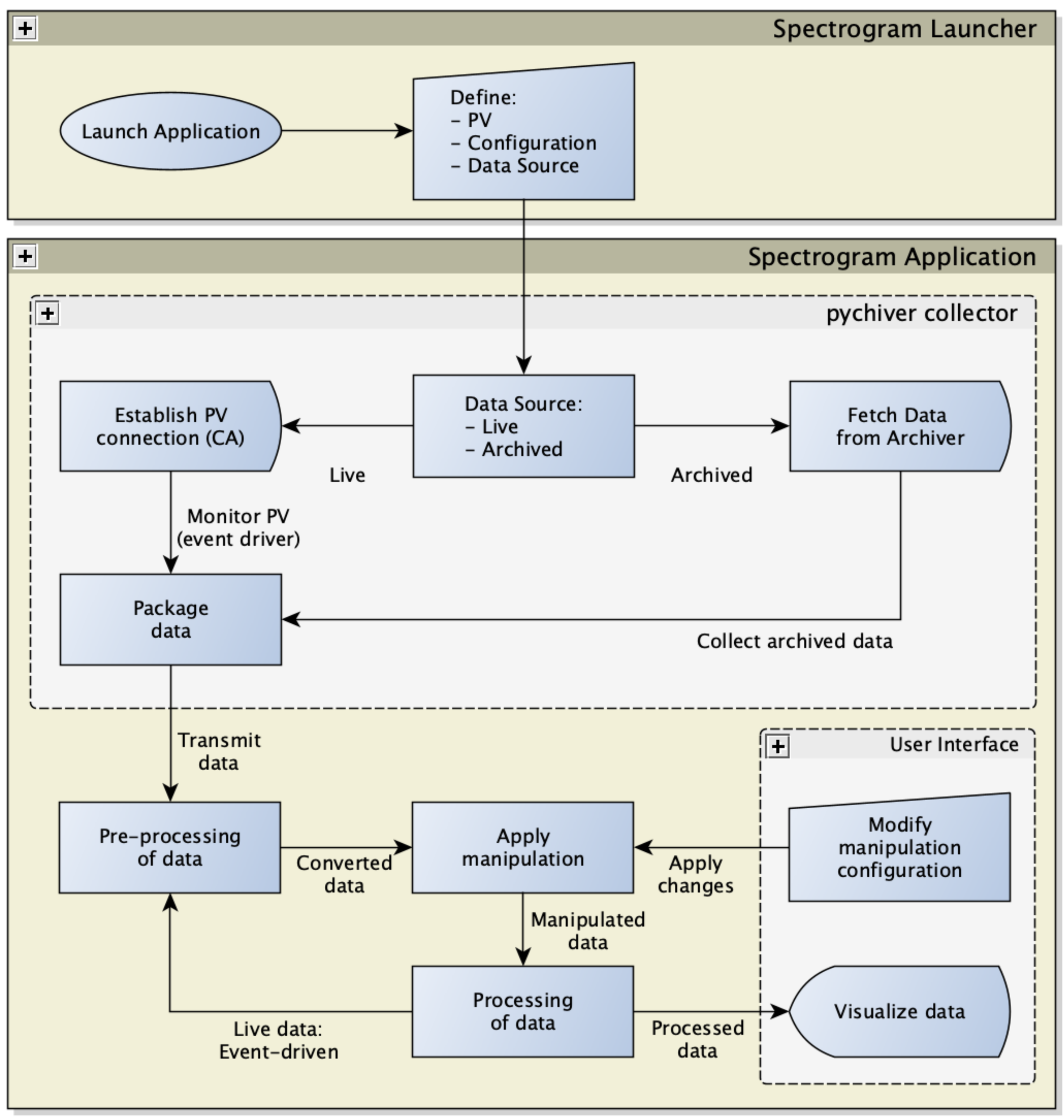

Fig. 1. A high-level flowchart of the Spectrogram application utilizing a simplified model of the pychiver collector.

