Fast tomographic reconstruction on multicore computers

J. I. Agulleiro[1] and J. J. Fernandez[2,*]

[1]Supercomputing and Algorithms group, University of Almeria, 04120 Almeria and [2]National Centre for Biotechnology, National Research Council (CSIC), 28049 Madrid, Spain

ABSTRACT

Summary: Tomo3D implements a multithreaded vectorized approach to tomographic reconstruction that takes full advantage of the computer power in modern multicore computers. Full resolution tomograms are generated at high speed on standard computers with no special system requirements. Tomo3D has the most common reconstruction methods implemented, namely weighted Back-projection (WBP) and simultaneous iterative reconstruction technique (SIRT). It proves to be competitive with current graphic processor unit solutions in terms of processing time, in the order of a few seconds with WBP or minutes with SIRT. The program is compatible with standard packages, which easily allows integration in the electron tomography workflow.

Availability: http://www.cnb.csic.es/%7ejjfernandez/tomo3d

Contact: JJ.Fernandez@cnb.csic.es

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

Electron tomography (ET) allows visualization of complex specimens in its native cellular context at near-molecular resolution (Lucic *et al.*, 2005). A series of projection images is taken with an electron microscope from a unique biological sample at different orientations around a single tilt axis. Tomographic reconstruction algorithms then combine those images so as to derive the 3D structure of the specimen. The number of images in the series usually ranges from 60 to 200. Due to the resolution needs, the image size generally ranges between 1024×1024 and 2048×2048 pixels, though 4096×4096 can now be used as well. Weighted Back-projection (WBP) is the standard reconstruction method. Iterative methods, such as Simultaneous Iterative Reconstruction Technique (SIRT), exhibit better performance in ET (Lucic *et al.*, 2005). However, they demand much longer processing time, which has prevented their extensive use so far.

The complexity of the algorithms along with the data size turns 3D reconstruction into a computational intensive problem. Traditionally, high performance computing has been used to cope with the demands in ET (Fernandez, 2008). Recently, the use of graphics processor units (GPUs) has revolutionized the field as they provide exceptional performance-to-cost ratio (Castano-Diez *et al.*, 2007; Vazquez *et al.*, 2010; Xu *et al.*, 2010), though a relatively specific

hardware component, the GPU, is still necessary. Current stand-alone computers present tremendous power, thanks to technological and architectural advances (Hennessy and Patterson, 2007), but they still have not been fully exploited in the field.

We present a program, Tomo3D, that implements an approach to tomographic reconstruction based on the full use of the computational power within modern multicore computers. It takes advantage of the multiple cores and of the vector instructions within the cores, and also uses general code optimization techniques to make the most of the computers. The program significantly accelerates the 3D reconstruction methods WBP and SIRT, and even outperforms current GPU solutions. Furthermore, this program enables real-time tomography on standard multicore computers, without the need for clusters or GPUs.

2 IMPLEMENTATION

Current computers usually ship with one or more multicore processor chips, every one comprising several powerful cores (e.g. dual-2, quad-4, hexa-6). Moreover, the individual cores contain vector instructions, typically known as SIMD (Single Instruction, Multiple Data), which perform multiple computations with different data elements of the same kind simultaneously (Hassaballah *et al.*, 2008). In particular, the SSE instructions (Streaming SIMD Extensions), present in all Intel and AMD processors, can perform four floating-point operations of the same type (addition, multiplication, etc) with their corresponding four pairs of operands. These features turn current computers into extremely powerful platforms with two levels of parallelism available for the programmer: threads and SSE instructions to exploit cores and vector units within the cores, respectively.

The 3D reconstruction problem in ET can be decomposed into a set of independent 2D reconstruction sub-problems corresponding to the slices perpendicular to the tilt axis. This has traditionally been the data decomposition used in ET for high-performance computing (Fernandez, 2008; Zheng *et al.*, 2007). Each of the 2D slices of the volume can then be computed with any of the reconstruction methods, either WBP or SIRT but now working in 2D, from the corresponding set of 1D projections. The reconstruction of the independent slices can then be carried out in parallel.

