PROCEEDINGS OF SPIE

SPIEDigitalLibrary.org/conference-proceedings-of-spie

High cone-angle x-ray computed micro-tomography with 186 GigaVoxel datasets

Glenn R. Myers, Shane J. Latham, Andrew M. Kingston, Jan Kolomazník, Václav Krajíček, et al.

Glenn R. Myers, Shane J. Latham, Andrew M. Kingston, Jan Kolomazník, Václav Krajíček, Tomáš Krupka, Trond K. Varslot, Adrian P. Sheppard, "High cone-angle x-ray computed micro-tomography with 186 GigaVoxel datasets," Proc. SPIE 9967, Developments in X-Ray Tomography X, 99670U (4 October 2016); doi: 10.1117/12.2238258

SPIE.

Event: SPIE Optical Engineering + Applications, 2016, San Diego, California, United States

High cone-angle x-ray computed micro-tomography with 186 GigaVoxel datasets

Glenn R. Myers[a], Shane J. Latham[a], Andrew M. Kingston[a], Jan Kolomazník[b], Václav Krajíček[b], Tomáš Krupka[b], Trond K. Varslot[c], and Adrian P. Sheppard[a]

[a]The Australian National University, ACT, Australia
[b]Eyen SE, Prague, Czech Republic
[c]FEI Co., Norway

ABSTRACT

X-ray computed micro-tomography systems are able to collect data with sub-micron resolution. This high-resolution imaging has many applications but is particularly important in the study of porous materials, where the sub-micron structure can dictate large-scale physical properties (e.g. carbonates, shales, or human bone). Sample preparation and mounting become difficult for these materials below 2mm diameter: consequently, a typical ultra-micro-CT reconstruction volume (with sub-micron resolution) will be around 3k × 3k × 10k voxels, with some reconstructions becoming much larger. In this paper, we discuss the hardware (MPI-parallel CPU/GPU) and software (python/C++/CUDA) tools used at the ANU CTlab to reconstruct ∼186 GigaVoxel datasets.

Keywords: Micro-tomography, X-ray computed tomography, Iterative tomographic reconstruction, GPGPU, MPI, Parallel computing, Big data

1. INTRODUCTION

X-ray micro-tomography imaging facilities are used for a variety of micron-scale 3d imaging applications. For example, the ANU CTLab works with researchers in fields such as geophysics, biomaterials, materials inspection, and paleontology, amongst others.[1,2] These applications share an emphasis on accurate determination of microstructure within the sample: features on the scale of $\sim 1\mu$m are important in predicting the bulk behavior of the sample.[1] However, they differ in their precise requirements; this leads to a broad range of (sometimes conflicting) demands on the image reconstruction hardware and software.

In geophysical applications, one is often concerned with characterizing the microstructure in a "representative volume" of a rock: i.e. a volume large enough that we can predict the bulk properties of the material on the meter/kilometer scale. The requirement to image a minimum representative volume naturally places a lower limit on the physical size on geophysical samples. Additionally, rocks of interest are often friable and will begin to disintegrate if the sample is cut below ∼2mm in any dimension. In order to resolve $\sim 1\mu$m features in a >2mm sample, the X-ray CT setup must produce a reconstructed volume image with at least 2500 voxels per side.

Other applications place an emphasis on throughput: in materials inspection tasks where a high number of samples must be imaged, samples may be stacked vertically atop one another and scanned together in batches. Thus, it is important to be able to image samples with high vertical-to-horizontal aspect ratios. This may also be important in cases where the sample displays a significant degree of structural asymmetry along a predictable axis, such as layered geological samples, or wood.

The combination of these two requirements leads to large (in the computational sense) reconstruction volumes; a 10:1 vertical-to-horizontal aspect ratio reconstruction with at least 2500 voxels on each of the horizontal axis, is approximately 156 GigaVoxels. Stored in 32-bit floating point format, such a volume is at least 625GB. Reconstructing even a reasonable portion of this image thus requires computation hardware with a large amount

Further author information: (Send correspondence to Glenn Myers)
Glenn Myers: glenn.myers@anu.edu.au

Developments in X-Ray Tomography X, edited by Stuart R. Stock, Bert Müller, Ge Wang, Proc. of SPIE Vol. 9967, 99670U · © 2016 SPIE · CCC code: 0277-786X/16/$18 · doi: 10.1117/12.2238258

of RAM, which is naturally achieved in supercomputers and even desktop clusters by connecting multiple smaller computation nodes using a protocol such as MPI (Message Passing Interface).