Samples

The application has been used in the ESS control room whilst being developed on a MacBook Pro, connected via Wi-Fi to office network, to capture and visualize different events to better understand the physics behind them. Its first test use was to capture the quench of a superconducting radiofrequency (SRF) cavity, as shown in Fig. 3.

Further on, it was also used during reconditioning of the Radio Frequency Quadrupole to understand why some pulse-length increments resulted in power interlocks (Fig. 4), what happens to the phase of the buncher cavity field during and after a beamless event (Fig. 5), and checking the evolution of the beam pulse profile during beam commissioning whilst increasing the proton beam current (Fig. 6).

Performance tests

The performance tests are set as benchmarking against the total number of events at 14 Hz to measure the amount of pulses captured and the visualization frame rate for different bin sizes and region of interests. Measuring the amount of waveforms received compared against the system's event receiver's cycle count and the frames per second yields the results shown in Fig. 6, showing that:

- Increased waveform size does not impact data capture, converging towards 67% with time for all test cases.
- For the large array small bin size, the frames per second decreases after ca 1.5min, suggesting that there are some issues with how the application handles buffering.
- The frames per second decreases rapidly with the amount of bins.

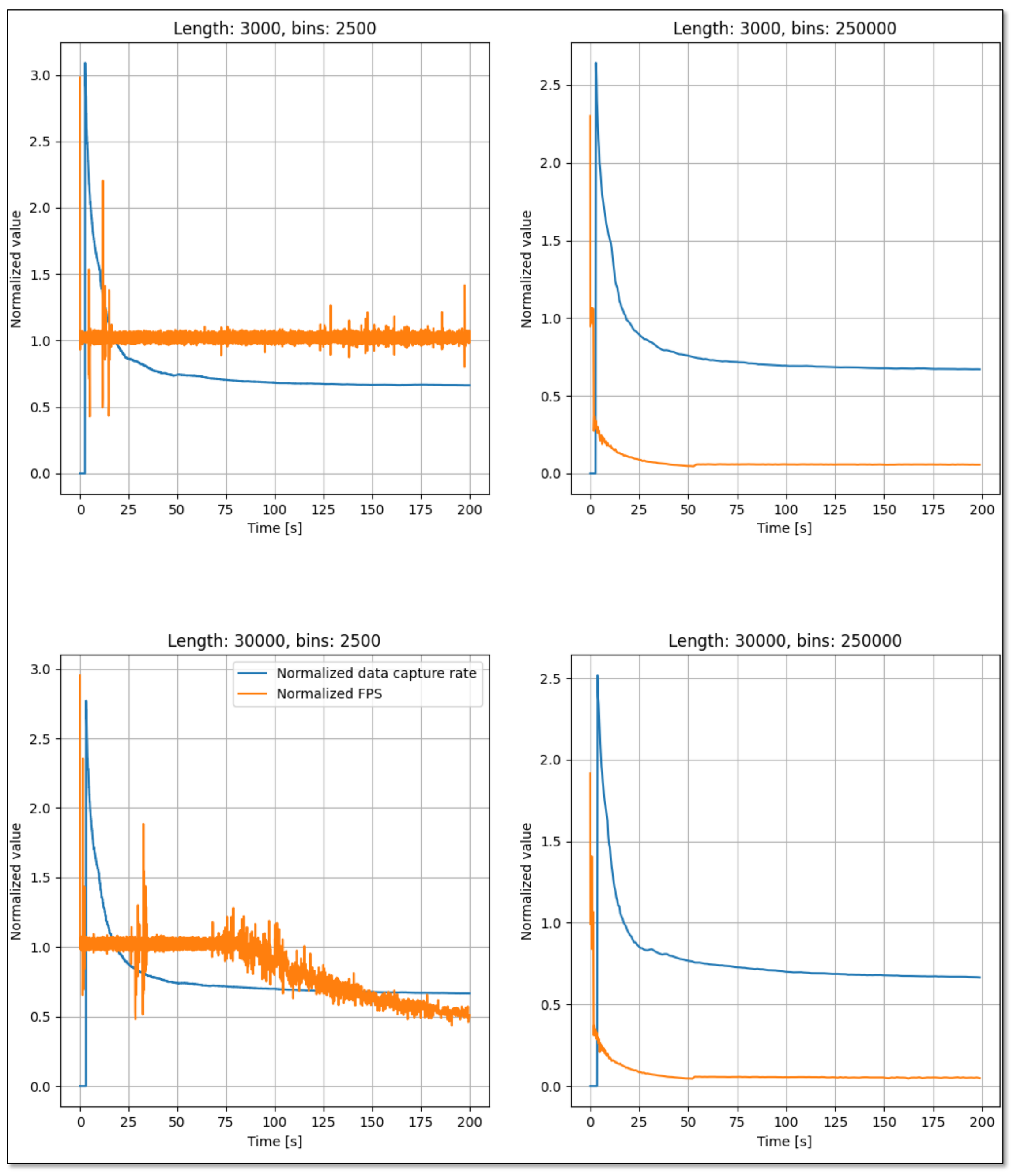

Fig. 2. Benchmark results showing the amount of pulses captured and the frames per second visualized in 3D.

Conclusion

Whilst the application is in a state ready to be and is used, further development is still needed with functionalities including a more stable implementation of the axis labels to ensure no overlap occurs, adding a axis calibrations (static or based on another waveform for the horizontal axis), and a function which separates the waveform in its current state into a new window (snapshot). The latter offers the advantage of observing the current state in real time whilst applying manipulation to and investigating an event in a separate window.

Acknowledgements

The authors want to thank the Linac team members at ESS suggesting that it would be interesting to visualize a SRF quench in 3D, prompting a rapid development of the 3D functionality of the Spectrogram application to have it ready in time for the real event. The authors also want to thank Beam Physics for being available for discussions on the physics of various events, including the ones described.

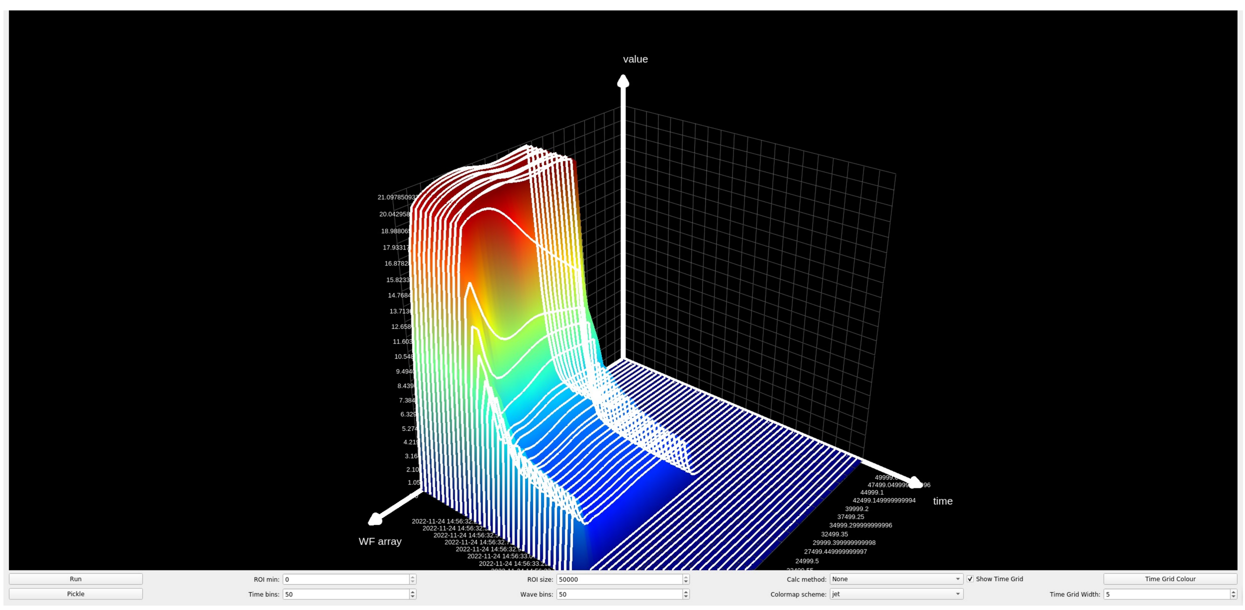

Fig. 3. A SRF quench captured with the first prototype of the application.