Tomo3D uses a multithreaded-vectorized approach to tomographic reconstruction (Fig. 1). It relies on the Single Program, Multiple Data (SPMD) paradigm whereby the different threads essentially execute the same task over different data. The program creates a number of threads, typically as many as cores in the computer, that run in parallel. The program keeps a pool with the slices to reconstruct, clustered in slabs of four slices. The slabs are asynchronously dispatched to the threads when these become idle. When a slab is alloted to a thread, its four slices are reconstructed simultaneously, thanks to the SSE instructions (Agulleiro *et al.*, 2010). Once a thread finishes the reconstruction of the slab, it requests for another one. The fact that slabs are dispatched on demand provides an inherent load-balancing mechanism (Wilkinson and Allen, 2005), and makes Tomo3D flexible to adapt to situations where the computer is shared by several users/programs.
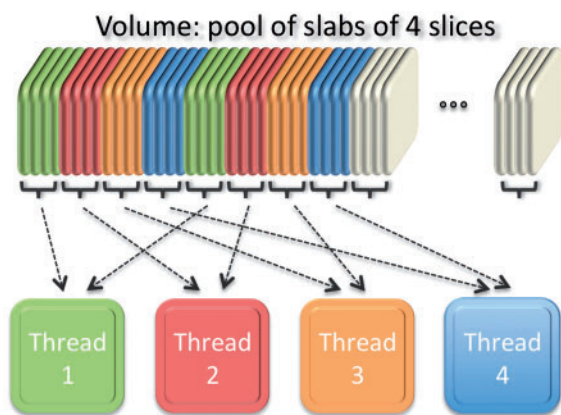
Fig. 1. Multithreaded-vectorized tomographic reconstruction in Tomo3D. A pool of slabs of four slices to be reconstructed is maintained. As many threads as cores run concurrently in the system. The slabs are asynchronously dispatched to the threads as soon as they are idle. The four slices in a slab are reconstructed simultaneously, thanks to vector instructions. In the figure, allocation of slabs to threads are color-coded.

Tomo3D has been implemented in C under Linux, using Pthreads and SSE intrinsics. Single-core code optimization techniques have also been used to better exploit the memory hierarchy and the internal processing units within the cores (Agulleiro *et al.*, 2010). Memory consumption has been reduced to a minimum, only keeping a small subset of slabs and precomputed tables (e.g. cosines, etc.) used throughout the reconstruction process. I/O operations have been optimized to reduce disk latencies. Tomo3D is equipped with the two most common reconstruction methods in the field: WBP and SIRT. For the iterative method SIRT, a number of options are available: the positivity constraint that ensures that all the points in the reconstruction be positive, a weighting to compensate the 'long object effect' (Xu *et al.*, 2010) and the possibility of resuming a reconstruction with more iterations. Tomo3D reads tilt-series and writes the output tomograms in MRC format and is fully compatible with standard ET packages (e.g. IMOD). Tomo3D is provided as a stand-alone, download-and-go, program that does not require any special hardware or software library in the system, which makes it especially attractive for structural biology laboratories. The program is freely available for predominant processing plafforms and can be built for other platforms on request to authors.

3 ILLUSTRATIVE PERFORMANCE TESTING

To illustrate the performance of Tomo3D in typical structural studies by ET, we used tilt-series of 70 and 140 images of sizes 1024×1024 and 2048×2048 to yield volumes of $1024 \times 1024 \times 256$ and $2048 \times 2048 \times 256$. The computation was carried out on a state-of-the-art server with two quad-core processors Intel Xeon E5405 at 2.0 GHz and on a standard desktop computer with a quad-core processor Intel Core 2 Q9550 at 2.83 GHz under Linux. Both reconstruction methods, WBP and SIRT (30 iterations) were tested. The experiments were carried out using eight threads on the server (denoted by 8T) and four threads on the desktop computer (denoted by 4T). For comparison, the reconstructions were also computed using the standard package in the field (see Supplementary Data).