X-ray computed tomography (CT) imaging involves collecting a number of radiographs, as the X-ray source and detector move relative to the sample. During X-ray CT image reconstruction we model the path of X-rays through the sample for every radiograph. When dealing with parallel-beam (or equivalently, fan-beam) X-ray illumination, or low cone-angle cone-beam illumination, the computational complexity of this operation scales as $O(N^3 \ln N)$.[3,4] However, at the ANU CTLab the detector (and hence sample) is placed extremely close to the sample, to increase the available X-ray flux.[5] These order $O(N^3 \ln N)$ decimation-based schemes have not been applied to geometries with high cone-angle cone-beam illumination.

When dealing with high cone-angle cone-beam X-ray illumination, the computational complexity of the X-ray CT reconstruction algorithm scales as $O(N^4)$, where N is the number of voxels along each side of the reconstructed volume image. Consequently, reconstructing a large (e.g. 156 GigaVoxel) dataset quickly becomes prohibitively complex, even on supercomputers. There are two standard practices for accelerating X-ray CT image reconstruction from lab-based microfocus sources: (i) moving the computation to massively parallel architectures such as general purpose graphical processing units (GPGPUs);[6] and (ii) using fast, analytical reconstruction algorithms such as Feldkamp-Davis-Kress[7] or Katsevich[8] filtered back-projection (FBP).

Filtered back-projection algorithms assume the X-ray source and detector traverse an ideal trajectory relative to the sample, typically circular or helical in nature.[8,9] Any uncorrected deviations from this trajectory (e.g. due to thermal motion, inaccuracy in movement stages, etc.) will then limit the resolution of the reconstructed volume image.[10] For high-resolution imaging of tall samples, these accumulated inaccuracies lead to an unacceptable loss in resolution.[11] This necessitates the use of (computationally expensive) iterative correction methods,[10–15] counteracting the computational advantages of FBP-type algorithms. Thus, in order to perform high-resolution imaging of tall samples we avoid using FBP reconstruction algorithms at the ANU CTLab. Instead, we adopt iterative X-ray CT reconstruction algorithms with a higher degree of computational complexity,[16] that use a generalized source/detector trajectory.[17]

In summary, the X-ray CT reconstruction hardware and software at the ANU CTLab needs to be able to perform iterative reconstruction of a 156 Gigavoxel volume in a reasonable timeframe (i.e. less than one day), on a variety of MPI-parallel, multi-node, GPGPU-enabled hardware platform (e.g. both a desktop cluster, and the GPU-enabled NCI Raijin supercomputer). Furthermore, the software must be suitable for use in "unconventional" iterative reconstruction tasks. For example, an algorithm may call for iterative refinement over a disconnected subset of the reconstructed volume image,[18] or over a time-sequence of spatially co-located volumes.[19] In this paper we show that the required performance can be achieved by parallelising the book-keeping and communication across the various communications layers, with computation occurring on the GPU. Finally, we present an iterative reconstruction of a 186 GigaVoxel volume image, performed in 16.15 hours on a 4-node desktop cluster.

2. HYBRID MPI/GPU PARALLEL PROCESSING HARDWARE

As discussed in the introduction we are using an MPI-parallel, multi-node, GPGPU-enabled hardware platform. In this paper we use a four-node desktop cluster, though this software has also been run on the GPU-enabled nodes of the NCI Raijin supercomputer. Each node in our cluster has 512GB DDR3 RAM, 2 Intel E5-2690v3 12-core CPUs, and 3 nVidia Titan X GPUs (each with 12GB onboard RAM). The four nodes share an MPI-interconnect across an Infiniband switch, and also share access to 8TB of storage spread across 4 solid-state drives (SSDs).