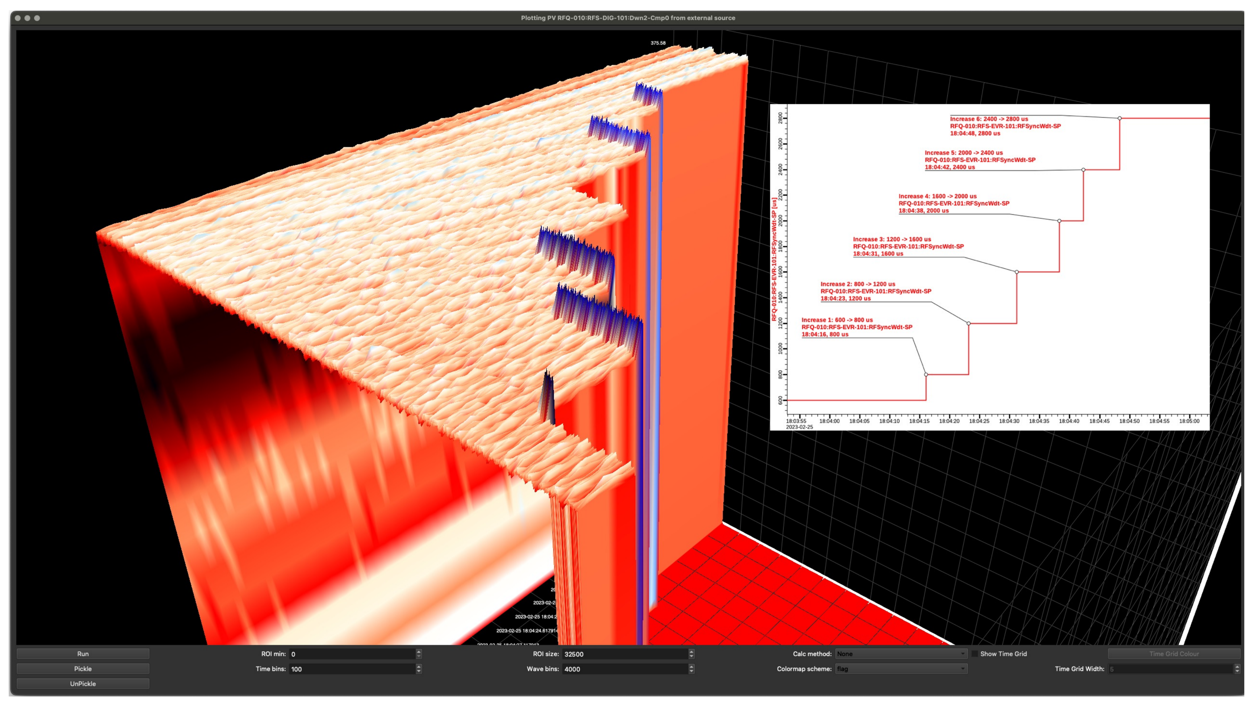

Fig. 5. Radio Frequency Quadrupole pulse length increase resulting in an initial slight increase of power for some incremented regions.

