Monte Carlo modeling of secondary electron imaging in three dimensions

John S. Villarrubia,[a] Nicholas W. M. Ritchie,[a] and Jeremiah R. Lowney[b]
[a]National Institute of Standards and Technology,† Gaithersburg, MD, 20899
[b]Under contract to NIST, Leonardtown, MD 20650

ABSTRACT

Measurements of critical dimensions (CDs), roughness, and other dimensional aspects of semiconductor electronics products rely upon secondary electron (SE) images in the scanning electron microscope (SEM). These images are subject to artifacts at the nanometer size scale that is relevant for many of these measurements. The most accurate measurements for this reason depend upon models of the probe-sample interaction in order to perform corrections. MONSEL, a Monte Carlo simulator intended primarily for CD metrology, has been providing the necessary modeling. However, restrictions on the permitted sample shapes are increasingly constraining as the industry's measurement needs evolve towards inherently 3-dimensional structures. We report here results of a collaborative project, in which the MONSEL physics has been combined with the 3D capabilities of NISTMonte, another NIST Monte Carlo simulator that was previously used principally to model higher energy electrons and x-rays. Results from the new simulator agree very closely with the original MONSEL for samples within the repertoire of both codes. The new code's predicted SE yield variation with angle of incidence agrees well with preexisting measurements for light, medium, and heavy elements. Capabilities of the new code are demonstrated on a model of a FinFET transistor.

Keywords: constructive solid geometry, critical dimension metrology, modeling, linewidth metrology, Monte Carlo methods, scanning electron microscopy (SEM), Monte Carlo modeling, secondary electron image

1. INTRODUCTION

Non-ideal instrument behavior is often important at the nanometer size scales relevant for the smallest features on electronic devices. It is becoming increasingly common for instrument models to play significant roles in the semiconductor electronics manufacturing process. So, for example, models of lithography tool transfer functions and etch behavior inform how feature sizes on the mask—and assist features near them—are to be designed to achieve the desired final result. Optical scatterometry, now a widely used metrology technique, employs optical and material models to convert non intuitive optical scattering patterns into the desired sample dimensions.

Modeling to correct measurement artifacts has also been employed for scanning-electron-microscope (SEM) metrology, for backscatter images, transmission, x-ray microanalysis, and dimensional metrology by using secondary electron (SE) images.[1-8] The model calculations are typically Monte Carlo simulations. Calculation of a linescan (e.g., 200 pixels with 10 000 simulated trajectories at each) may require 10 minutes to an hour on a typical desktop computer. This might once have been considered an obstacle to the use of model-based measurements for process-control, where throughput is critical, but library-based methods[2-7] can reduce the model-evaluation time to seconds during production, when time is critical, at the cost of some pre-production preparation time to build the library. The model-based measurement can provide widths and sidewall angles of lines, both of which have compared favorably to subsequent cross-sectional or atomic force microscope measurements on the same lines.[2,4-7] These developments mean that for best accuracy, modeling should be as much an integral part of SEM measurements as it is in optical scatterometry. Its use may be essential since the requirements for metrology tool accuracy and precision[9] are tighter than the native spatial resolution of the tool (i.e., the size of beam spread effects, due both to finite beam size and scattering within the sample). To the extent that a model will "plug in" to a metrology SEM, better models will translate to better measurements.

Metrology, Inspection, and Process Control for Microlithography XXI, edited by Chas N. Archie
Proc. of SPIE Vol. 6518, 65180K, (2007) · 0277-786X/07/$18 · doi: 10.1117/12.712353

For our part of the earlier work referenced above, we built model libraries by using the MONSEL simulator. MONSEL was developed at the National Institute of Standards and Technology (NIST) in the mid 1990s.[10-12] It was an outgrowth of earlier work on electron trajectory modeling by Myklebust et al.[13] MONSEL was the first program to model low energy SE generation by specific scattering processes (as opposed to modeling only the higher energy backscattered electrons or SE generation by proportionality to energy loss). It is a series of programs, each written for a fixed class of sample shapes. Each class consists of a basic shape characterized by a set of parameters. A typical example is in Fig. 1, which shows a sample consisting of three layers on top of which are three lines. The lines are infinite in extent and uniform in cross-section. Thus the sample is 3-dimensional (3D), as suggested by the perspective lines shown on the middle line in the figure. The lines are separated by a pitch, *p*, and are generally trapezoidal in shape, though they may have rounded corners and a "jog" in the sidewall. Adjustable parameters include the pitch, width, sidewall angle, radius of the upper corners, jog height, and jog width. As this example suggests the simulator was intended primarily for applications in linewidth metrology, although one of the other programs in the series modeled samples with rectangular pillars or holes in a 2×2 array.

There are a growing number of industry metrology applications for which the restrictions on sample geometry are too limiting. For example, although the sidewall jog could be used to simulate a "foot" on a line by placing it suitably low, this foot could not be realistically rounded or sloped. Vias, contact holes, and parts of new devices (e.g., FinFETs) have rounded or complicated 3D confined areas that provide opportunities for escaping electrons to reenter the sample. These may either reduce the measured intensity or increase it (if the ones recaptured are outnumbered by the SE they create upon reentry). The original MONSEL series requires each layer to be uniform in composition, but there are industrial samples (e.g., a Cu-filled Damascene trench in an interconnect layer) in which two or more materials share the same height coordinate. Increasingly automated critical dimension- (CD-) SEMs mark the position of a feature edge along its entire length, following lines as they turn corners or come to an end. Strictly, the intensity that corresponds to the edge of such a feature will differ at corners or line ends. These locations are inherently 3D. Lines in real materials are not strictly uniform in cross section, but exhibit roughness along the edges. There is increasing interest in measuring line edge roughness—and correspondingly the necessity to model its effect upon SEM images.

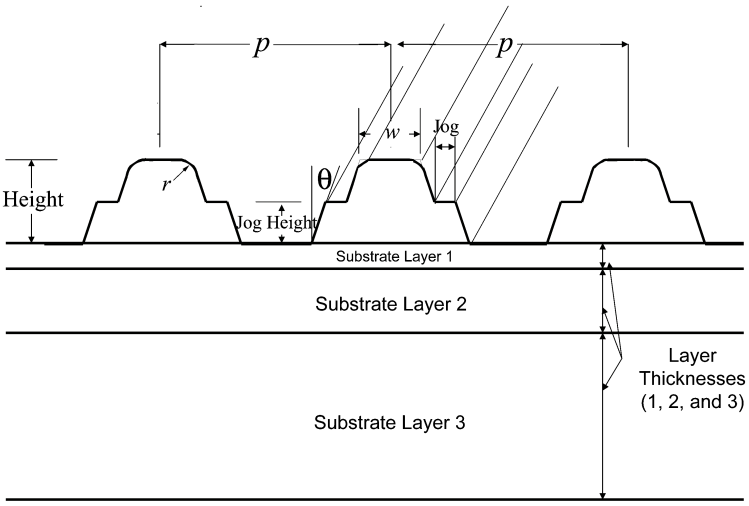

FIG. 1. A typical MONSEL sample geometry.