We conceptually separate this hardware into layers, arranged into a hierarchy of decreasing data storage capacity and increasing processing capacity (as we descend):

- The top layer of our hierarchy is the 8TB of shared SSD storage, with maximum capacity and zero processing power.
- This is connected via SATA-3 to the next layer consisting of 96 CPU cores (8 × Intel E5-2690v3 12-core CPUs), and 1TB of RAM.

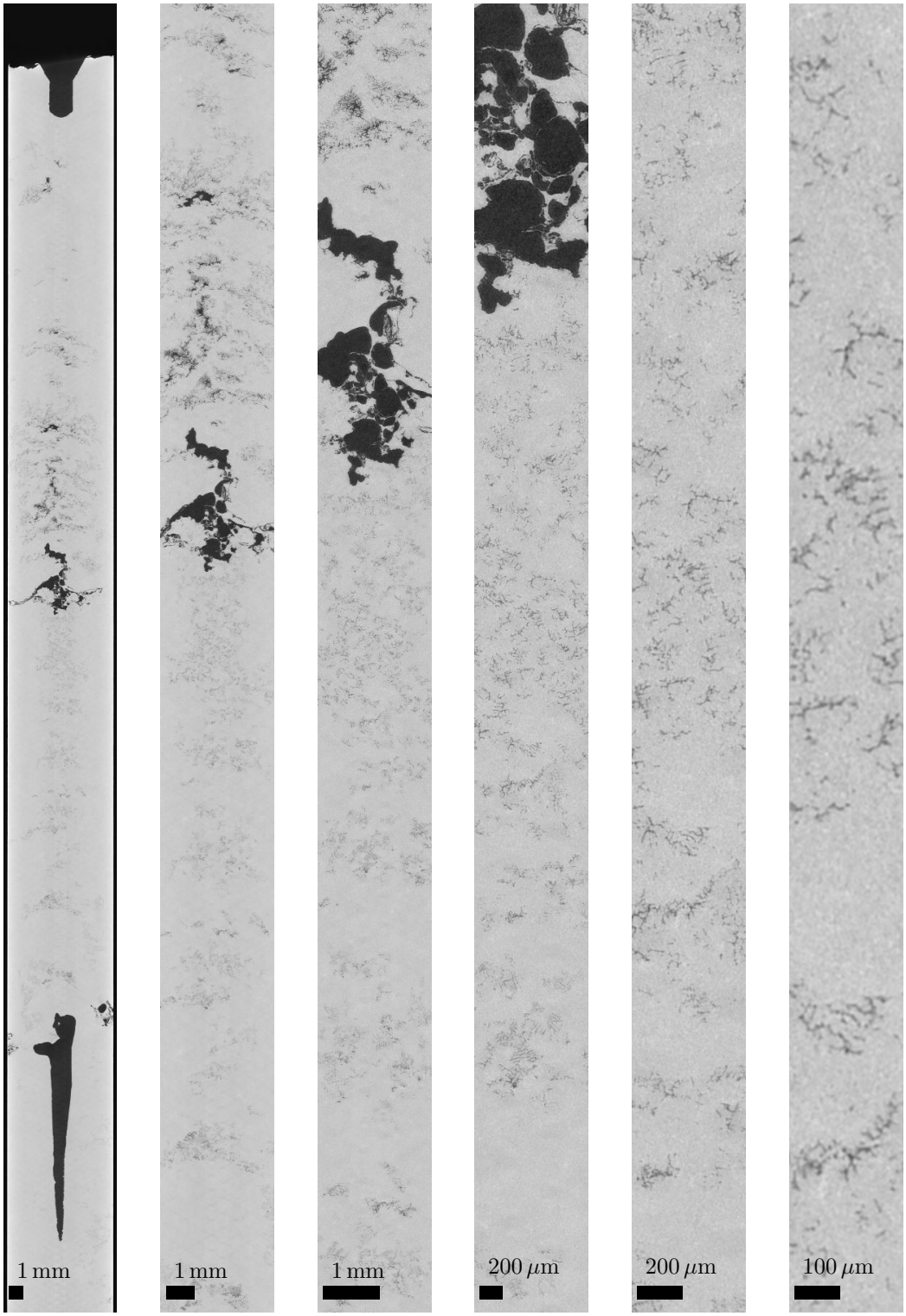

Figure 1. Vertical slices through a 16.15 hour, 186GigaVoxel reconstruction of a sandstone sample from 19,964 radiographs (3040^2 pixel), using one iteration of a preconditioned multigrid Landweber solver.[16] The left-most slice shows the entirety of the object. From left to right, the slices are progressively magnified by factors of 2, and the vertical and horizontal extent reduced proportionately. Voxels are 3.2μm cubed.