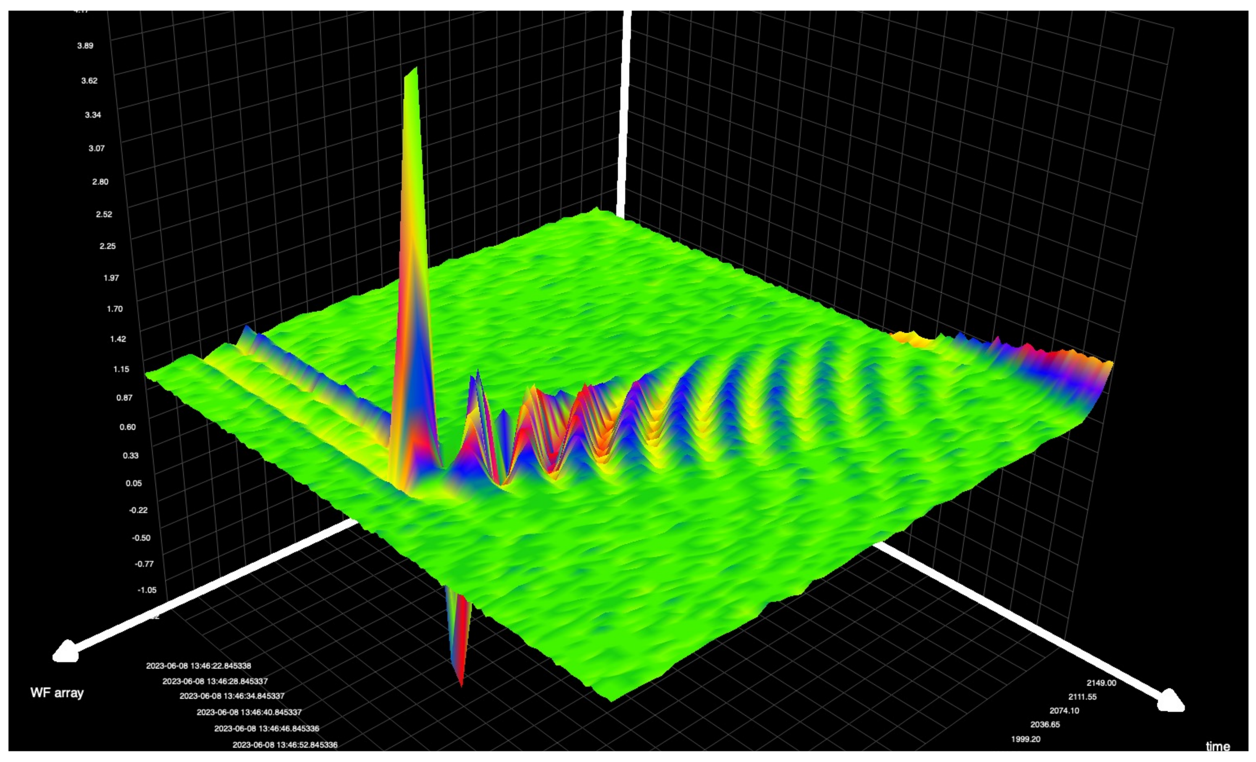

Fig. 4. Buncher cavity phase before, during and after a beam loss event.