Table 1 shows the processing times required by Tomo3D for the different experiments as well as the average speedup factors (based upon the time required by a single core with no use of vector instructions). As far as the processing time is concerned, Tomo3D succeeds in yielding full resolution reconstructions with WBP in the order of seconds (3–27 s and 2–16 s with 4 or 8 cores, respectively), which enables it for real-time ET (Zheng *et al.*, 2007). Even more interestingly, Tomo3D is able to provide iterative reconstructions in just a few minutes (3–23 min and 2–16 min, respectively).

Finally, Tomo3D was compared to the most advanced GPU solution in the field. A reconstruction of $1424 \times 2024 \times 591$ was computed from 61 images of

Table 1. Processing times (seconds) and speedup factors[a]

	70 images		140 images		Speedup	
WBP	4T	8T	4T	8T	4T	8T
$1024 \times 1024 \times 256$	3.4	2.1	6.0	4.2	11.9	25.1
$2048 \times 2048 \times 256$	13.1	8.4	26.9	15.9	11.3	26.0
SIRT (30 iter.)	4T	8T	4T	8T	4T	8T
$1024 \times 1024 \times 256$	165.8	119.6	331.6	239.3	14.1	27.7
$2048 \times 2048 \times 256$	675.0	493.5	1380.3	1014.7	13.7	26.4
	61 images					
SIRT (1 iter.)	GPU[b]	4T	8T			
$1424 \times 2024 \times 591$	58.5	32.2	24.6			

[a]4T denotes 4 threads on a desktop computer with one quad-core processor (2.83 GHz). 8T denotes 8 threads on a server with two quad-core chips (2.00 GHz).
[b](Xu *et al.*, 2010). Nvidia GPU GTX 280.

1424×2024 using SIRT. Table 1 presents the results obtained with Tomo3D and those presented by Xu *et al.* (2010) on a Nvidia GPU GTX 280 with one iteration of SIRT. It is clearly observed that Tomo3D with four and eight threads on the desktop and server, respectively, outperforms the GPU solution.

4 CONCLUSION

The multithreaded-vectorized approach to tomographic reconstruction in Tomo3D allows computation of full resolution tomograms at high speed on standard computers. There are no special system requirements to run the program and, furthermore, it is fully compatible with standard packages in the field. All these are the major contributions of Tomo3D from the user's point of view. The good performance of Tomo3D stems from the full exploitation of the computing power latent in modern multicore computers. The processing time required by the methods implemented in Tomo3D, WBP and SIRT, turns out to be competitive with GPUs.

Conflict of Interest: none declared.

REFERENCES

Agulleiro,J.I. *et al.* (2010) Vectorization with SIMD extensions speeds up reconstruction in electron tomography. *J. Struct. Biol.*, **170**, 570–575.

Castano-Diez,D. *et al.* (2007) Implementation and performance evaluation of reconstruction algorithms on graphics processors. *J. Struct. Biol.*, **157**, 288–295.

Fernandez,J.J. (2008) High performance computing in structural determination by electron cryomicroscopy. *J. Struct. Biol.*, **164**, 1–6.

Hassaballah,M. *et al.* (2008) A review of SIMD multimedia extensions and their usage in scientific and engineering applications. *Comput. J.*, **51**, 630–649.

Hennessy,J.L. and Patterson,D.A. (2007) *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann, San Francisco, CA.

Lucic,V. *et al.* (2005) Structural studies by electron tomography: From cells to molecules. *Annu. Rev. Biochem.*, **74**, 833–865.

Xu,W. *et al.* (2010) High-performance iterative electron tomography reconstruction with long-object compensation using GPUs. *J. Struct. Biol.*, **171**, 142–153.

Zheng,S.Q. *et al.* (2007) UCSF tomography: An integrated software suite for real-time electron microscopic tomographic data collection, alignment, and reconstruction. *J. Struct. Biol.*, **157**, 138–147.

Vazquez,F. *et al.* (2010) A matrix approach to tomographic reconstruction and its implementation on GPUs. *J. Struct. Biol.*, **170**, 146–151.

Wilkinson,B. and Allen,M. (2005) *Parallel Programming*. Prentice Hall, Upper Saddle River, NJ.