doi:10.1017/S1431927613005862
Microsc. Microanal. 19 (Suppl 2), 2013

SEM Autofocusing and Astigmatism Correction using FFT and GPGPU Techniques

N.H.M. Caldwell, A.J. Marshall, B.C. Breton and D.M. Holburn

Department of Engineering, University of Cambridge, Trumpington Street, Cambridge CB2 1PZ, United Kingdom

In our previous work, we reported on the useful contribution that employing GPGPUs (General Purpose Graphics Processing Units) for highly parallel computations could make to automation in scanning electron microscopy, specifically automated focusing [1]. With tens to thousands of processor cores in graphics cards and NVIDIA's Compute Unified Data Architecture (CUDA) enabling access to them by extensions of standard programming languages, the potential performance improvements can be substantial [2]. This paper reports on research into exploring GPGPU for instrument automation by considering an alternative algorithm for combined autofocusing and astigmatism correction.

K.H. Ong and coworkers published an algorithm for analyzing Fast Fourier Transforms (FFT) of SEM images and using this to iteratively autofocus and correct astigmatism [3]. In a focus series, the best focused image is the image with the sharpest features. In the frequency domain, a Fourier transform of an image will appear as a point cloud, with higher frequency components represented as points further from the origin; thus the most focused image will have the widest point cloud. Astigmatism in the image will cause the point cloud to be elliptical rather than circular – the direction of the major axis of the ellipse reflects the direction of the astigmatism. From a given starting point, the algorithm applies a fixed offset to the working distance in each direction and takes the FFT of the image at each offset. Each FFTs is first logarithmically scaled and then filtered via a threshold to give a cluster of points, the number of which provides a measure of the image sharpness. The two values are compared using a focus metric, which yields the direction in which better focus can be found and the magnitude of the improvement in focus. If the magnitude is less than a given threshold, focus is deemed to have been found; if not, the working distance is altered by a fixed amount in the direction of better focus and the process repeats until focus is found. To correct for astigmatism, the algorithm takes two FFTs, one at a positive offset, the other at a negative offset from the focus. The FFTs are segmented into eight regions and a threshold applied, yielding a set of astigmatism points. The astigmatism points are counted and analogous segment counts compared to give metrics for x- and y-astigmatism. The stigmator correctors are adjusted to increase or decrease stigma in whichever direction the stigmatism is more severe and the process repeats until the metrics indicates that astigmatism has been eliminated (or reduced to an acceptable level). Computation of FFTs and frequency domain threshold filtering are tasks well suited to parallelized GPGPU implementations, and a project begun to reimplement the algorithm under CUDA.

The instrumental setup was a Carl Zeiss 1430VP SEM, retrofitted with a low-specification NVIDIA GeForce 8600 GT graphics card with 32 processor cores. Carl Zeiss' Application Programming Interface to the SmartSEM™ software permitted direct access and control of instrument parameters, both locally and across a network. The experimental software was designed and implemented using a combination of the CUDA Toolkit (version 4.x), the NVIDIA GPU Computing SDK (version 4.x), and Microsoft Visual Studio 2008. To avoid the necessity of writing a CUDA FFT suite from scratch, the NVIDIA CUFFT library was used. This is an implementation of the FFTW (Fastest Fourier Transform in the West), based on the Cooley-Tukey algorithm. Unfortunately the CUFFT library (at least in version 4.x) has a number of undocumented limitations; ruling out various possible programming errors

and uncovering what these actually were required significant time and effort. The CUFFT library (at time of writing) only works correctly for 2D FFTs if the transform is complex-to-complex and the input is a square matrix. Moreover the 2D FFT only provides the non-redundant coefficients (for computational efficiency); the focus and astigmatism correction requires the full FFT output so the rest of the coefficients must be recovered. The final implementation uses a sequence of CUDA kernels to preprocess 512-square SEM images into a form acceptable to CUFFT, invokes the library, post-processes the output to recover the full FFT, and then applies the necessary threshold filtering and counting. The CUDA approach did achieve a significant speedup – in speed tests using FFTW in both cases, the 2D FFT tasks requires 78ms using the low-spec GeForce card versus 995 ms on an Intel Core2 Duo CPU.

Trials of the algorithm revealed a number of serious limitations, related to the fixed size offsets, which could prevent the algorithm "homing in" on correct focus. Where the offset is too small compared to the difference between current and desired working distance, the offset images will be similar and thus it will be difficult to correctly discern the direction of the necessary change and the sharpness metrics. Where the offset is too large relative to the difference between the current and desired working distance, the metrics will again be unreliable. If the working distance adjustment is small relative to the total distance to focus, the algorithm will take multiple iterations to complete. If the adjustment is too large, it will overshoot and may "hop around" the search space without converging, leading to significant risk of hysteresis effects. Similar issues exist with the astigmatism correction. The algorithm as described only detects the relative difference in focus between two images – if the algorithm starts at a fully defocused image, then neither of the offset images may be sufficiently sharper and the algorithm may be tricked into believing focus has been found.

A number of mechanisms were attempted as means of improving the algorithm. Firstly the threshold used in the focus metric was made dependent upon the magnification so that for higher magnification, a more precise working distance would be sought. The focus offset was similarly made dependent upon the magnification. To avoid the issue of overshooting, the focus adjustment step size was reduced as the algorithm approaches focus, making it dependent on magnification and the focus metric. Although the changes showed promise, it was not possible within the project's duration to test them thoroughly over a significantly diverse range of instrument operating parameters, nor to enhance the accuracy and robustness of the astigmatism correction. In conclusion, while the research again shows the utility of a GPGPU approach, the importance of original algorithmic robustness is critical. Further work will concentrate on improving the robustness of the astigmatism correction component, as this appears less encumbered by the discovered limitations and more likely to provide a reliable method [4].

References:

[1] N.H.M. Caldwell *et al*, Microsc. Microanal. **18 (Suppl S2)** (2012), p. 1210.
[2] J. Sanders and E. Kandrot, *CUDA by Example: An Introduction to General-Purpose GPU Programming*, Addison Wesley, 2010
[3] K.H. Ong, J.C.H. Phang and J.T.L. Thong, Scanning **19** (1997), p. 553.
[4] NVIDIA GeForce and CUDA are trademarks of NVIDIA Corporation. SmartSEM™ is a trademark of Carl Zeiss Microscopy. This research was supported by funding from Carl Zeiss Microscopy. The authors gratefully acknowledge the assistance of Carl Zeiss personnel, especially Daniel Aldridge and David Hubbard.