Proceedings of the 13th International Conference
on Computational and Mathematical Methods
in Science and Engineering, CMMSE 2013
24–27 June, 2013.

The ASTRA Tomography Toolbox

Willem Jan Palenstijn[1], K. Joost Batenburg[2,1] and Jan Sijbers[1]

[1] *iMinds-Vision Lab, Universiteit Antwerpen, Belgium*

[2] *CWI, Amsterdam, The Netherlands*

emails: WillemJan.Palenstijn@ua.ac.be, Joost.Batenburg@cwi.nl, Jan.Sijbers@ua.ac.be

Abstract

We present the ASTRA (All Scale Tomographic Reconstruction Antwerp) toolbox: an open source, GPU-accelerated library for 3D image reconstruction in tomography. While most of the current software tools available for tomography offer only limited flexibility in the geometry of the experimental setup and the set of available reconstruction algorithms, the ASTRA toolbox provides a set of highly flexible building blocks that can be used to *construct* advanced reconstruction algorithms, effectively removing these limitations. We describe how the design of the ASTRA toolbox allows for full flexibility in specifying the geometry while still maintaining an efficient mapping onto the underlying hardware. The ASTRA toolbox comes with a MATLAB interface for easy user interaction and is available for both Windows and Linux.

Key words: tomography, GPU computing, software

1 Introduction

The aim of tomographic reconstruction is to create a 3D representation of an object starting from a series of its projections, taken along a range of angles. Depending on the type of tomography, projection images are acquired using X-rays, electrons, or some other type of beam that interacts with the object. Traditionally, reconstruction is performed using backprojection methods such as Filtered Backprojection (FBP) and adapted versions like Feldkamp-Davis-Kress (FDK) [9]. These algorithms are computationally efficient, but suffer from artefacts if only limited, or highly noisy projection data is available.

Algebraic methods employ a linear model of the projection process, effectively solving the linear system $\boldsymbol{W}\boldsymbol{x} = \boldsymbol{p}$, where the matrix $\boldsymbol{W}$ denotes the projection operator, the

 ISBN: 978-84-616-2723-3

vector $\boldsymbol{p}$ denotes the projection data, and the vector $\boldsymbol{x}$ corresponds to the image to be reconstructed. Algebraic methods can yield more accurate reconstructions, but are typically much slower due to their iterative nature. Examples of such methods are the Simultaneous Algebraic Reconstruction Technique (SART)[9], the Simultaneous Iterative Reconstruction Technique (SIRT)[6], or Conjugate Gradients Least Squares (CGLS)[8].

More recently, sophisticated algorithms have been developed that are capable of incorporating various kinds of prior knowledge in the reconstruction. As examples, we mention the FISTA algorithm [3, 4] for Total Variation minimization (TV-min) and the Discrete Algebraic Reconstruction Technique (DART) for discrete tomography [2].

A common drawback of iterative methods, and particularly those that involve prior knowledge, is that significantly more computation time is required to perform reconstructions compared to backprojection methods. Typically, the computationally most expensive parts are the so-called Forward Projection (algebraically the evaluation of $\boldsymbol{Wx}$) and Back Projection (algebraically $\boldsymbol{W^T y}$). Efficient implementations of these operations are therefore essential.

Various software packages have been developed for tomographic reconstruction. Some of these packages were designed with a particular application in mind, are often highly optimized and typically employs a fixed set of reconstruction algorithms (e.g., [1, 12]). On the other hand, there are several toolboxes available that are specifically aimed at algorithm development (e.g., [5, 7]), but these are more suitable for prototyping than for the development of high-performance algorithms that can process large datasets.

The ASTRA toolbox is an open source toolbox that aims to fill this gap by providing flexible high-performance implementations of Forward Projection (FP) and Back Projection (BP) operators, as well as ready-to-use iterative reconstruction algorithms built on top of these. By exposing building blocks that are both flexible and highly efficient, the ASTRA toolbox is not only suitable for prototyping, but the resulting algorithms can immediately be applied to large experimental datasets with high efficiency. The toolbox uses NVIDIA Graphics Processing Units (GPUs) to accelerate the computation, and has a MATLAB interface for convenient algorithm prototyping.

The toolbox is available as open source software (GPLv3) and can be downloaded from `http://visionlab.ua.ac.be/software/` for Windows and Linux operating systems.

In the following sections, the features and performance characteristics of the ASTRA toolbox are summarized.

2 Features

As mentioned in the introduction, the ASTRA toolbox utilizes GPU acceleration to provide high-performance and flexible building blocks for advanced reconstruction algorithm development in MATLAB.

