doi:10.1017/S1431927615008788 Paper No. 0799 Microsc. Microanal. 21 (Suppl 3), 2015

Ettention: Building Blocks for Iterative Reconstruction Algorithms

Tim Dahmen[1,3], Lukas Marsalek[2,3*], Nico Marniok[3,*], Beata Turoňová[3,4], Sviatoslav Bogachev[3], Patrick Trampert[1], Stefan Nickels[1] and Philipp Slusallek[1,3]

[1] German Research Center for Artificial Intelligence, 66123 Saarbrücken, Germany
[2] Eyen SE, Na Nivách 1043/16, 141 00 Praha 4, Czech Republic
[3] Saarland University, 66123 Saarbrücken, Germany
[4] IMPRS-CS Max-Planck Institute for Informatics, 66123 Saarbrücken, Germany

We present a novel software package for tomographic reconstruction in electron microscopy, named Ettention [1]. The software consists of a set of modular building-blocks for iterative reconstruction algorithms. Ettention simultaneously features (1) a modular, object-oriented software design, (2) optimized access to high-performance computing (HPC) platforms such as graphic processing units (GPU) or many-core architectures like Xeon Phi, and (3) accessibility to microscopy end-users via integration in the IMOD package and user interface. We provide developers with a clean application programming interface (API) that allows for extending the software easily and thus makes it an ideal platform for algorithmic research while hiding most of the technical details of high-performance computing. Several case studies are provided to demonstrate the feasibility of the concept [2].

Ettention is built on top of the OpenCL API. While this allows the software to run on many hardware architectures, the OpenCL programming model still exposes a number of technical properties, such as the need to very explicitly handle parallelism and memory management. Therefore, Ettention introduces the concepts of “building blocks” for iterative reconstruction algorithms. Those building blocks include forward projections, back projections, image manipulation, and input/output operations. They encapsulate memory management and parallelism and can be combined to reconstruction algorithms. As Ettention allows the reconstruction volume resolutions to exceed memory on the HPC device, most operations involve consecutive upload of parts of the volume to the GPU, the execution of a kernel that operates on parts of the volume, and the combination of partial results. The system maps memory objects, such as volumes or reconstruction images to configurable HPC representations, such as float buffer or a GPU texture. By tracking the up-to-date status of the different representations the system can detect at runtime when to transfer data to device memory or back.

Compared to existing software packages for tomographic reconstruction in electron microscopy, Ettention fills a gap between IMOD and the ASTRA toolbox. For algorithmic research without the need for immediate application to high-resolution data, we recommend using the ASTRA toolbox because of the powerful MATLAB language binding. The limitation to the in-core case, i.e. to small reconstruction volumes, is less important in this case. For microscopy experimentalists, who require well-established reconstruction algorithms, we recommend using IMOD because of its highly-optimized reconstruction performance on high-resolution data. For projects that require both, algorithmic innovation and immediate application to high-resolution experimental data, we believe that the Ettention software package should be considered because it delivers a reasonable (though not optimal) performance even on high-resolution data and exposes a rich and well-structured API that allows for fast and efficient implementation of algorithmic innovations.

References:

[1] T Dahmen *et al* "The Ettention Software Package" *under review Ultramicroscopy* (2015).
[2] T Dahmen et *al Microsc. Microanal.*(2014) **vol. 20**, pp. 548–60

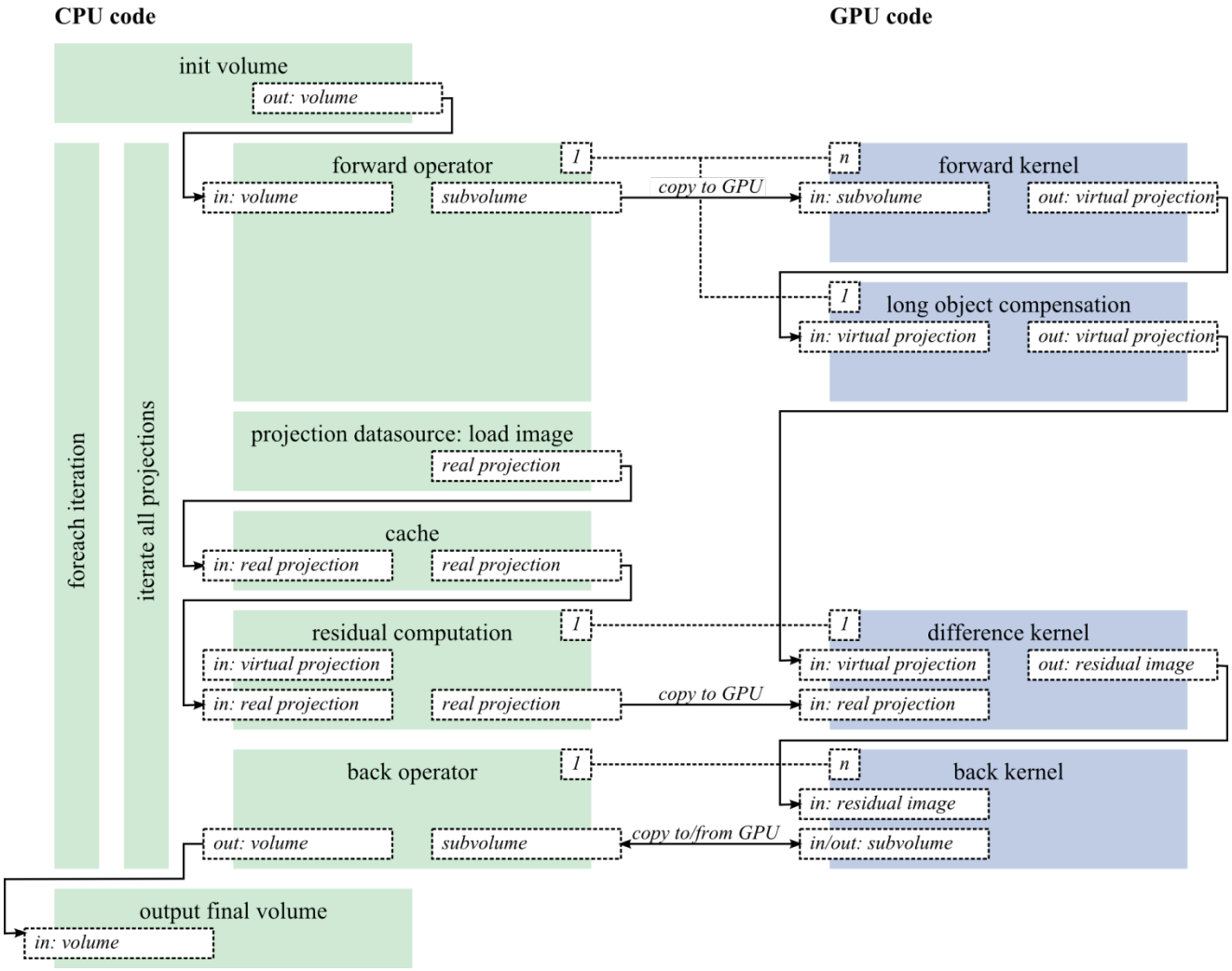

Figure 1: A block diagram of the SART algorithm as an example for an iterative reconstruction algorithm implemented using the building blocks, also called “operators” in Ettention. CPU code objects including dataflow are shown on the left side, GPU kernels on the right side. Some operators like the residual computation corresponds 1:1 with calls to GPU kernels, while the forward and back operators, which work on volumes, require more than one call to the corresponding kernel.

https://doi.org/10.1017/S1431927615008788 Published online by Cambridge University Press