We have therefore undertaken a project to update the MONSEL simulator. The project extends to fully general 3-D shapes the sample geometries that can be modeled, and it will eventually improve the underlying physics of the models. The first part of the project is essentially complete, and is the subject of this report. Revisiting the model physics is just beginning. There may be one or two places where we briefly discuss improvements we are making to the physics, but this is not the emphasis in this report. In fact, even when we have developed improved models these have always been developed as options, with the original versions also implemented in the new code. Because of this the new code can be compared to the previous one by using identical models and samples within both their repertoires. In this way it was insured that the extensive revisions to the code did not introduce unintended changes. In Sec. 2 we describe the content and organization of the new 3D simulator. In Sec. 3 we compare simulations by the new code with those from the original version and with some experimental data. We also present a representative example of the new version's improved capabilities.

2. DESCRIPTION OF THE 3-DIMENSIONAL SIMULATOR

The simulator was written in the Java programming language because NIST has a preexisting open source Java program, NISTMonte,[14] that performs Monte Carlo simulations of electron trajectories for the purpose of modeling backscattered

and x-ray production for microanalysis. NISTMonte already had the required 3D capability. However, it did not have the necessary physical models for SE generation. The new simulator was therefore written as a Java package (named MONSEL after the original series) that interfaces with NISTMonte. The MONSEL package supplies the missing SE physics. In many cases it was necessary for MONSEL to extend existing NISTMonte classes to provide needed new capabilities. In a small number of cases it was necessary to modify existing NISTMonte classes themselves to generalize them, permitting the new capabilities without losing the old ones. In the discussion that follows we use of the names of code objects or occasional short sections of code. A plain-English example here may benefit readers who are unfamiliar with any object-oriented programming language. The line, `double calculateEnergyLoss(double len, Electron pe);`, describes a method (i.e., function) named `calculateEnergyLoss` that takes two input arguments. The first, named `len`, is double precision. The second, named `pe`, is of type `Electron`. (The reader can safely assume that `Electron` in this context is an object defined elsewhere in the code, if not explicitly in this paper, and presumably contains the position, energy, direction of motion, and other relevant properties of an electron.) The function returns a double precision number, presumably the energy lost by this electron traversing a length `len`. These short references should be understood as pseudocode, with possible name changes or omission of less important details to simplify the presentation.

2.1 The phenomena to be modeled

As mentioned above, the physics remains largely unchanged from the original MONSEL series. An overview is given here for the sake of completeness, and because the nature of the phenomena to be modeled guides the design of the simulator. However, many of the details are left to the references.

The simulator follows electrons as they enter the material, scatter, lose energy, and eventually (some of them) exit and find their ways to a detector. The approach is to explicitly model a small number of the most important phenomena, and then capture the effect of the larger number of less important or less well-known phenomena in an average way. The explicitly treated phenomena are of two types: 1) scattering between tracked electrons and occupants of the bulk material (e.g., nuclei, other electrons, plasmon or other collective excitations) and 2) scattering of electrons at boundaries between materials. The effect of the remaining phenomena is summarized by their effect on the energy of the tracked electron in a "continuous slowing down" approximation. As the name implies, instead of random and discrete energy losses, the effect of these events is modeled by a continuous energy loss equal to their expected average rate.

2.1.1 Model for scattering events in the bulk

These scattering events are described by an exponential probability distribution

$$P(x) = \gamma e^{-\gamma x} \tag{1}$$

where γ is the total scattering rate and $P(x)$ is the probability density function for scattering. That is, $P(x)dx$ is the probability that a scattering event will occur when the primary particle (always an electron for us) has traversed a distance between x and $x + dx$. The average distance the electron moves before scattering (the mean free path, λ_{MFP}) is then

$$\lambda_{\mathrm{MFP}} = \int_0^\infty x P(x) dx = \frac{1}{\gamma}, \tag{2}$$

so the scattering rate is just the reciprocal of the mean free path. The probability (with subscript ns indicating "no scattering") that the electron will survive unscattered for a distance λ is

$$P_{\mathrm{ns}}(\lambda) = 1 - \int_0^\lambda \gamma e^{-\gamma x} dx = e^{-\gamma\lambda}. \tag{3}$$

The random variable λ is called the free path. We can use a uniform random number generator to simulate this distribution by setting this probability equal to R, a random number uniformly distributed in the interval $(0, 1)$. Solving for the free path in terms of the random number then yields

$$\lambda = -\ln(R)/\gamma. \tag{4}$$

When more than one scattering mechanism is at work in a material the total rate is the sum of the individual rates, γ_i. The probability that a given mechanism was responsible is just its scatter rate divided by the total scatter rate. The several scatter mechanisms can be assigned adjoining intervals of size γ_i/γ so that in combination they fill the interval from 0 to 1. A second random number between 0 and 1 is then drawn. It will lie within the interval assigned to one of the scatter mechanisms, and that mechanism is then assigned responsibility for this scattering event.

As the foregoing discussion makes clear, the electron's free path on any particular leg of its trajectory and the assignment of responsibility for scattering to one of the mechanisms in the material depend upon a single parameter, the scattering rate, for each scattering mechanism. This rate is related to the total scattering cross section via $\gamma_i = N_A \rho \sigma_i^T / A$ where N_A is Avogadro's number, ρ is the material density, σ_i^T is the total cross section for this scattering mechanism, and A is the mass of one mole of the material. The total cross section generally depends upon the electron's energy, the properties of the medium through which the electron moves, and possibly (for anisotropic materials) the direction of the electron's motion relative to the material orientation. The particular form of this scattering rate function differs from one scattering mechanism to another.

Once responsibility for scattering is assigned, the actual scattering event must be simulated. The effects of a general scattering event that are taken into account in MONSEL are these: (1) It may change the energy of the primary electron. (2) It may change the direction of motion of the primary electron. (3) It may generate an SE with some energy and direction of motion. All of these effects are statistical in nature. They are specified by probability distribution functions that are related to the differential scattering cross sections. Once again, the details are mechanism-specific. In general, however, MONSEL interfaces to a scattering model through these two functions, a `freePath()` function that returns the distance traversed before a scattering event occurs and a `scatter()` function that simulates the actual scattering event. New mechanisms can be added by specifying these two functions.

Three such scattering mechanisms are specified in the current implementation. These are particular models for elastic scattering of an electron from a nucleus, impact ionization, in which the primary electron transfers energy to a lightly bound electron in the material, which thereby becomes an SE, and a plasmon-mediated mechanism, in which the primary electron excites a plasmon excitation that possibly subsequently decays by generation of an SE.

Mean free paths (reciprocals of the scattering rates) in Si for the three scattering mechanisms are shown in Fig. 2 for the energy range of interest for most CD metrology. Elastic scattering is treated by using the Mott cross sections,[15] estimated with the fitting method of Browning.[15-17] The effect of impact ionization of *tightly* bound electrons (e.g., core electrons) on the energy of the primary electron is included in the slowing down model (to be discussed in Sec. 2.1.3) but its effect on SE generation is ignored. The cross sections for such processes are typically low, and they therefore do not contribute significantly to SE yields. Impact ionization of lightly bound electrons are approximated by the Möller cross sections,[18] which are strictly accurate only for completely free electrons. If SE of all energies are included, the Möller cross section approaches infinity. However, any SE with energy less than the work function cannot escape the sample to be detected. The work function is therefore a natural low energy cutoff. (The Fig. 2 Möller curve is the mean free path between collisions that produce these relevant electrons, and this is the reason the curve turns up at the lowest energies. When the primary electron energy approaches the cutoff value, the scattering rate must go to zero and the mean free path therefore to