 ISBN: 978-84-616-2723-3

Specifically, it has the following features.

- Support for 2D geometries (parallel beam, fan beam) and 3D geometries (cone beam, parallel beam) with full flexibility for choosing the positions of source and detectors.
- Reconstruction voxel size independent of detector pixel size for higher or lower resolution reconstructions.
- Standard iterative algorithms with full GPU acceleration.
- Accelerated FP and BP for use in custom algorithms from MATLAB.

A distinguishing feature is that in the 3D geometries, it is possible to freely position the detector in 3D space for each projection direction separately, as well as the X-ray source (for cone beam geometries) or ray direction (for parallel beam geometries). This allows the transparent use of less traditional trajectories such as dual tilt axis setups, modelling any misalignment in experimental setups, or even the formation of diffraction spots in X-ray diffraction contrast CT [11].

To illustrate basic usage of the toolbox from MATLAB, we show a script executing a SIRT reconstruction of a 2D slice using GPU acceleration. The downloadable version of the ASTRA toolbox contains additional sample scripts.

This script shows the concepts of setting up the geometry, creating data objects, setting up a reconstruction algorithm, and running it.

```
% Set up the geometry:
% A 256x256 reconstruction volume, and 180 projection angles
% between 0 and pi, with 256 detector pixels of width 1.0.
vol_geom = astra_create_vol_geom(256, 256);
proj_geom = astra_create_proj_geom('parallel', 1.0, 256, ...
                                   linspace2(0,pi,180));


% Create the ASTRA data objects for the sinogram and reconstruction.
% The matrix 'data' is expected to already contain the sinogram.
sinogram_id = astra_mex_data2d('create', '-sino', proj_geom, data);
rec_id = astra_mex_data2d('create', '-vol', vol_geom, 0);

% Set up the parameters for a reconstruction using the GPU.
% This will use SIRT. Other GPU options include
% SART_CUDA, EM_CUDA, FBP_CUDA.
cfg = astra_struct('SIRT_CUDA');
cfg.ReconstructionDataId = rec_id;
```

 ISBN: 978-84-616-2723-3

```
cfg.ProjectionDataId = sinogram_id;
% Add constraints to limit the reconstruction to the interval [0,1]
cfg.option.MinConstraint = 0.0;
cfg.option.MaxConstraint = 1.0;

% Create the algorithm object from the configuration structure
alg_id = astra_mex_algorithm('create', cfg);

% Run 150 iterations of the algorithm
astra_mex_algorithm('iterate', alg_id, 150);

% Get and show the result
rec = astra_mex_data2d('get', rec_id);
figure; imshow(rec, []);

% Clean up. Note that the used GPU memory is tied up in the
% algorithm object, and main RAM in the data objects.
astra_mex_algorithm('delete', alg_id);
astra_mex_data2d('delete', rec_id);
astra_mex_data2d('delete', sinogram_id);
```

3 Performance

To illustrate the GPU performance of the ASTRA toolbox, we report the results of a series of benchmarks.

The first set of experiments, in Table 1, consists of volumes reconstructed as a stack of 2D slices using the SIRT algorithm. The reconstructions were carried out on three datasets with sizes typical for Electron Microscopy, which are also used as benchmarks in [1, 10, 13, 14]. We compared our performance against these results. We have used a NVIDIA GTX 280 and GTX 680 for this test. The reported times are in seconds for a single SIRT iteration of the entire volume.

All three datasets consisted of 61 projection images of size 356×506 for Dataset A, 712×1024 for Dataset B, and 1424×2024 for Dataset C. The reconstructed volumes were $356 \times 506 \times 148$, $712 \times 1012 \times 296$, and $1424 \times 2024 \times 591$ voxels, respectively.

The second set of experiments, in Table 2, shows the performance for true 3D reconstructions with a cone beam geometry. These reconstructions require more memory to perform, so we have used a NVIDIA Tesla M2090 with 6GB RAM for this benchmark. The reported times are again in seconds for a single SIRT iteration of the entire volume. Both datasets use 180 projection images of sizes 256×256 and 512×512. The reconstruction

 ISBN: 978-84-616-2723-3

volumes are $256 \times 256 \times 256$ and $512 \times 512 \times 512$ voxels, respectively.

The third and final set of experiments, in Table 3, shows the performance for even larger 3D reconstructions, utilizing multiple NVIDIA Tesla M2090 cards simultaneously. This set of constructions uses an Electron Microscopy geometry with slightly misaligned positioning so the reconstruction can not be carried out as a stack of 2D reconstructions. Both datasets use 80 projections, of sizes 512×512 and 1024×1024. The corresponding reconstruction volumes were $512 \times 512 \times 512$ and $1024 \times 1024 \times 1024$ voxels, respectively.

