Nebula

Monte Carlo simulator of electron–matter interaction

van Kessel, L.; Hagen, C. W.

DOI
10.1016/j.softx.2020.100605

Publication date
2020

Document Version
Final published version

Published in
SoftwareX

SoftwareX 12 (2020) 100605

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

Nebula: Monte Carlo simulator of electron–matter interaction

L. van Kessel *, C.W. Hagen

Delft University of Technology, Department Imaging Physics, Lorentzweg 1, 2628 CJ Delft, The Netherlands

ARTICLE INFO

Article history:
Received 11 June 2020
Received in revised form 1 October 2020
Accepted 1 October 2020

Keywords:
Monte Carlo simulation
Electron scattering
GPU
SEM
Lithography
Metrology

ABSTRACT

Monte Carlo simulations are frequently used to describe electron–matter interaction in the 0–50 keV energy range. It often takes hours to simulate electron microscope images using first-principle physical models. In an attempt to maintain a reasonable speed, empirical models are sometimes used.

We present an open-source software package with first-principle physical models, which can run on GPUs for fast results. Typical electron microscope images can be obtained in minutes instead of hours on consumer-grade hardware, without any loss of accuracy.

Code metadata

Current code version	1.0.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX_2020_254
Code Ocean compute capsule	N/A
Legal Code Licence	BSD-3-Clause
Code versioning system used	git
Software code languages, tools, and services used	C++, CUDA, Python, CMake, HDF5, ELSEPA
Compilation requirements, operating environments & dependencies	C++11, CUDA 8.0, Python 3.5
If available Link to developer documentation/manual	N/A
Support email for questions	L.C.P.M.vanKessel@tudelft.nl

1. Motivation and significance

The interaction of 0–50 keV electrons with matter is of great interest in several fields of science. For example, scanning electron microscopes (SEMs) are used as metrology tools by the semiconductor industry to study process parameters such as critical dimension (CD) and line-edge roughness (LER). As device features shrink below 20 nm, understanding the image formation process in SEMs becomes increasingly important. Additionally, the manner in which electrons propagate through photoresist defines the CD and LER in both electron-beam lithography and extreme ultraviolet lithography. Furthermore, electron-beam induced patterning is a promising technique for high-resolution patterning within an SEM. In these techniques, slow electrons emitted from the substrate dissociate precursor gas molecules to induce deposition or etching. A better understanding of electron scattering and secondary electron emission is crucial to advancing each of these techniques.

The Monte Carlo method is frequently used to simulate the cascade of electrons in matter [1–8]. Unfortunately, this software is often not publicly or freely available, or uses outdated physical models. Full Monte Carlo simulations are also slow, often taking hours per simulation run even on modern multi-core CPUs. This has led to the development of several semi-empirical models [3,9,10] to trade accuracy for speed. Verduin et al. [11] demonstrated that Monte Carlo simulations, using semi-empirical scattering models, can be accelerated using GPU hardware. We have designed an open-source software package with first-principle physical models that can run either on GPUs or multi-core CPUs.

From a physical perspective, we make broadly similar assumptions as most Monte Carlo simulators. An electron is treated

* Corresponding author.
E-mail address: L.C.P.M.vanKessel@tudelft.nl (L. van Kessel).

https://doi.org/10.1016/j.softx.2020.100605

as a classical point particle, scattering through the volume of the material in discrete events. The electron is treated as if it is in free flight between such events. We also assume that all electrons can be treated independently. We distinguish three basic types of physical processes: inelastic scattering, elastic scattering and material boundary crossing. The first two take place in the bulk of the material; the latter represents interactions at the interface between two materials. Several models exist in literature to describe the mean free path between scattering events, as well as the detailed scattering behaviour (*e.g.* energy loss, deflection) when an event takes place. By default, we use Penn dielectric function algorithm [12,13] to describe inelastic scattering. Secondary electron generation from the valence band and the corresponding deflection of the primary electron are treated by the method of Mao et al. [14]. Secondary electron generation from inner shells is treated as described by Kieft & Bosch [3] and Verduin [8]. Binding energies of inner shells, as well as their ionization cross sections, are taken from the LLNL Evaluated Electron Data Library [15]. For elastic scattering, we use Mott scattering calculated by ELSEPA [16] for energies > 200 eV, electron–acoustic phonon scattering as described by Schreiber and Fitting [17] for energies $<$ 100 eV, and interpolation in between. When electrons cross a boundary between materials, a simple quantum mechanical step function model [18] is used. In addition to these models, the semi-empirical model by Kieft & Bosch [3] is available in our software. This comprises modified inelastic and boundary crossing models. Situations that rely on the wave properties of electrons, such as diffraction experiments, cannot be described due to the physical approximation of the electron as a point particle. Auger de-excitation and incoherent X-ray scattering are not currently supported, but planned for the near future. Due to the assumption that all electrons are independent, charging effects cannot be described.