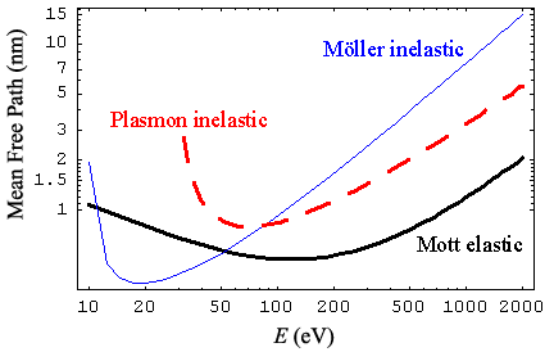

FIG. 2. Mean free paths for three scattering mechanisms.

infinity.) Plasmons are treated in the plasmon-pole approximation, with a high energy cutoff determined by the onset of Landau damping.[12] Plasmons are responsible for a very prominent peak in the energy loss spectra of many materials, so it is known that primary electrons excite the plasmons. However, some coincidence studies have failed to observe significant numbers of resulting SE in some materials, like Si and C.[19,20] To allow for the possibility that plasmon excitations may produce fewer than the expected number of secondaries, our implementation includes an efficiency parameter between 0 and 1. Values less than 1 proportionately reduce the rate of SE generation by this mechanism.

2.1.2 Barrier scattering model

Differences in an electron's potential energy in different materials result in a potential energy step at the boundary. The effect of the step on electrons is analogous to refraction of light at an interface. The electron may transmit through the boundary if $E\cos(\theta)^2 > \Delta U$ with E the electron's kinetic energy, θ the angle of incidence ($0°$ for normal incidence) and ΔU the difference between the potential energy in the new region compared to the present one. If the electron transmits, its kinetic energy changes by ΔU with all of the corresponding momentum change occurring in the component normal to the interface. In general this means a change in the electron's direction of motion (refraction). If the electron does not transmit, it undergoes total internal reflection—the sign of its momentum component normal to the interface is reversed.

To generalize, a barrier scattering model requires input of the electron's properties (its energy and direction of motion), the properties of the materials on the near side of the boundary and on the far side, and the orientation of the boundary (e.g., its normal vector). A model-specific calculation is then performed, the output of which is a new direction and energy for the electron and a determination of whether the electron has transmitted or reflected.

Either a quantum mechanical or a classical model can fit within this general framework. Currently, we have implemented a classical one. The original MONSEL series used a less accurate simple cutoff at the work function, a model that we have also implemented to facilitate comparisons.

2.1.3 Slowing down model

A general slowing down model is simply a function that returns an amount of energy lost for an electron that traverses a given distance. The required inputs are the material properties, the electron's energy, and possibly (for anisotropic materials) its direction.

In the original MONSEL series the energy loss implementation is based upon this energy loss expression:

$$\frac{dE}{ds} = -\alpha\frac{\rho}{E}\sum_i \frac{c_i Z_i}{A_i}\ln\left[1.166\left(\frac{E}{J_i}+k_i\right)\right] \qquad (5)$$

This is Joy and Luo's generalization[21] of Bethe's[22,23] stopping power formula for lower energies. E is the electron's kinetic energy, s is displacement along its path, ρ is the material density, i indexes the elemental constituents of the material, c_i, Z_i, A_i, and J_i are respectively the weight fraction, atomic number, molar mass, and average ionization energy of the i^{th} constituent, α is a proportionality constant (approximately 2.02×10^{-31} J^{2}m^2 in SI units) and k_i is a dimensionless constant, approximately 0.85 but material dependent, tabulated for several elements by Joy and Luo. Since it is frequently necessary to model materials with elements not in their table, an alternative method of estimating k_i is needed. For this

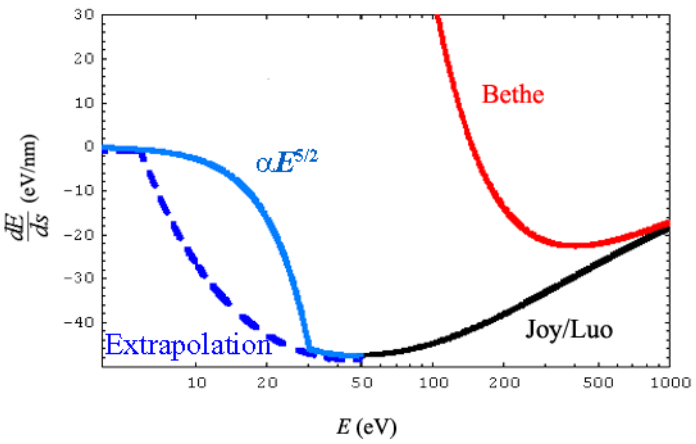

FIG. 3. Stopping power curves in Si. The Joy/Luo expression (Eq. 5) is shown solid above 50 eV, dashed where extrapolated below.

purpose we use the ansatz proposed by Lowney,[12] who observed that the tabulated values were all such that dE/ds approached 0 for $E = \varphi + 1\ \mathrm{eV}$, with φ the material work function. Making this a requirement implies that $k_i = 0.8576 - (\varphi + 1\ \mathrm{eV})/J_i$. The fact that this stopping power reaches 0 at $\varphi + 1\ \mathrm{eV}$ means an electron with energy below this has infinite range, an unphysical result. A small additional energy loss (the "residual energy loss") was therefore added to the loss specified by Eq. 5. The size of this loss is a fitting parameter. Its value was adjusted to match the total yield to a measured yield on a flat sample at a single incident energy. It is typically 1 eV/nm or less. Stopping power curves in the low energy regime are shown in Fig. 3. Equation 5 is shown as a solid line labeled "Joy/Luo" down to 50 eV, the limit claimed by its authors. It is shown dashed where extrapolated below 50 eV.

Step lengths, Δs, are usually quite short. Mean free paths at the relevant energies in most solids are a fraction of a nanometer (Fig. 2). For this reason it is often sufficiently accurate to estimate the energy loss as $\Delta s\, dE/ds$, with dE/ds evaluated at the electron's initial energy. Occasionally, however, there is an unusually long step or the stopping power becomes so large and the electron's energy so small that an appreciable fraction of the energy is lost within a single step. In these cases, the fact that stopping power is changing as the energy changes becomes important. If the energy loss within a step would be greater than 10 % of the electron's energy, we divide the single step into smaller lengths for better accuracy.

The new simulator duplicates this stopping power implementation for the purpose of comparison, but we believe this likely overestimates the stopping power at low energies. We therefore also implement the alternative low energy form (continuous curve below 50 eV) based upon an $E^{5/2}$ stopping power estimate in the low energy limit by Nieminen.[24]

2.2 Building blocks

The process of porting the original MONSEL model into a modern object-oriented language involved definition of a number of program objects. This section gives an overview of the important objects, which serve as building blocks out of which the simulation is constructed. The next section will describe how these building blocks are combined to produce the simulation.