	Dataset A	Dataset B	Dataset C
Xu et al. — GTX 280 [14]	2.10	10.66	58.47
Vázquez et al. — Tesla C2050 [13]		3.70	37.91
Agulleiro et al. — Xeon 5405 (8 thr.) [1]	0.35	2.94	24.55
ASTRA — GTX 280 [10]	0.45	2.00	11.13
ASTRA — GTX 680	0.35	1.27	6.67

Table 1: Comparison of runtime (in seconds) for a single SIRT iteration of various recent CPU and GPU implementation.

	256×256	512×512
ASTRA — Tesla M2090	0.42	3.28

Table 2: Benchmarks of ASTRA runtime (in seconds) for a single SIRT iteration.

	$512 \times 512 \times 512$	$1024 \times 1024 \times 1024$
ASTRA — 1× Tesla M2090	2.32	26.13
ASTRA — 2× Tesla M2090	1.58	17.51
ASTRA — 4× Tesla M2090	1.62	9.92

Table 3: Benchmarks of ASTRA runtime (in seconds) for a single SIRT iteration, using multiple GPUs.

4 Conclusion

In this paper, we have presented the ASTRA toolbox: an open source, GPU-accelerated toolbox for 3D tomographic image reconstruction with a user friendly Matlab interface. The reported benchmarks show that the ASTRA toolbox outperforms current state-of-the-art tomography software libraries with respect to computational speed. Since it allows full flexibility of the acquisition geometry, it provides a basis for the rapid development

 ISBN: 978-84-616-2723-3

of advanced iterative reconstruction methods, which can then be applied directly to large experimental datasets.

Acknowledgements

This work was financially supported by the iMinds-SuperCT project (Interdisciplinary Institute for Technology, a research institute founded by the Flemish Government).

References

[1] J.-I. Agulleiro and J. J. Fernández. Evaluation of a multicore-optimized implementation for tomographic reconstruction. *PLOS ONE*, 7(11):e48261, 2012.

[2] K. J. Batenburg and J. Sijbers. DART: a practical reconstruction algorithm for discrete tomography. *IEEE Transactions on Image Processing*, 20(9):2542–2553, 2011.

[3] A. Beck and M. Teboulle. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. *IEEE Transactions on Image Processing*, 18(11):2419–2434, 2009.

[4] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[5] J. Fessler. Image reconstruction toolbox. `http://web.eecs.umich.edu/~fessler/code/`.

[6] P. Gilbert. Iterative methods for the 3D reconstruction of an object from projections. *Journal of Theoretical Biology*, 36(1):105–117, 1972.

[7] P. C. Hansen and M. Saxild-Hansen. AIR tools — a MATLAB package of algebraic iterative reconstruction methods. *Journal of Computational and Applied Mathematics*, 236(8):2167–2178, 2012.

[8] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 157(49):409–436, 1952.

[9] A. C. Kak and M. Slaney. *Principles of Computerized Tomographic Imaging*. SIAM, 2001.

[10] W. J. Palenstijn, K. J. Batenburg, and J. Sijbers. Performance improvements for iterative electron tomography reconstruction using graphics processing units (GPUs). *Journal of Structural Biology*, 176:250–253, 2011.

 ISBN: 978-84-616-2723-3

[11] P. Reischig, A. King, L. Nervo, N. Vigano, Y. Guilhem, W. J. Palenstijn, K. J. Batenburg, M. Preuss, and W. Ludwig. Advances in X-ray diffraction contrast tomography: flexibility in the setup geometry and application to multiphase materials. *Journal of Applied Crystallography*, 46(2):297–311, 2013.

[12] K. Thielemans, S. Mustafovic, and C. Tsoumpas. STIR: software for tomographic image reconstruction release 2. In *Nuclear Science Symposium Conference Record, 2006, IEEE*, volume 4, pages 2174–2176, 2006.

[13] F. Vázquez, E. M. Garzón, and J. J. Fernández. Matrix implementation of simultaneous iterative reconstruction technique (SIRT) on GPUs. *The Computer Journal*, 54(11):1861–1868, 2011.

[14] W. Xu, F. Xu, M. Jones, B. Keszthelyi, J. Sedat, D. Agard, and K. Mueller. High-performance iterative electron tomography reconstruction with long-object compensation using graphics processing units (GPUs). *Journal of Structural Biology*, 171(1):142–153, 2010.

 ISBN: 978-84-616-2723-3