The simulator has been designed to be flexible to use. It does not attempt to make any assumptions about the intended use case. For example, it can be used to simulate secondary electron (SE) and backscattered electron (BSE) SEM images, secondary electron yields, or the reflected electron energy distribution. Amongst others, this software package has been used to probe the angle dependence of secondary electron emission for the design of microchannel plates [19]. Possible future work may investigate the influence of surface roughness on electron yield. The software has also been used to understand the relationship between sidewall roughness and measured line-edge roughness in top-down SEM images [20]. This study involved a large parameter sweep which would not have been feasible without the simulator's GPU support.

2. Software description

The Monte Carlo simulator works as follows. At the beginning of an iteration, an electron has a known position, direction and energy. If the electron is in vacuum, it cannot scatter elastically or inelastically, and it is moved to the next intersection with a material boundary in a straight line. If the electron is in a material, the elastic and inelastic scattering processes are probed for random free path lengths. The exact distance of travel to the next event is a random number, typically given by the exponential Lambert–Beer law, with the mean free path depending on the electron's energy. The nearest of the three events (elastic, inelastic or boundary crossing) is chosen. The electron is moved in a straight line to the next event, and the event is performed. The outcome of the event is, in general, random. A material boundary crossing may involve reflection, refraction and/or an energy gain due to the difference in inner potential between the materials. In the semi-empirical model of Kieft & Bosch, an electron may also be absorbed at an interface. A scattering event in the bulk of the material may involve a deflection of the electron or energy loss. Inelastic events may create a secondary electron (SE). If an SE is created, it is added to the list of active electrons and processed in the same way as the primary electron. The procedure mentioned here is repeated until an electron reaches a detector, or its energy drops to an insufficient level to leave the material.

Our Monte Carlo simulator takes three types of input:

- **The geometry**, which is provided as a list of triangles plus the materials on either side. These triangles represent the interfaces between materials in the sample. The shape of the detector is also part of the geometry, as a special "detector" material.
- **The primary electrons**, which are provided as a list where each entry corresponds to one electron.
- **Material file(s)**, one for each material in the sample. These store detailed scattering cross sections describing the scattering behaviour of electrons in each material.

The simulator's output is a list of the position, direction and energy of the detected electrons, as well as tags to identify the corresponding input electron. This list may be post-processed to obtain the desired final output, which could, for example, be an SEM image or an energy spectrum. The detectors are idealized "electron counters" with 100 % efficiency. Realistic detectors, *e.g.* in an electron microscope, have a complex setup-dependent sensitivity to the electron's emission energy and angle, which we do not include (but may be included by the user in a post-processing step). For the convenience of the user, we do include specialized "SE" and "BSE" detectors, which are implemented as simple energy filters at 50 eV.

2.1. Software architecture

The software package consists of two main components: the simulator (*Nebula*) and a material file generation tool (*cstool*, short for "[scattering] cross section tool"). The flow of data is depicted in Fig. 1.

Cstool takes material parameters as a human-readable text file, computes and compiles scattering cross sections for that material and stores the output in a file. Cstool is intended to be run once for each material of interest. It depends on the software package ELSEPA [16], which needs to be compiled and installed by the user. It automatically downloads additional required input from the LLNL Evaluated Electron Data Library [15]. The material files can contain data for multiple physical models. By default, they store data for both the inelastic models of Penn and Kieft & Bosch.

In addition to material files, Nebula takes the geometry and primary electrons as input. The geometry is provided as a list of triangles, represented in text format. The positions of the detectors are part of the geometry. The primary electrons are also input as a list, but the data is stored in a simple binary format to save disk space and speed up creation and loading. A primary electron is defined by its starting position, initial direction of movement, and energy. Each primary electron may also be tagged with two integers. These do not change throughout the electron's lifetime, and are passed on to all SEs that the primary electron creates. For example, these tags can be used to represent the pixel in the SEM image that the electron belongs to. The simulator's output is a list of electrons in the same file format as the primary electrons. It stores the electrons' position, direction and energy at the instant the detector is reached.