2.2.1 Material description

NISTMonte has an Element class to represent elements. A collection of elements along with their relative amounts, expressed either as atomic or weight fractions, is stored in the `Composition` class. A `Material` is defined to be a `Composition` together with a density. This definition of a material was sufficient to simulate elastic scattering of higher energy electrons (for which barrier scattering is negligible) and x-ray production. SE models require a number of additional material parameters, both because the electron generation models depend upon them and because SE are typically low energy for which barrier scattering is no longer insignificant. MONSEL therefore has an `SEmaterial` class that extends the NISTMonte `Material` class. The extension includes provision for storing the material work function, the energy of its plasmon, a parameter related to the size of the potential energy step at a material-vacuum boundary, and a discrete representation of the density of occupied electronic states. The density of states is represented by a collection of energies measured with respect to the vacuum level and the corresponding density (electrons/m^3) at each of these energies. This mechanism can be used to represent the density of states crudely (e.g., as a single pair of numbers, one representing the total density of all the lightly bound electrons and the other representing a kind of average binding energy) or with as much precision as one might desire (by digitizing and storing an entire density-of-states curve).

2.2.2 Shapes

Geometry is handled by using constructive solid geometry (CSG). In this scheme complex shapes are built up from 3D shape primitives by using set operations (union, intersection, difference, etc.). In NISTMonte a shape is an object that implements a `Shape` interface, which specifies two methods, `contains(double[] pos)` and `getFirstIntersection(double[] pos0, double[] pos1)`. The first returns `true` if its argument lies inside the shape and `false` otherwise. The second returns a number representing the distance to the position of the boundary intersection nearest pos0 in the direction of pos1. To accommodate boundary scattering models it is necessary to additionally know the direction of the outward pointing normal vector of the shape at this intersection. MONSEL therefore has a `NormalShape` interface that extends the Shape interface by provision of an additional method to return this information.

Primitive shapes currently implemented in NISTMonte include spheres, cylinders, and a `MultiPlaneShape`. For this purpose a `Plane` is specified by a point that lies within the plane and a normal vector. The plane divides space in two. All points on the side pointed to by the normal vector are deemed to be outside. The rest are inside. A `MultiPlaneShape` contains a collection of planes, and a point is deemed to be inside the shape if it is inside all of the planes that comprise the shape. A `MultiPlaneShape` can be used to approximate any convex object to any desired degree of accuracy, simply by making the number of planes large enough, in much the same way that a curve in 2D can be approximated by a large enough series of small connected line segments. Concave objects can be divided into convex pieces, each of which is then described by such a shape. MONSEL extends each of NISTMonte's primitive shapes to make a `NormalShape` counterpart. In addition to the primitive shapes, shapes formed by union, intersection, difference, and shape complement (reversing the definitions of inside and outside) are defined.

2.2.3 Regions and subregions

The sample and chamber are defined by specifying a number of Regions and subregions (Fig. 4). A Region associates a Shape and a Material. The region may also wholly contain other Regions (subregions) with their own shapes and materials. The material at a point is deemed to be the material of the innermost region that contains that point.

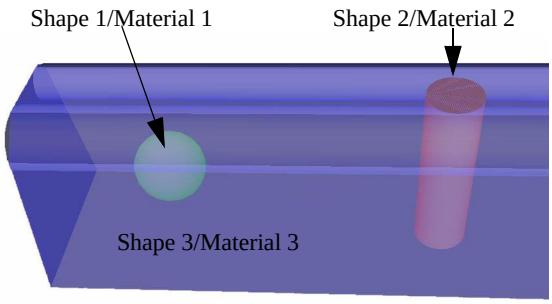

FIG. 4. Examples of a region and subregions. Each region is specified by its shape and material. Region 3 is the union of a `Multiplane` shape with cylinders to provide rounding at two corners. It has two subregions, one spherical and one cylindrical, representing voids or imbedded material.

2.2.4 Electrons and the electron gun

The Electron class contains fields for the electron's kinetic energy, its position, its direction of motion, and its current Region. (It has fields for additional quantities mainly useful for bookkeeping or trajectory plotting purposes, like step counts, its previous position and previous Region, etc.) An `ElectronGun` interface specifies functions associated with making new electrons to initiate a trajectory. The interface specifies methods for setting the center of the electron beam and the energy of the electrons. Implementations of the interface may model a Gaussian beam spread (as does the currently used implementation) or something more complicated, including whatever optical or chromatic aberrations one might wish to simulate.

2.2.5 The material scatter model interface

NISTMonte contains a `MonteCarloSS` class, which is an "executive" class that runs a Monte Carlo simulation using an externally supplied physical model. As part of the current project the interface between this class and the model physics was generalized somewhat. It will now run a simulation for any model that implements a `MaterialScatterModel` interface. The interface specifies these methods:

```
Material getMaterial();
double getMinEforTracking();
void setMinEforTracking(double minEforTracking);
double freePathLength(Electron pe);
Electron scatter(Electron pe);
Electron barrierScatter(Electron pe, Region nextRegion);
double calculateEnergyLoss(double len, Electron pe);
```

An instance of a `MaterialScatterModel` is associated with each material in the simulation. The `getMaterial()` method returns the material with which the model is associated. `MonteCarloSS` tracks electrons until their energies fall below a (possibly material-dependent) minimum. The interface provides a getter and a setter for this parameter. (MONSEL by default sets this to the work function.) The next two methods are required for modeling electron scattering of the type discussed in Sec. 2.1.1. The `barrierScatter()` and `calculateEnergyLoss()` methods are required by Sec. 2.1.2 and Sec. 2.1.3. The Electron returned by the scatter and `barrierScatter()` methods is the SE (if any) pro-

duced by the model. If no SE is generated (e.g., scattering was elastic) the method returns `null`, but it generally still does alter the properties (e.g., the direction or energy) of the primary electron, a reference to which is passed as its input argument. The `barrierScatter()` method also can in principle return an SE, although our implementation always returns `null`. The `nextRegion` input to the `barrierScatter()` method is a reference to the region on the other side of the barrier that the electron has hit. (The identify of the region on the inside is the same as the electron's `currentRegion`, available from the first argument.)

Note that although the material is not an explicit input to these methods it is an implicit one, since each of these methods is associated with a particular `Material` and has access to its properties. In use, therefore, an instance of a `MaterialScatterModel` is associated with each region. This has advantages for speed optimization. Since methods in any given instance of the model are always called for the same material, the `MaterialScatterModel` constructor can precompute and cache any required combinations of material and other constants. For example, the `calculateEnergyLoss()` method, which determines the energy loss according to Eq. 5 (modified slightly by addition of a residual energy loss), precomputes and caches the values of $\alpha\rho$, as well as J_i and $c_i Z_i / A_i$ for each constituent. This calculation need only be performed once, during simulation setup, instead of repeatedly each time the energy loss needs to be calculated.

2.2.6 Model objects

The `MaterialScatterModel` interface described in the last section is a general interface, an implementation of which is needed for any model for which trajectories are to be run. MONSEL's particular implementation of the interface defines additional interfaces, one for a `ScatterMechanism`, one for a `BarrierScatterMechanism`, and one for a `SlowingDownAlgorithm`. As their names imply, objects that implement these interfaces supply the methods required for scattering within the bulk (Sec. 2.1.1), scattering at an interface barrier (Sec. 2.1.2), and the slowing down calculation (Sec. 2.1.3). The modeler may write as many different versions of these as desired. For example, one might define one `ScatterMechanism` for impact ionization based upon Möller's quantum mechanical derivation for free target electrons, as we currently do, and an alternative based upon Gryzinski's[25] semi-classical approach that takes into account binding energies. MONSEL's `MaterialScatterModel` is implemented as a template. Which of the predefined models is employed within a given simulation is determined at run-time by which methods the user adds to the template. The template "registers" for use one slowing down algorithm, one barrier scattering mechanism, and a list of other scattering mechanisms. The list generally includes one mechanism for scattering from nuclei (e.g., the above-described Browning form for the Mott mechanism) and one or more SE production mechanisms (e.g., Möller and plasmon).