- The MPI-interconnect links this to the next layer of 12 CPU threads which manage the GPGPUs, and share the remaining 1TB of RAM. Note that this is an exception to the general rule that each layer leads to an increase in processing power.
- Finally, this is then connected via PCI-e to the bottom-most layer, with minimum storage and maximum processing capacity, consisting of 36,864 CUDA processing cores with 144GB of on-board GPU RAM (12 $\times$ nVidia Titan X GPUs). It is worth noting that when writing the GPU kernels, this "final layer" must be further subdivided into blocks and threads.

Each layer is capable of operating in parallel with the others. Thus, we have two available means of parallelism: computation may be spread within a layer (e.g. MPI-parallelisation within the CPU-layer, as occurs on most supercomputers), and/or it may be spread across different layers (CPU computation, MPI communication, PCI-e communication, and GPU computation can all occur in parallel).

3. ASYNCHRONOUS PARALLELISATION

The hierarchical nature of our hardware suggests we take a similar approach in our software. At each layer, we subdivide the available radiographs and 3d volume data into smaller sub-problems that may be processed independently. Each sub-problem consists of a connected subset of the 3d volume data, and the regions of radiographs (or a subset thereof) that are associated with it.

The size of these sub-problems is chosen to fit on the next layer down: for example, a sub-problem (including both radiographs and 3d image volume) must be less than 12GB to fit on a GPU. Creating sub-problems that can be solved independently leads to some data redundancy: if two sub-volumes overlap in projection, then that region of the radiograph must be duplicated and passed down with both sub-problems. The sub-problem shape is chosen (in conjunction with the imaging trajectory) to minimize this duplication. Typically this results in cubic sub-volumes, as they have a minimal overall footprint on the radiographs and minimal overlap for most trajectories. An exception would be parallel-beam CT reconstruction (e.g. reconstruction from synchrotron data), where layers parallel to the rotation axis are ideal. Depending on the particulars of the hardware, the size and shape of sub-problems at each layer can be further tuned to increase performance.

Once a sub-problem is created, it is solved on the next layer down. Following this, the results are then passed back up, collected, and re-integrated to reverse the separation into sub-problems. This approach is taken recursively at every level, with the same ray-tracing calculations being used at each level to determine what region of each radiograph (if any) is relevant to each subset of the 3d volume data. At the bottom-most layer (i.e. within the innermost loop of the GPU kernel), these ray-tracing calculations link one pixel in a radiograph, with one voxel in the associated 3d volume image.

In this arrangement, as the storage capacity of each layer in our hierarchy diminishes, the number of sub-problems increases. This leads to an increase in the number of associated ray-tracing operations (and thus computational complexity), taking maximum advantage of the increases in computational capacity as we descend the hierarchy. As the demands of each layer change, the programming language changes accordingly. The top-level operations, MPI communication, and user interface are written in python for rapid prototyping in research applications. Data re-arrangement for the GPU places a greater emphasis on speed, and is coded in C++, and the GPU kernels at the bottom-most layer are necessarily coded in CUDA.

Whilst the subdivision and re-integration of sub-problems is simply book-keeping and data copying, it has a significant computational cost and often becomes the bottleneck in reconstruction performance. Thus, we have attempted to parallelise it as much as possible across different layers, as well as within each layer. For example, whilst data for sub-problem B is being re-arranged across the MPI interconnect into sub-sub-problems B-1 through B-12, sub-sub-problems A-1 through A-12 are being re-arranged and divided across the PCI-e connection to the GPGPUs, etc. In the next section, we will show that this form of parallelism results in software sufficiently fast to meet the requirements laid out in section 1.