The file format of geometry and electron files has been deliberately kept simple with the intention that users will write scripts

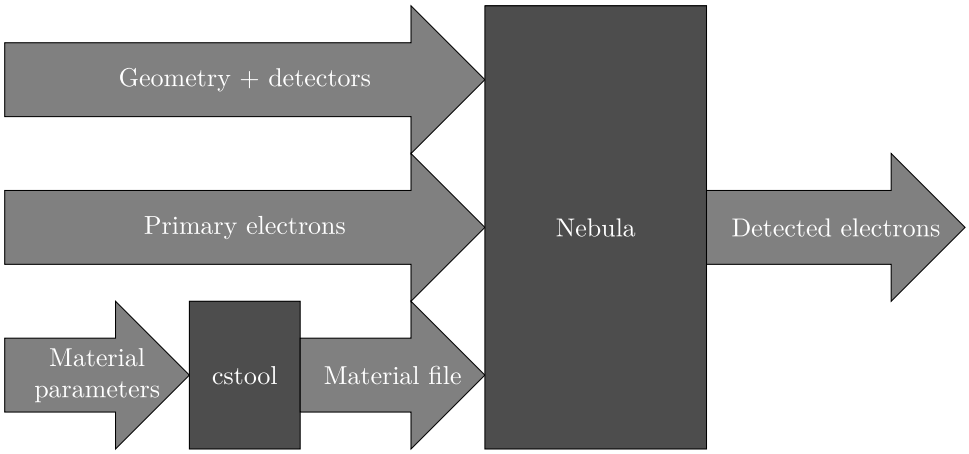

Fig. 1. Illustration of the data flow in the software. Nebula is the main simulator, Cstool compiles scattering cross sections for a material.

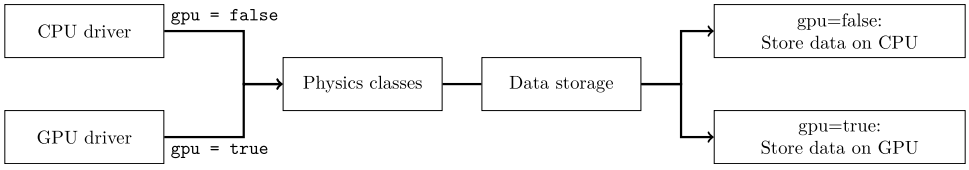

Fig. 2. Illustration of the CPU–GPU split in Nebula. Drivers are responsible for managing the electrons in the simulation. Physics code is shared between CPU and GPU versions. A gpu template parameter is passed to the data storage classes, which copy data to the GPU if necessary.

to generate their own input. Example scripts are provided with the software for the user's convenience.

Parallelization on the GPU is achieved by the method of Verduin et al. [11]. Each thread is assigned a single electron. As dynamic memory allocation is not possible in a CUDA kernel, some "electron slots" must be empty so that SEs can be accommodated when ionization events take place. Meanwhile, the fact that there are different types of scattering events, each with different code to execute, potentially leads to severe warp divergence. Both these issues are solved by introducing a sorting step after the next event was chosen but before it is performed. Electrons which will undergo inelastic events (and may create an SE) are put at the front, empty slots are at the back. This makes an empty slot for the SE easy to find, and warp divergence is reduced because all inelastic, elastic and boundary crossing events are clustered.

Parallelization on the CPU is much simpler than on the GPU. Each thread is assigned a primary electron, and each thread maintains a list of SEs.

We wanted to share common code (such as that used for the physical models, or the collision checking with the geometry) as much as possible between the CPU and GPU versions of the simulator. This is possible thanks to the syntactic similarity between C++ and CUDA. The algorithmic differences in parallelization between CPU and GPU devices complicate matters, however. Our solution is to move the algorithm-specific code into "drivers". These are responsible for managing the electrons: accepting new electrons to be simulated, and extracting electrons that reached a detector. They also query the physical models for free path lengths, move the electrons to their new positions, and call the physical models to perform the events. The GPU driver is also responsible for the sorting step. The drivers pass a template parameter specifying whether the code is to be run on a GPU to the classes that handle the scattering physics. The physics classes do not use this parameter directly, but they pass it to utility classes for storing large amounts of data. If the simulation runs on a GPU, this data needs to be explicitly copied to the device. This organization is depicted in Fig. 2.