2.2.7 Detectors

The existing NISTMonte implementation of detectors did not require modification for our purposes. Various "events" that a detector might wish to track are signalled by the simulator. These include, for example, the start of a new trajectory, the generation of a new SE, an exit material event (when an electron crosses a boundary between regions), and a backscatter event (when an electron, having escaped the sample, strikes the chamber wall). A "listener" (generally including one or more detectors) subscribes to be notified of these events. Notification includes the type of event as well as access to information about the electron. When notified it takes some detector-specific action, such as incrementing a count or histogram. The existing backscatter detector in NISTMonte keeps histograms of both energies and angles of electrons that hit the chamber wall. MONSEL simply sums the counts in energy bins less than 50 eV for the SE count and the remaining bins for a total backscattered electron count.

2.3 Program organization

This section provides a high level view of how the previously defined pieces are combined to perform a simulation.

The building blocks described so far are best considered a library of high level routines useful for performing simulations, but without a "main()" method. That is, we have not yet described the starting point for program execution. The starting point is provided by a Jython interpreter. (Jython is a Java version of the Python scripting language.) That is, it is a window into which Jython commands may be typed and executed interactively, or from which Jython scripts may be run. The

user in effect writes a short program to perform a simulation. This scheme has the disadvantage of being a non-intuitive user interface, which therefore requires a certain amount of expertise to operate. However, it has the advantage of great flexibility. A linescan or series of them is implemented as a repeat loop in which the landing position of the electrons is changed. One could also easily use a repeat loop to run multiple simulations, varying the sample shapes to span a range around the target shape of a particular manufacturing process. Measured images could then be compared to the resulting model-based library to correlate object dimensions with the observed image.[2-7]

A typical simulation process can be divided into initialization, simulation, and output sections. The sample contains materials with properties described by parameters that are set during initialization. The scattering mechanisms that operate within each material are chosen, parameters of the mechanisms (e.g., the plasmon efficiency parameter) if any are set if they differ from the defaults, and these are added to the `MaterialScatterModel` template for each material. Regions of the sample (their shapes and associated `MaterialScatterModels`) are defined. These regions become subregions of the spherical chamber. Typically the chamber is a vacuum-filled spherical region. The sample is comprised of one or more subregions of the chamber. (Regions of the sample may also have subregions as shown in Fig. 4, e.g., to describe inclusions or voids.) An electron gun object is created and its properties (beam energy, and shape, target position, etc.) defined. Finally, any desired detectors are instantiated and registered with the simulator.

From the point of view of the script, the actual running of the simulation is just a matter of sending an appropriate start command to `MonteCarloSS`. Within `MonteCarloSS` (the executive class that runs simulations, as mentioned above) the simulation consists of a user-specified number of trajectories. Each trajectory in turn consists of a sequence of trajectory legs, or steps (Fig. 5). The state of the electron (position, energy, etc.) at the beginning of the first leg is determined by the electron gun properties. Each subsequent leg starts with an electron in the state it had at the conclusion of the previous leg. Within a leg the sequence is (1) move (2) decrement energy (3) scatter. `MonteCarloSS` calls the `freePathLength()` method of the current region's `MaterialScatterModel` to determine how far the electron moves before it scatters. It then checks for the nearest boundary crossing in its current direction of motion, checking boundaries of its current region and all of that region's subregions. It moves the shorter of these distances. Its energy is then decremented as prescribed by the `MaterialScatterModel`'s `calculateEnergyLoss()` method. If the length of the step was determined by a boundary crossing, one of two things may happen. If the boundary is the chamber wall, the electron is dropped from further simulation. If it is a different boundary, the `barrierScatter()` method decides the electron's new region, direction of motion, and kinetic energy. On the other hand, if the length of the step was determined by a scattering event, the `scatter()` method is called to simulate its effects. This method may alter the electron's energy and/or direction of motion. It will also return either an SE or null. If it returns an SE, the original electron is placed on a stack while the SE trajectory is followed. Before the start of the next leg the electron's energy is compared to the minimum energy for tracking in this material. If it is less, the electron is dropped from the simulation and replaced by the next electron, if any, on the stack. In this way, one trajectory includes the original electron and all of its SE cascade.

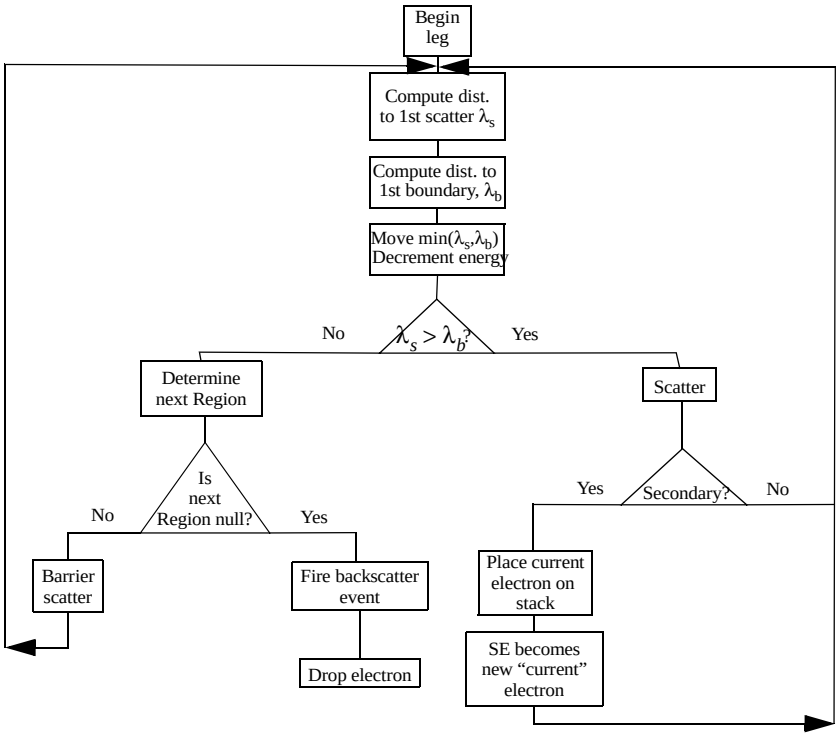

FIG. 5. One leg of a simulated trajectory.