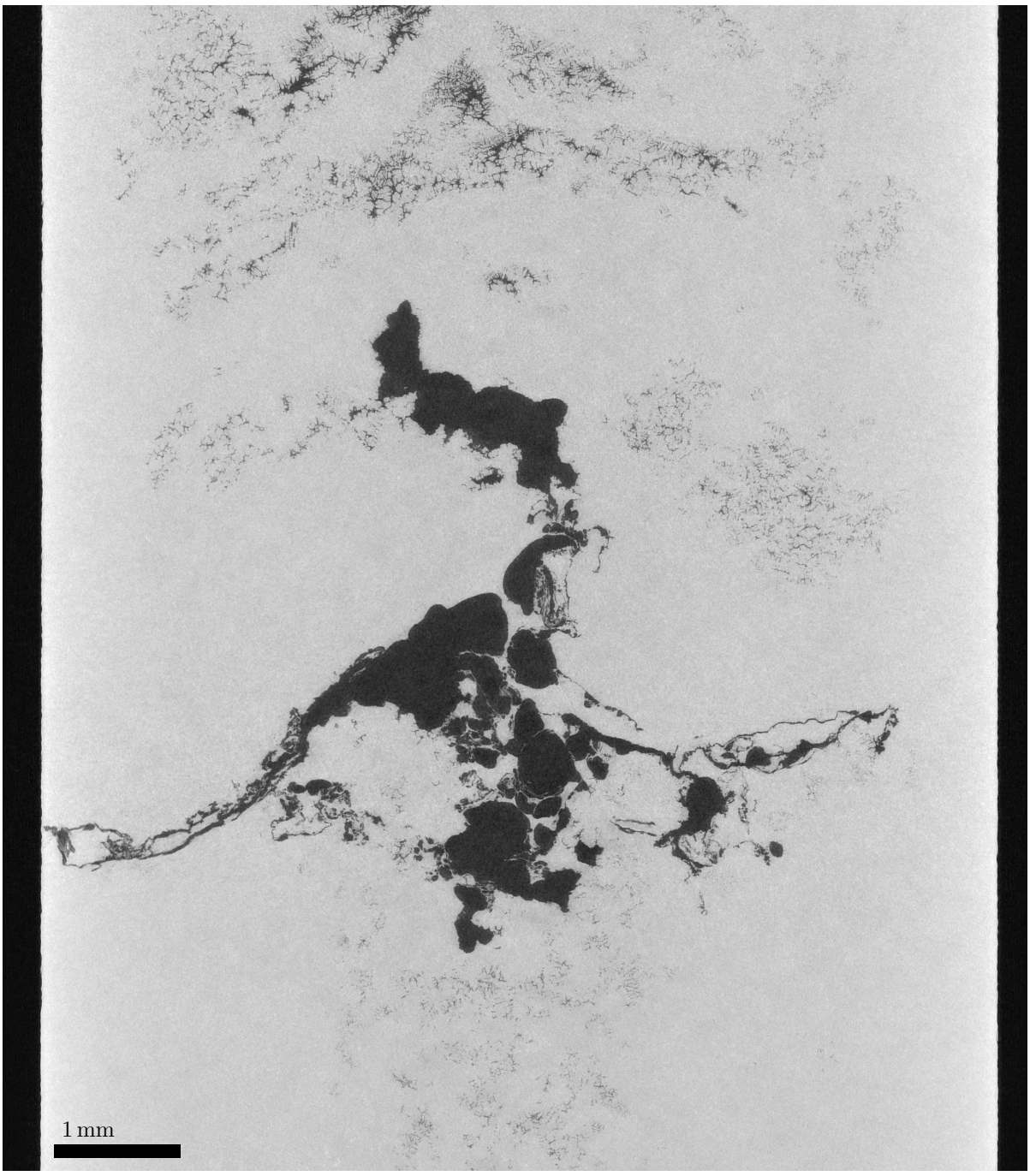

Figure 2. A single subset of a vertical slice through a 16.15 hour, 186GigaVoxel reconstruction of a sandstone from 19,964 3040^2 pixel radiographs, using one iteration of a preconditioned multigrid Landweber solver.[16] Voxels are 3.2μm cubed.