Physical models are chosen at compile time and compiled into the executable. This is also achieved by means of the template system in C++. The driver takes the enabled physical models as a variadic template parameter pack. We opted for this design because it offers great flexibility for adding more physical models, without unused code cluttering the executable.

2.2. Software functionalities

The following key features set Nebula apart from other simulators:

- Nebula is fast, able to run on the GPU for speed or on multi-core CPUs if there is no GPU available.
- Nebula is based on first-principle physical models. Some other simulators employ empirical models to gain speed [3,9,21].
- Two choices of physical models are included: a set of fully physical models and the semi-empirical models of Kieft and Bosch [3].

Nebula has been used for the following purposes, though its use is not limited to these:

- To simulate SE and BSE SEM images [20,22].
- To obtain the positions of bond-breaking events in electron-beam lithography resists.
- To simulate electron yields as a function of incidence angle, energy and surface roughness [19]; as well as the emission energy and angle [23].

3. Illustrative examples

A simple use case is presented in Fig. 3. Fig. 3a shows a ring-like silicon structure on top of a silicon substrate. There is a "detector box" around this structure to capture all emitted

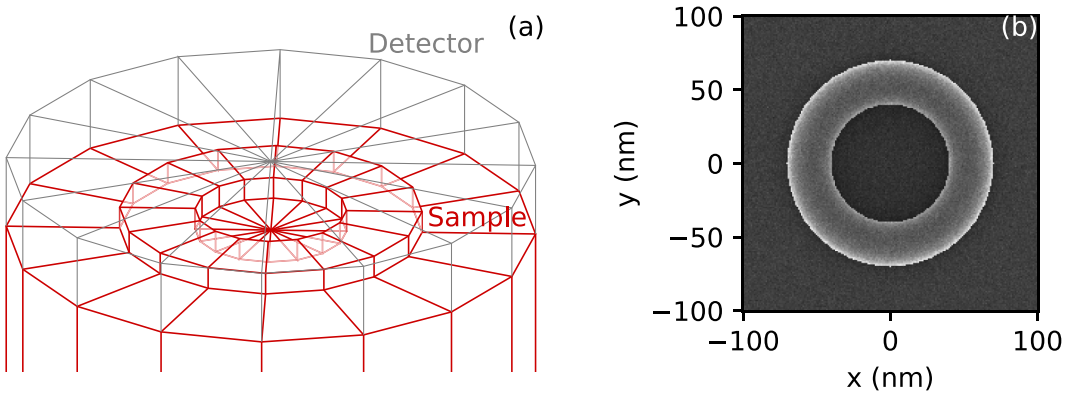

Fig. 3. (a) Illustration of a simple geometry, including the location of detectors. (b) Corresponding SE SEM image with a 1 keV beam.

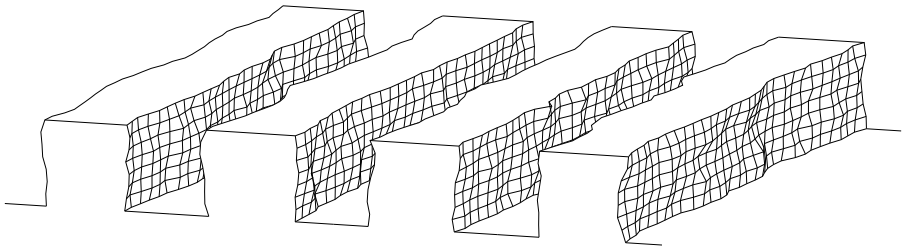

Fig. 4. Illustration of the geometry studied in Fig. 5.