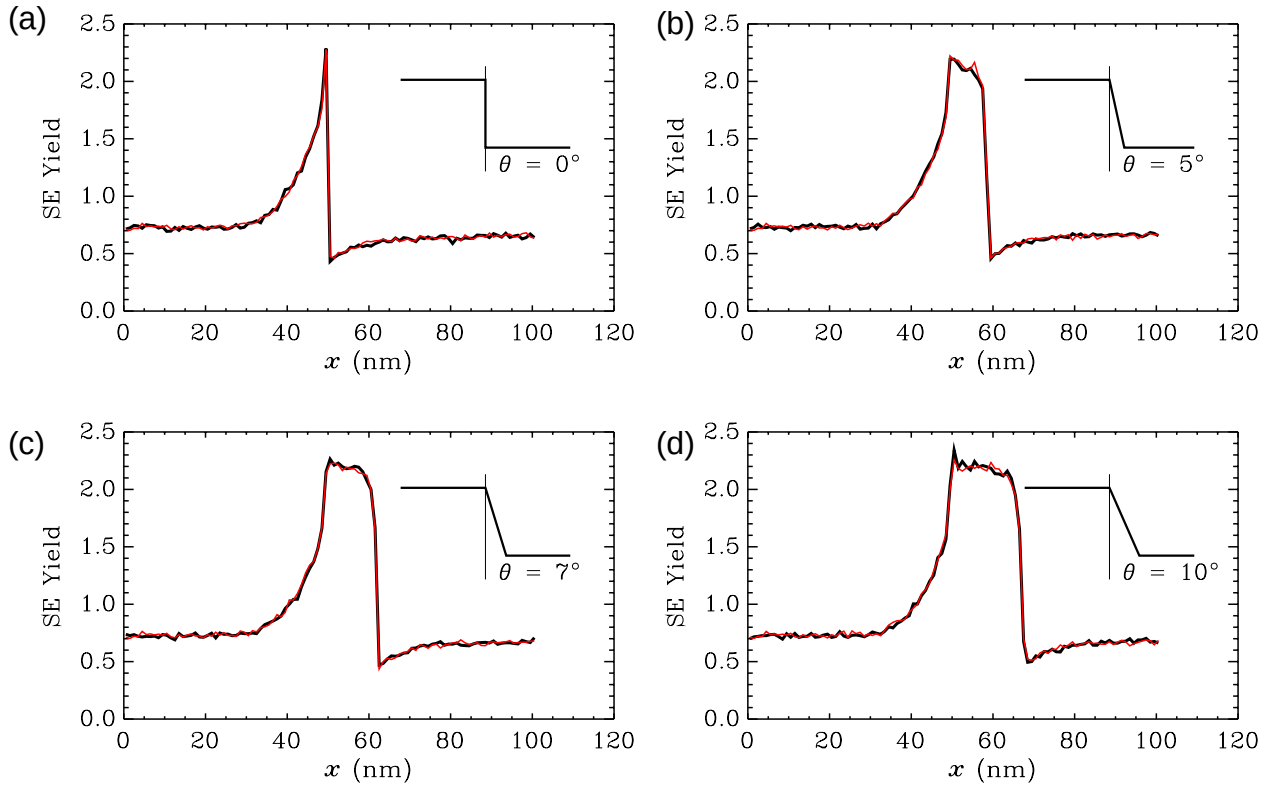

FIG. 6. Comparison of SE yields between original MONSEL (thin lines, red in color versions) and the 3D MONSEL (thick black lines 10 000 1 keV electrons incident at 1 nm spacing on 100 nm high Si lines with varying sidewall angle. Target geometry is shown in the ir Only data from the right edge is shown.

For simplicity the above description has omitted mention of event firings. Within the above sequence, events are fired when the trajectory begins or ends, when an electron hits the chamber wall, escapes from one region into the next or when an SE is generated or dropped. Each such event causes a signal to be sent to registered listeners, including detectors. These tally information as described in Sec. 2.2.7. After completion of the simulation, the Jython script will normally then query the detectors and report the output, for example by saving statistics to a file.

3. RESULTS

In this section we discuss three kinds of results from SE simulations using the new simulator. These are comparisons to the original MONSEL series, a comparison to existing experimental data, and a demonstration of some of the new capabilities.

In a few cases we have made versions of various scatter mechanisms that differ from the original MONSEL series, but in each of these cases we also made a version intended to be identical to the previous simulator. The purpose was to facilitate a comparison in which the geometrical aspects of the software (sample description, computation of boundary crossings, etc.) were upgraded, but the physical model remained the same. The sample shapes treatable by the new simulator are a superset of those available to the original code. By choosing samples within the repertoire of both versions, it is possible to perform a direct comparison, in this way to insure that no unintended changes have been introduced.

A series of such comparisons are shown in Fig. 6 for 1 keV electrons incident on Si lines with varying sidewall angle. There is considerable scatter in available experimental values. (See, for example, the values and references in David Joy's database.[26]) The yield of about 0.72 at center-of-line (at $x = 0$ nm in the figure) for both codes is roughly centered

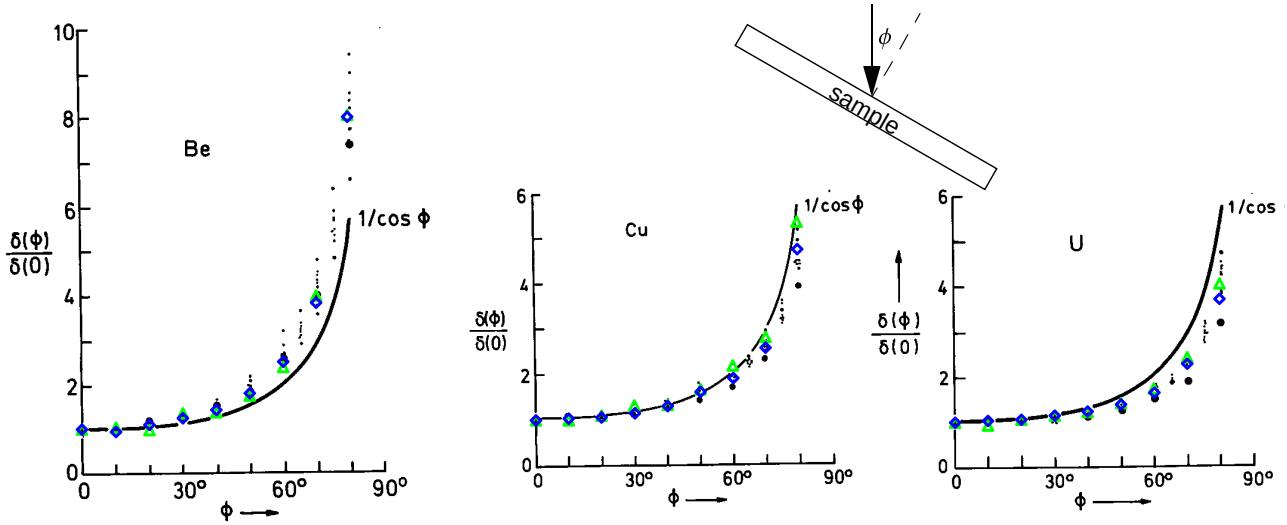

FIG. 7. SE yield variation with angle of incidence, ϕ, on elements spanning a large atomic number range. Triangles and diamonds are Monte Carlo results of the present simulator at 20 keV and 10 keV respectively. These are superimposed on graphs from Reimer.[28] In the original graphs the large filled circles are Monte Carlo results (a different model from MONSEL's), and the small dots are experimental data in the range of 10 keV to 100 keV. The $1/(\cos\phi)$ curve represents a commonly used approximation.

within this distribution when the residual energy loss was set to 1 eV/nm. The small fluctuations in yield on flat areas of the sample are due to the intrinsically statistical nature of a Monte Carlo simulation. The profiles were in excellent agreement at all tested sidewall angles.

