Journal of Synchrotron Radiation

ISSN 0909-0495

TomoPy: A framework for the analysis of synchrotron tomographic data

Doğa Gürsoy,* Francesco De Carlo and ? ?

Advanced Photon Source, Argonne National Laboratory, 9700 S. Cass Ave., Argonne IL 60439-4837 USA.
Correspondence e-mail: dgursoy@aps.anl.gov

Abstract goes here.

1. Introduction

Analysis of large tomographic datasets at synchrotron light sources is becoming progressively more challenging due to the increasing data acquisition rates that new technologies in X-ray sources and detectors enable. The next generation of synchrotron facilities that are currently under design or construction throughout the world will provide diffraction limited X-ray sources and is expected to boost the current data rates by several orders of magnitude and stressing the need for the development and integration of efficient analysis tools more than ever. Several synchrotron users, to take advantage of specific instrument characteristics and specifications, already collect data at different facilities, but they are often left alone integrating this data with the available or home-grown software tools.

There are often strong similarities within the same class of instruments that are not systematically utilized to build a coordinated software development platform or framework where each facility and user group can take advantage from or contribute in, ultimately saving on-site resources by sharing the computing tasks with the user community. Here we describe in detail an attempt to provide such a collaborative framework for the analysis of synchrotron tomographic data that has the potential to unify the effort of different facilities and beamlines performing similar tasks. The basic principles of this Python/C++ based framework, called *TomoPy*, are: OS and data format independent, parallelizable, modular and supporting a functional programming style that many researchers in Academia prefer.

2. Background

When it comes to the digital storage of tomographic experimental data and the development of analysis tools at synchrotron light sources around the world, the situation is very heterogeneous. As different research teams and instruments have grown at various facilities, they have often developed local data and analysis models based on instrument hardware specificity and often drawing upon the particular preferences of a scientist writing software.

Many tomographic analysis tools utilize licensed, and often expensive, software packages like Matlab (MathWorks, 2014) and IDL (Exelis, 2014), others rely on specific, and often complex to maintain, computing infrastructure like MPI-based CPU or GPU clusters. The availability of inexpensive multi-core CPU workstations and the reduced cost of computer memory allowed to simplify the computing infrastructure required to process tomographic data (Rivers, 2012), and started the development of single workstation tools able to perform most computing tasks.

This transition is affecting all major synchrotron facilities where new effort is now dedicated into developing new tools that can be deployed at the facility for real time processing as well as distributed to users for off site data processing.

3. TomoPy: A python based framework

There is no free lunch when it comes to the integration of analysis methods. For example, Matlab's or IDL's coding practice which is based on matrix and linear algebra operations that many researchers get accustomed to come with pricy licensing fees compared to open source alternatives. R, (Foundation, 2014), as an open-source alternative, offers a wide variety of statistical data analysis tools but has a steep learning curve and is usually hard to maintain for large projects. Python with its standard library on the other hand provides a modular, readable and manageable framework that researchers can use freely or contribute easily without facing any hassles. Besides, the native control software running at several synchrotron facilities, EPICS, (EPICS, 2014), is interfaced to Python (Newville, 2014), allowing data analysis and real-time feedback on the instrumentation status. These features make Python the method of choice for development framework.

Tomographic data processing at synchrotrons is essentially modular in nature such that the processes can be divided into a number of small independent steps. A natural way to segment the processes is to group functions according to the similarities in transformations applied on data as depicted in figure 1.

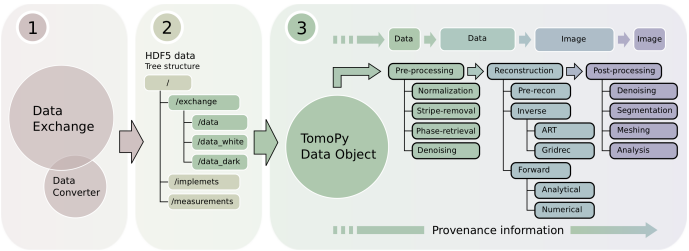

Figure 1
Processes Segmentation

Explain how anyone can contribute. How it is distributed. BSD License. Documentation.

4. Methods

In synchrotron tomography, the imaging sample is placed in the beam path and the x-ray intensity profiles (also called tomograms or projections) are acquired by rotating the sample during x-ray exposure. Almost all data acquisition protocols require another set of data taken in the absence of the sample which is usually refereed to as the white-field (or flat-field) measurements. These two set of measurement together with the rotation angle information are the entry point of the analysis pipeline.

4.1. Pre-processing module

Preprocessing module consists of denoising functions and aims to prepare data for the image reconstruction.

4.1.1. Normalization The need for this acquisition is to compensate for the effect of optical components in the beam path which are superimposed on the projections and show up artifacts in reconstructed images.

4.1.2. Ring removal: The reconstruction quality can be dramatically degraded when local inhomogeneities consistently appear in every projection data. This is usually caused by the impurities of the scintillation detector and can not be totally removed with the flat-field correction. These measurement artifacts usually stand out as vertical stripes in sinograms and are transformed to reconstructions as ring shaped artifacts. The process of clearing these artifacts are called as the ring-removal.

There are numerous ways to correct these effects. The easiest and computationally most effective version is the median filtering along the sinogram. Median filtering works nicely without compromising reconstruction quality if the stripes are a few pixel width. One of the most powerful methods is the Wavelet-Fourier method which exploits the appearance of the

4.1.3. Phase retrieval Single-step methods.

Touch maybe briefly multi-energy approaches with energy-selective measurements.

4.1.4. Registration Data fusion: Local tomography.

Phase-correlation: Correction of wabbling in images. Alignment for multi-step phase retrieval.

A 180 degrees rotation of the sample is generally sufficient to efficiently solve the reconstruction problem. However for some samples, especially when the sample size is larger than the field of view of the detector, a rotation of 360 degrees can be used as an option for data acquisition.

4.2. Reconstruction

Determination of center of rotation.

Gridrec. ART.

4.3. Post reconstruction

??

4.4. Storing Data

Formats supported. Many are happy with TIFF.

Explain storing provenance and its importance.

5. Discussions

Wrap the important points, vision and aims and conclude.

Appendix A
Appendix title

Appendix text.

Acknowledgements

Work supported by U.S. Department of Energy, Office of Science, under Contract No. XXX.

References

EPICS, (2014). The experimental physics and industrial control system.
URL: *http://www.aps.anl.gov/epics/*

Exelis, (2014). Exelis visual information soultion.
URL: *http://www.exelisvis.com*

Foundation, R., (2014). The r project for statistical computing.
URL: *http://www.r-project.org/*

MathWorks, (2014). Matlab the language of technical computing.
URL: *http://www.mathworks.com*

Newville, M., (2014). Pyepics: Epics channel access for python.
URL: *http://cars.uchicago.edu/software/python/pyepics3/*

Rivers, M. (2012). In Developments in X-Ray Tomography VIII, vol. 8506. SPIE.