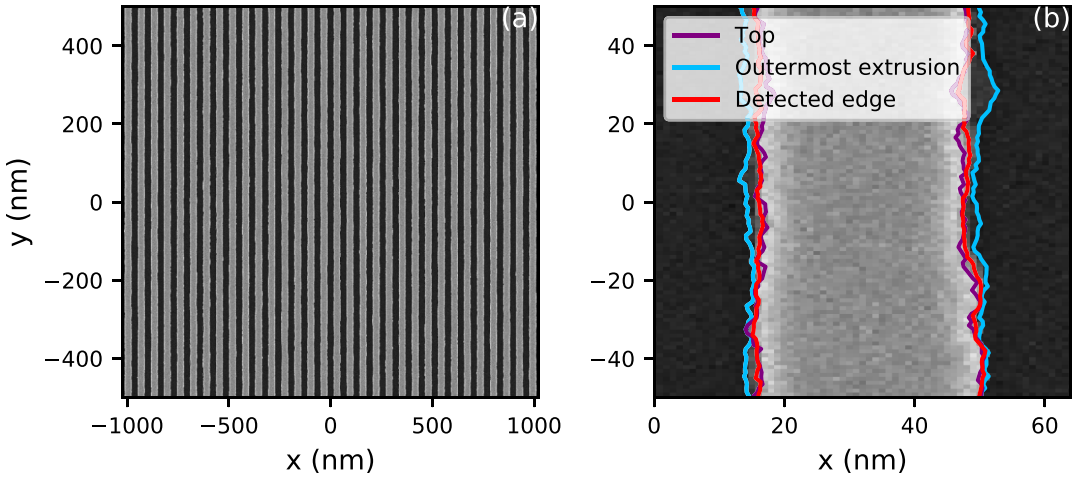

Fig. 5. Simulated SEM image of a dense line-space pattern. (a) shows the entire image, (b) shows a small section. Also shown in figure (b) are three contours: the true shape of the top slice of the line, the outermost extrusion of the full height of the line, and the edge detected by a contouring algorithm.

electrons. The primary electrons from the SEM start just below the main detector plane, above the sample surface. The corresponding SE SEM image can be seen in Fig. 3b. Only < 50 eV electrons are counted in the image; the detector is an idealized "electron counter".

We also present a use case from the semiconductor world, previously published in [20]. The situation is illustrated in Fig. 4. We are interested in a sample with a dense pattern of lines and spaces. The lines are made of PMMA, the substrate is silicon. The lines have rough sidewalls. The sample is observed by a top-down SEM.

This simulation run features more than 10 million triangles making up the geometry and more than 200 million primary electrons at 300 eV landing energy. It took less than 10 min on an Nvidia GTX1080. The resulting SEM image can be seen in Fig. 5a. Fig. 5b shows a small section, indicating the true position of the top slice of the rough lines, the true position of the outermost extrusion of the line over the full height, and a contour detected in the SEM image. In the full study [20], we have shown that the contour detected from the SEM image follows the outermost extrusion of the line if the line is isolated (has no neighbours). In this case of dense lines and spaces, however, some electrons do not escape the lower regions between the lines. As a result, the contour detected from the SEM image is in between the top slice and the outermost extrusion.

4. Impact

The speed of the simulator makes it possible to perform large systematic studies that were not previously feasible. Our method

Table 1
Speed comparison of the GPU simulation method compared to CPU simulations. We repeat the numbers of Verduin et al. [11], who replicated the physical models of Kieft et al. [3]. Their simulator ran on an Nvidia GTX480, the single-threaded CPU simulator of Kieft et al. [3] ran on an Intel Xeon X5650. We also compared Nebula to the simulator reported by Zou et al. [24]. We used the same physical models, sample (Au lines on a Si substrate), and electron beam settings. We used an Nvidia GTX1080, Zou et al. used an unspecified 10-core CPU.

Beam energy	Single-thread CPU [3]	GPU [11]	Speedup
5 keV	2w5d12h	32 m	894×
3 keV	1w4d20h	22 m	796×
1 keV	3d11h	10 m	538×
800 eV	2d17h	8 m	530×
500 eV	1d10h	5 m	472×
300 eV	15h16m	3 m	387×
Beam energy	10-core CPU [24]	Nebula GPU	Speedup
3 keV	2 h	90 s	80×

of running the simulator on a GPU is similar to that of Verduin et al. [11]. They report a considerable speedup when comparing their simulator to a single-threaded version of the simulator by Kieft et al. [3]. We found a similar speed difference between Nebula and the 10-core in-house simulator reported by Zou et al. [24]. Full details are provided in Table 1.