4. RESULTS

4.1 8 GVox backprojection

We begin by presenting an artificial "worst-case" scenario to illustrate: (i) the computational cost of data subdivision and re-integration; and (ii) the speed of the computation kernel itself. In this section, We have simulated 2880 radiographs (2048^2 pixel, 45GB) through a 2048^3 voxel (32GB) volume. These radiographs are collected along a perturbed circular scanning trajectory, with a source-object distance of 4000 voxels. The reconstruction takes place on a single CPU-thread, using only a single GPU (containing 3072 CUDA cores). This leaves no room for parallelism within any but the bottom-most layer. Parallelism across layers is likewise restricted: sub-division only occurs at the bottom-most layer where 160 separate calls are made to the GPU.

The backprojection operation takes 482s in total, with approximately 56s spent performing entirely redundant (in this 1-core, 1-GPU case) data management operations at the higher layers. This supports our earlier assertions that data management and rearrangement consume a non-trivial amount of time, even in this case where the "sub-division" is trivial.

4.2 186 GVox iterative reconstruction in 16 hours

A more realistic example is presented here, to illustrate the performance of the code base in real-world conditions. Zuzana Patakova at FEI collected 19,964 radiographs (3040^2 pixel) of a 10:1 vertical:horizontal aspect-ratio sandstone.

The scanning geometry was a space-filling trajectory[17] with a stride of 15.3 degrees and each voxel projecting to approximately 1800 radiographs, using a source-sample distance of 7.53mm, and a source-detector separation of 330.16mm. Each radiograph was generated using a 0.52 second exposure with an X-ray current of 70μA, and an accelerating voltage of 80keV.

Reconstruction (see fig. 2) of the 186 GigaVoxel (744GB), 2528×2528×29,184 3d volume image was performed in 16.15 hours, using one iteration of a multigrid preconditioned Landweber algorithm.[16] This involved one full-resolution backprojection and one full-resolution projection operation, as well as multiple lower resolution projection and backprojection operations. Assorted $O(N^3)$ operations (e.g. soft thresholding and bilateral filtering) were also performed on the GPU during reconstruction. In contrast to the previous case, this computation used the full 4-node desktop cluster, and the problem was sub-divided at each layer. Thus, MPI communication occurred in parallel with other operations, making the total time spent in MPI communication not particularly meaningful.

The initial sub-division from SSD storage into RAM involved separating the volume into ten separate reconstruction problems, each 2528×2528×3000 voxels. In order to solve each of these sub-problems, the high cone-angle of the illumination made it necessary to pad each sup-problem with an additional 3000 voxels of overscan in the vertical direction. Each of these 2528×2528×3000+3000 voxel sub-problems was solved in approximately 1.6 hours.

5. CONCLUSION

We have demonstrated that a hierarchical, MPI-parallel, GPU-accelerated approach to tomographic reconstruction, is capable of routine iterative reconstruction of large (∼186 GigaVoxel) datasets in less than 24 hours, on a 4-node desktop cluster. The required speed is achieved by parallelising the computation across different layers in the hardware hierarchy, to hide the computationally costly bookkeeping and data subdivision/reintegration operations.

ACKNOWLEDGMENTS

The authors wish to acknowledge Zuzana Patakova at FEI Co., for collecting the experimental data presented in this paper. The authors also wish to acknowledge Ondřej Pacovský at Eyen Co. for productive discussion, and Eyen Co. for allowing use of the proprietary Eyen CUDA Image Processing library (www.eyen.eu). This research was supported under the Australian Research Council's Linkage Projects funding scheme (project number LP150101040), in collaboration with FEI.

REFERENCES

[1] Caubit, C., Hamon, G., Sheppard, A. P., and Øren, P. E., "Evaluation of the reliability of prediction of petrophysical data through imagery and pore network modelling," in [*22nd International Symposium of the Society of Core Analysts*], Society of Core Analysts (October 2008). SCA2008-33.

[2] Evans, P. D., Lube, V., Averdunk, H., and Limaye, A., "Visualizing the Microdistribution of Zinc Borate in Oriented Strand Board Using X-Ray Microcomputed Tomography and SEM-EDX," *Journal of ...* (2015).