For measuring the shapes and sizes of features we are mainly concerned with topographic contrast. In the SEM the biggest source of topographic contrast is the variation of SE yield with angle of incidence. In Figure 6d, for example, the reason the SE yield is so much higher at $x = 59$ nm than at $x = 0$ nm is that at the former position the beam is incident on the middle of the sloped sidewall. Its angle of incidence, relative to the local surface normal, is 80°. At $x = 0$ nm its angle of incidence is 0°. Simple considerations can explain the main features of this source of contrast. SE created more than a few nanometers from the surface have no opportunity to escape and be detected, because they lose too much energy. When the local surface is tilted, the incident beam stays within a given "escape depth" of the surface for a longer length of path. This simple picture explains much of the observed contrast, but there are other effects as well. For example, there is an observed dependence upon the atomic number, Z, of the sample. Figure 7 shows the ratio of SE yield at angle ϕ (the angle

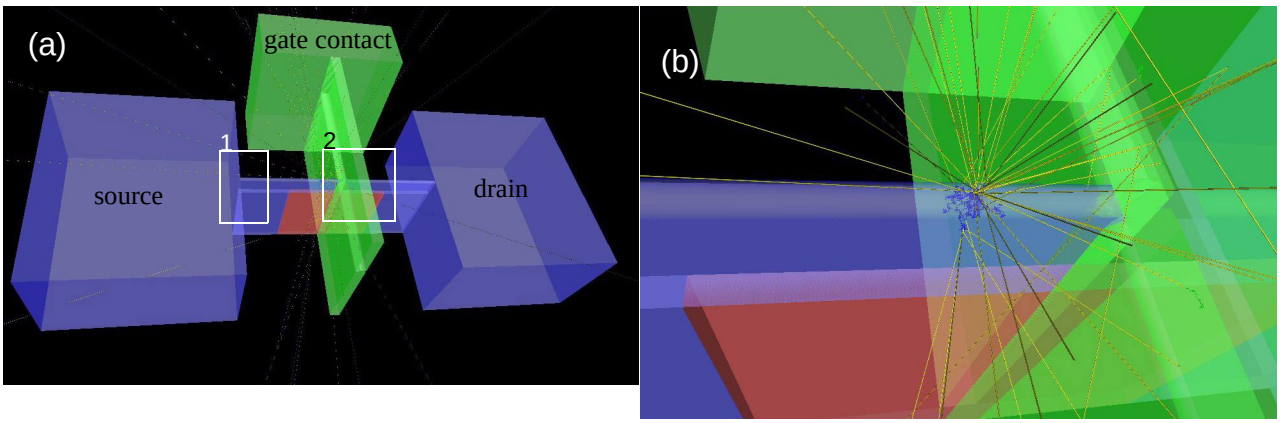

FIG. 8. A FinFET model. (a) An overview showing Si source and drain (left and right boxes), a thin connecting Si fin with HfO_2 gate oxide, and a gate metal (TiN) gate electrode with a crossing fin-like contact. The substrate (not shown, but coincident with the bottom of the pictured structures) is Si. (6) A close-up of a beam impact point just to the left of the metal contact. Some trajectories within the fin are seen to reach the sides, where electrons escape, some to be recaptured by neighboring structures.

of incidence with respect to the local surface normal, as shown upper right) to the yield at 0° for three elements spanning a large range of Z values. The $1/\cos\phi$ curve represents a rule of thumb based upon the simple model just described. The smallest dots are measured data. The larger dots are Monte Carlo model results for incident electron energies in the range from 10 keV to 100 keV from another Monte Carlo model.[27] These parts of the figure are reproduced with permission from Ref. 28. Results of the present MONSEL model at 10 keV and 20 keV are overlaid as the open diamonds and triangles respectively. The data and MONSEL results are above the $1/\cos\phi$ curve for low Z (beryllium), close to slightly below the curve for medium Z (copper), and below the curve at high Z (uranium).

Figure 8 shows an example of the kinds of applications for which the 3D capability of the new code is intended. Figure 8a shows a model of a FinFET transistor. The Si substrate is not shown. The materials in the model are Si for the source, drain, and connecting fin, HfO_2 for the gate oxide, and TiN for the metal gate. For Si we used a work function of 4.85 eV and plasmon energy of 16.5 eV. For TiN the work function was 3.74 eV and plasmon at 6 eV.[29] HfO_2 work function and plasmon energy were not readily available in the limited time available to us, so we used "best guess" values of 10 eV and 16.5 eV, similar to those we have used in the past for SiO_2. These values can presumably be improved in the future, but we do not expect the simulation results to be sensitive to their choice because at the 1 keV incident energy in this simulation

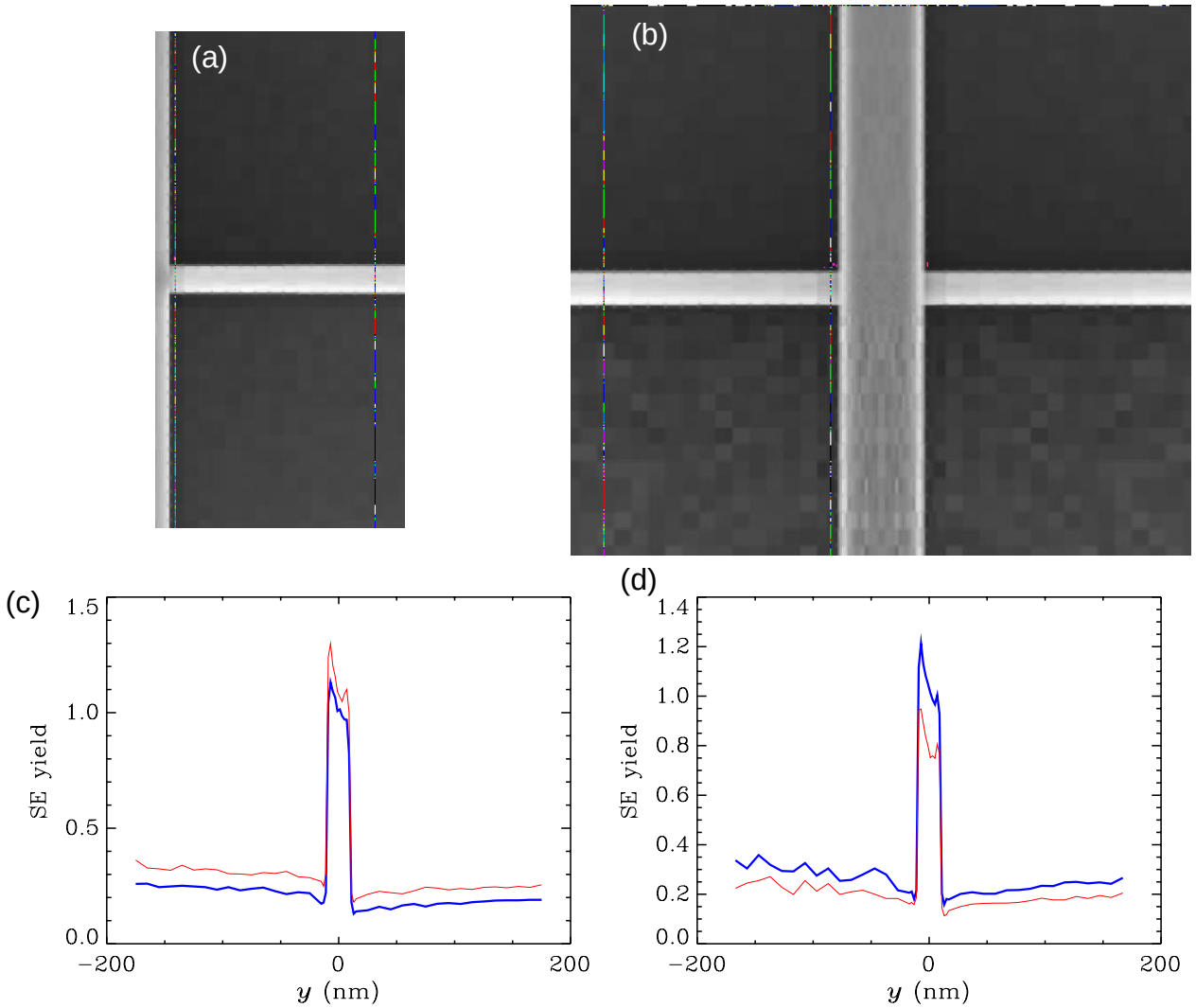

FIG. 9. a) Simulated images from locations marked in Fig. 8a. (a) Location 1, at the source/fin intersection. (b) Location 2, at the fin/gate electrode intersection. Intensity profiles along vertical cuts are shown in (c) for location 1 and (d) for location 2. In each case two profiles are compared, with the thicker line corresponding to the left-most of the two lines marked in each image.