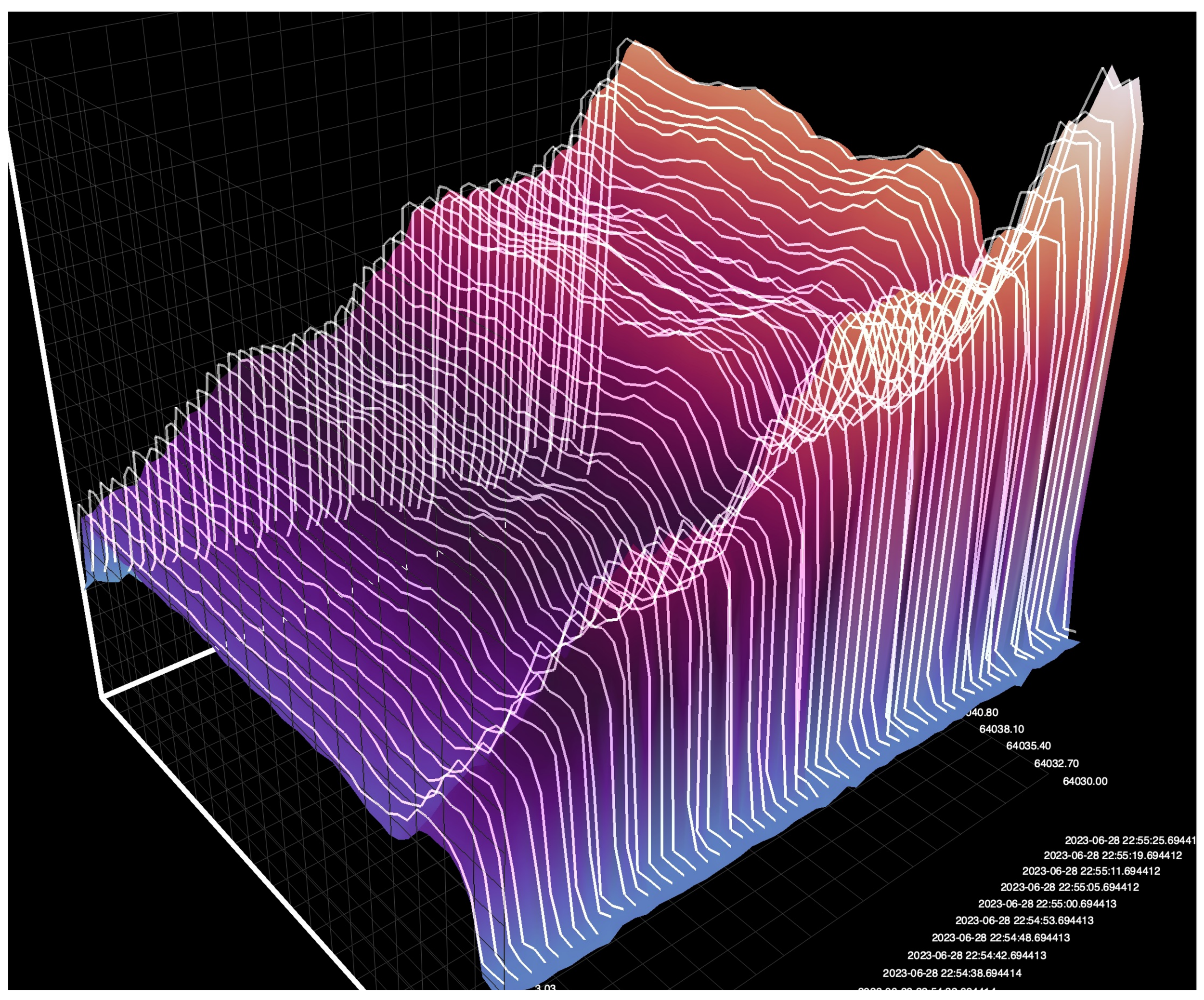

Fig. 6. Beam current pulse profiles captured by a Faraday Cup whilst increasing the beam current.

References

G. Van Rossum and F. L. Drake, "Python 3 Reference Manual", 2009, CreateSpace, Scotts Valley, CA, doi:10.1016/j.ymssp.2005.08.011.

F. Léonard, "Phase spectrogram and frequency spectrogram as new diagnostic tools", Mechanical Systems and Signal Processing, Vol. 21(1), 2007, pp. 125-137, doi:10.1016/j.ymssp.2005.08.011.

PyQt5, "PyQt5 Reference Guide", 2022, Riverbank Computing, https://www.riverbankcomputing.com/software/pyqt/.

https://gitlab.esss.lu.se/accop/pytools/pychiver.

M. Woo et al., "OpenGL programming guide: the official guide to learning OpenGL, version 1.2", 1999, Addison-Wesley Longman Publishing Co., Inc.

C. R. Harris et al., "Array programming with NumPy", Nature, vol. 585, 2020, pp. 357-362, Springer Science and Business Media LLC doi:10.1038/s41586-020-2649-2.

M. Newville, "PyEpics: Python Epics Channel Access", 2019, https://pyepics.github.io/pyepics/.

O. Moore et al., "PyQtGraph - High Performance Visualization for All Platforms", Proceedings of the 22nd Python in Science Conference, 2023, pp. 106-113, doi:10.25080/gerudo-f2bc6f59-00e.

L. R. Dalesio et al., "The experimental physics and industrial control system architecture: past, present, and future", Nucl. Instrum. Methods Phys. Res., Sect. A, vol. 352, 1994, pp. 179-184, doi:10.1016/0168-9002(94)91493-1.

@ESSneutron

TUMBCMO12