[3] Brandt, A., Mann, J., Brodski, M., and Galun, M., "A fast and accurate multilevel inversion of the Radon transform," *Siam Journal on Applied Mathematics* (2000).

[4] George, A. and Bresler, Y., "Fast and accurate decimation-in-angle hierarchical backprojection algorithms," in [*IEEE Nuclear Science Symposium Conference Record, 2005*], 5 pp., IEEE (2005).

[5] Sheppard, A., Latham, S., Middleton, J., Kingston, A., Myers, G., Varslot, T., Fogden, A., Sawkins, T., Cruikshank, R., Saadatfar, M., Francois, N., Arns, C., and Senden, T., "Techniques in helical scanning, dynamic imaging and image segmentation for improved quantitative analysis with x-ray micro-ct," *Nuclear Instruments and Methods B* **324**, 49–56 (2014).

[6] Nesterets, Y. and Gureyev, T., "High-performance tomographic reconstruction using graphics processing units," in [*18th World IMACS Congress and MODSIM09 International Congress on Modelling and Simulation*], Anderssen, R., Braddock, R., and Newham, L., eds., 1045–1051, Modelling and Simulation Society of Australia and New Zealand and International Association for Mathematics and Computers in Simulation (2000).

[7] Feldkamp, L. A., Davis, L. C., and Kress, J. W., "Practical cone-beam algorithm," *J. Opt. Soc. Am. A* **1**, 612–619 (1984).

[8] Katsevich, A., "Theoretically exact filtered backprojection-type inversion algorithm for spiral CT," *Siam Journal on Applied Mathematics* (2002).

[9] Natterer, F., [*The Mathematics of Computerized Tomography*], Society for Industrial and Applied Mathematics, Philadelphia (2001).

[10] Kingston, A., Sakellariou, A., Varslot, T., Myers, G. R., and Sheppard, A., "Reliable automatic alignment of tomographic projection data by passive auto-focus," *Medical Physics* **38**, 4934 (2011).

[11] Latham, S., Kingston, A., Recur, B., Myers, G., and Sheppard, A., "Multi-resolution radiograph alignment for motion correction in x-ray micro-tomography," *Proc. SPIE, Developments in X-ray Tomography X* (in press).

[12] Dengler, J., "A multi-resolution approach to the 3D reconstruction from an electron microscope tilt series solving the alignment problem without gold particles," *Ultramicroscopy* (1989).

[13] Mayo, S., Miller, P., Gao, D., and Sheffield-Parker, J., "Software image alignment for X-ray microtomography with submicrometre resolution using a SEM-based X-ray microscope.," *Journal of Microscopy* **228**, 257–263 (Dec. 2007).

[14] Myers, G., Kingston, A., Varslot, T., and Sheppard, A., "Extending reference scan drift correction to high-magnification high-cone-angle tomography," *Optics Letters* **36**, 4809–4811 (2011).

[15] Bleichrodt, F. and Batenburg, K. J., "Automatic Optimization of Alignment Parameters for Tomography Datasets," *Image Analysis* (2013).

[16] Myers, G., Kingston, A., Latham, S., Recur, B., Turner, M., Beeching, L., and Sheppard, A., "Rapidly-converging multigrid reconstruction of cone-beam tomographic data," *Proc. SPIE, Developments in X-ray Tomography X* (in press).

[17] Kingston, A., Myers, G., Latham, S., Veldkamp, J., and Sheppard, A., "Optimal x-ray source scanning trajectories for iterative reconstruction in high cone-angle tomography," *Proc. SPIE, Developments in X-ray Tomography X* (in press).

[18] Batenburg, K. and Sijbers, J., "DART: a fast heuristic algebraic reconstruction algorithm for discrete tomography," *Image Processing, 2007. ICIP 2007. IEEE International Conference on* **4**, IV–133–IV–136 (2007).

[19] Myers, G. R., Geleta, M., Kingston, A. M., Recur, B., and Sheppard, A. P., "Bayesian approach to time-resolved tomography.," *Optics Express* **23**, 20062–20074 (July 2015).