The simulations presented in [20], which were also discussed in the second illustrative example, took several days to complete with our simulator. Several additional months were spent on simulations that were required for the investigation but did not end up in the final publication. It would have been impractical to perform this study with a slower simulator. Previous authors [25,26] investigated similar situations, but were limited to small parameter sweeps. Our faster simulator allowed us to perform a much more thorough investigation. We were able to build a general understanding of the simple situation where lines are isolated and show what changes when a dense pattern of lines and spaces is used.

5. Conclusions

Nebula is a fast and accurate simulator of electron–matter interaction. It uses first-principle physics, can run on GPUs, but also works on CPUs when there is no GPU available. It has been used for a variety of different applications in the fields of electron microscopy, electron beam induced deposition, electron beam lithography, and electron detection, both by academia and industry. Functionality will continue to be implemented as the software is used, and physical models will be updated and added as continuing research provides new insights into the field of electron scattering.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We would like to thank Thomas Verduin, Pieter Kruit, Erik Kieft, Annemarie Theulings and Kerim Arat for interesting discussions. We would also like to thank Ruben Maas, Willem van Mierlo, Benoit Gaury, Ton Kiers, Thomas Huisman, Janina Löffler, Lucas Audebert and Guido Rademaker for valuable user feedback.

References

[1] Li YG, Mao SF, Li HM, Xiao SM, Ding ZJ. Monte Carlo simulation study of scanning electron microscopy images of rough surfaces. J Appl Phys 2008;104(6):064901. http://dx.doi.org/10.1063/1.2977745.
[2] Babin S, Borisov S, Ivanchikov A, Ruzavin I. CHARIOT: software tool for modeling SEM signal and e-beam lithography. Physics Procedia 2008;1(1):305–13. http://dx.doi.org/10.1016/j.phpro.2008.07.110.
[3] Kieft E, Bosch E. Refinement of Monte Carlo simulations of electron–specimen interaction in low-voltage SEM. J Phys D: Appl Phys 2008;41(21):215310. http://dx.doi.org/10.1088/0022-3727/41/21/215310.
[4] Dapor M, Ciappa M, Fichtner W. Monte Carlo modeling in the low-energy domain of the secondary electron emission of polymethylmethacrylate for critical-dimension scanning electron microscopy. J. Micro/Nanolithogr. MEMS MOEMS 2010;9(2):023001. http://dx.doi.org/10.1117/1.3373517.
[5] Salvat F, Fernández-Varea JM, Sempau J. PENELOPE-2011: A Code System for Monte Carlo Simulation of Electron and Photon Transport. OECD, Nuclear Energy Agency, Organization for Economic Co-operation and Development; 2011, URL https://www.oecd-nea.org/science/docs/2011/nsc-doc2011-5.pdf.
[6] Demers H, Poirier-Demers N, Couture AR, Joly D, Guilmain M, De Jonge N, Drouin D. Three-dimensional electron microscopy simulation with the CASINO Monte Carlo software. Scanning 2011;33(3):135–46. http://dx.doi.org/10.1002/sca.20262.
[7] Villarrubia JS, Vladár AE, Ming B, Kline RJ, Sunday DF, Chawla JS, List S. Scanning electron microscope measurement of width and shape of 10 nm patterned lines using a JMONSEL-modeled library. Ultramicroscopy 2015;154:15–28. http://dx.doi.org/10.1016/j.ultramic.2015.01.004.
[8] Verduin T. Quantum Noise Effects in e-Beam Lithography and Metrology (Ph.D. thesis), Technische Universiteit Delft; 2017, http://dx.doi.org/10.4233/uuid:f214f594-a21f-4318-9f29-9776d60ab06c.
[9] Cizmar P, Vladár AE, Ming B, Postek MT. Simulated SEM images for resolution measurement. Scanning 2008;30(5):381–91. http://dx.doi.org/10.1002/sca.20120.
[10] Babin S, Borisov SS, Trifonenkov VP. Fast analytical modeling of SEM images at a high level of accuracy. Proc. SPIE 2015;9424:94240I. http://dx.doi.org/10.1117/12.2086072.
[11] Verduin T, Lokhorst SR, Hagen CW. GPU accelerated Monte-Carlo simulation of SEM images for metrology. Proc. SPIE 2016;9778:97780D. http://dx.doi.org/10.1117/12.2219160.
[12] Penn DR. Electron mean-free-path calculations using a model dielectric function. Phys Rev B 1987;35(2):482–6. http://dx.doi.org/10.1103/PhysRevB.35.482.
[13] Shinotsuka H, Tanuma S, Powell CJ, Penn DR. Calculations of electron stopping powers for 41 elemental solids over the 50 eV to 30 keV range with the full Penn algorithm. Nucl Instrum Methods Phys Res 2012;270(1):75–92. http://dx.doi.org/10.1016/j.nimb.2011.09.016.
[14] Mao SF, Li YG, Zeng RG, Ding ZJ. Electron inelastic scattering and secondary electron emission calculated without the single pole approximation. J Appl Phys 2008;104(11):114907. http://dx.doi.org/10.1063/1.3033564.
[15] Perkins ST, Cullen DE, Seltzer SM. Tables and graphs of electron-interaction cross-sections from 10 eV to 100 GeV derived from the LLNL evaluated electron data library (EEDL), Z=1–100. UCRL-50400 1991;31:21–4. http://dx.doi.org/10.2172/5691165.
[16] Salvat F, Jablonski A, Powell CJ. ELSEPA—Dirac partial-wave calculation of elastic scattering of electrons and positrons by atoms, positive ions and molecules. Comput Phys Comm 2005;165(2):157–90. http://dx.doi.org/10.1016/j.cpc.2004.09.006.
[17] Schreiber E, Fitting H-J. Monte Carlo simulation of secondary electron emission from the insulator SiO2. J Electron Spectrosc Relat Phenom 2002;124(1):25–37. http://dx.doi.org/10.1016/S0368-2048(01)00368-1.
[18] Shimizu R, Ding Z-J. Monte Carlo modelling of electron-solid interactions. Rep Progr Phys 1992;55(4):487–531. http://dx.doi.org/10.1088/0034-4885/55/4/002.
[19] Löffler J, Thomet J, Belhaj M, van Kessel L, Hagen CW, Ballif C, Wyrsch N. Monte Carlo modeling of electron multiplication in amorphous silicon based microchannel plates. In: 2019 IEEE Nuclear Science Symposium and Medical Imaging Conference. NSS/MIC, 2019, p. 1–6. http://dx.doi.org/10.1109/NSS/MIC42101.2019.9059971.
[20] van Kessel L, Huisman T, Hagen CW. Understanding the influence of three-dimensional sidewall roughness on observed line-edge roughness in scanning electron microscopy images. J. Micro/Nanolithogr. MEMS MOEMS 2020;19(3):034002. http://dx.doi.org/10.1117/1.JMM.19.3.034002.
[21] Arat KT, Bolten J, Klimpel T, Unal N. Electric fields in scanning electron microscopy simulations. Proc. SPIE 2016;9778:97780C. http://dx.doi.org/10.1117/12.2219182.
[22] Pu L, Wang T, Huisman TJ, Maas R, Goosen M, Dillen H, Leray P, Fang W. Analyze line roughness sources using power spectral density (PSD). Proc. SPIE 2019;10959:109592W. http://dx.doi.org/10.1117/12.2516570.

[23] Kumar TP R, Hari S, Damodaran KK, Ingólfsson O, Hagen CW. Electron beam induced deposition of silacyclohexane and dichlorosilacyclohexane: the role of dissociative ionization and dissociative electron attachment in the deposition process. Beilstein J. Nanotechnol. 2017;8(1):2376–88. http://dx.doi.org/10.3762/bjnano.8.237.

[24] Zou YB, Khan MSS, Li HM, Li YG, Li W, Gao ST, Liu LS, Ding ZJ. Use of model-based library in critical dimension measurement by CD-SEM. Measur.: J. Int. Measur. Confed. 2018;123:150–62. http://dx.doi.org/10.1016/j.measurement.2018.02.069.

[25] Lawson R, Henderson C. Investigating SEM metrology effects using a detailed SEM simulation and stochastic resist model. Proc. SPIE 2015;9424:94240K. http://dx.doi.org/10.1117/12.2086051.

[26] Verduin T, Lokhorst S, Kruit P, Hagen C. The effect of sidewall roughness on line edge roughness in top-down scanning electron microscopy images. Proc. SPIE 2015;9424:942405. http://dx.doi.org/10.1117/12.2085768.