the only electrons that enter the oxide are a few chance ricochets off of neighboring structures. Figure 8b shows trajectories resulting from an impact point on the Si fin just to the left of the TiN crossing contact. The trajectories are for 1 keV electrons. The fin width in this model was 20 nm with top corners rounded (10 nm radius). At this width some of the scattered electrons emerge from both sides of the fin to escape or be recaptured by neighboring structures.

Figure 8a is marked to show the approximate locations of two simulated images that were scanned. These images are shown in Fig. 9a and b. The images exhibit some 3D effects. They are generally darker near the top. (Notice the background in the Fig. 9c and d profiles is lower at high *y* than at low *y*.) This occurs because the gate contact above the fin recaptures some SE that would have otherwise escaped. The profiles taken near other structures are darker than the otherwise similar profiles taken farther away. The magnitude (and even the sign) of these effects depends upon the extent to which emerging SE are extracted to the detector using electric fields. The simulations shown above assumed no extraction.

4. CONCLUSIONS

The MONSEL electron trajectory simulator has been upgraded to accept a much wider range of sample shapes. Samples may now be fully 3D. Although this involved translating the entire code into Java, comparisons between the new code and the previous one for sample geometries within both their repertoires produce results that are the same to within expected statistical variations of Monte Carlo simulators. The new simulator was in good agreement with published measurement data for secondary electron yield variation with angle of incidence. New capabilities were demonstrated on a model of a FinFET transistor.

ACKNOWLEDGEMENTS

James Price and Benjamin Bunday of SEMATECH provided the FinFET schematic upon which Fig. 8 was based. One of the authors (JV) thanks András Vladár, long-time advocate of 3D modeling, for his encouragement to undertake this project. The project was funded by the NIST Office of Microelectronic Programs and the NIST Manufacturing Engineering Laboratory.

REFERENCES

1. M. Davidson and N. Sullivan, “Monte Carlo simulation for CD SEM calibration and algorithm development,” *Proc. SPIE* **2439**, pp. 334-344 (1995).
2. M. P. Davidson and A. E. Vladár, “An inverse scattering approach to SEM line width measurements,” *Proc. SPIE* **3677**, pp. 640-649 (1999).
3. J. S. Villarrubia, A. E. Vladár, J. R. Lowney and M. T. Postek, “Edge Determination for Polycrystalline Silicon Lines on Gate Oxide,” *Proc. SPIE* **4344**, pp. 147-156 (2001).
4. J. S. Villarrubia, A. E. Vladár, J. R. Lowney, and M. T. Postek, “Scanning electron microscope analog of scatterometry,” *Proc. SPIE* **4689,** pp. 304-312 (2002).
5. J. S. Villarrubia, A. E. Vladár, B. D. Bunday, and M. Bishop, “Dimensional Metrology of Resist Lines using a SEM Model-Based Library Approach,” Proc. SPIE **5375**, 2004, pp. 199-209.
6. J. S. Villarrubia, A. E. Vladár, and M. T. Postek, “Scanning electron microscope dimensional metrology using a model-based library,” Surf. Interface Anal. **37**, pp. 951-958 (2005).
7. D. V. Gorelikov, J. Remillard, N. T. Sullivan, and M. Davidson, “Model-based CD-SEM metrology at low and ultralow landing energies: implementation, and results for advanced IC manufacturing,” Surf. Interface Anal. **37**, pp. 959-965 (2005).
8. M. Tanaka, J. S. Villarrubia, and A. E. Vladár, “Influence of Focus Variation on Linewidth Measurements,” *Proc. SPIE* **5752**, pp. 144-155 (2005).

9. *International Technology Roadmap for Semiconductors,* 2005, Table 118a, Metrology Section, p. 12. Semiconductor Industry Association (http://www.itrs.net/reports.html).

10. J. R. Lowney and E. Marx, "User's Manual for the Program MONSEL-1: Monte Carlo Simulation of SEM Signals for Linewidth Metrology," NIST Spec. Pub. 400-95, 1994.

11. J. R. Lowney, "Application of Monte Carlo simulations to critical dimension metrology in a scanning electron microscope," *Scanning Microscopy* **10**, pp. 667-678 (1996).

12. J. R. Lowney, "Monte Carlo simulation of scanning electron microscope signals for lithographic metrology," Scanning **18**, pp. 301-306 (1996).

13. R. L. Myklebust, D. E. Newbury, and H. Yakowitz, "NBS Monte Carlo Electron Trajectory Calculation Program," in NBS Spec. Pub. 460, pp. 105-128 (1976).

14. Nicholas W. M. Ritchie, "A new Monte Carlo application for complex sample geometries," Surf. Interface Anal. **37**, pp. 1006-1011 (2005).

15. N. F. Mott, Proc. Royal Society London A124, 425 (1929).R. Browning, T. Eimori, E. P. Traut, B. Chui, and R. F. W. Pease, "An Elastic Cross-Section Model for Use with Monte-Carlo Simulations of Low-Energy Electron-Scattering from High Atomic-Number Targets," Journal of Vacuum Science & Technology B, vol. 9, pp. 3578-3581 (1991).

16. R. Browning, T. Z. Li, B. Chui, J. Ye, R. F. W. Pease, Z. Czyzewski, and D. C. Joy, "Empirical Forms for the Electron-Atom Elastic-Scattering Cross-Sections from 0.1 to 30 Kev," Journal of Applied Physics, vol. 76, pp. 2016-2022, Aug 15 1994.

17. R. Browning, T. Z. Li, B. Chui, J. Ye, R. F. W. Pease, Z. Czyzewski, and D. C. Joy, "Low-Energy-Electron Atom Elastic-Scattering Cross-Sections from 0.1-30 Kev," Scanning, vol. 17, pp. 250-253, Jul-Aug 1995.

18. C. Möller, "Zur Theorie des Durchgangs schneller Elektronen durch Materie," Ann. Phys. **14**, 531 (1932).

19. J. Drucker and M. R. Scheinfein, "Delocalized Secondary-Electron Generation Studied by Momentum-Resolved Coincidence-Electron Spectroscopy," Phys. Rev. B**47**, pp. 15973-15975 (1993).

20. M. R. Scheinfein, J. Drucker, and J. K. Weiss, "Secondary-Electron Production Pathways Determined by Coincidence Electron-Spectroscopy," Phys. Rev. B**47**, pp. 4068-4071 (1993).

21. D. C. Joy and S. Luo, "An Empirical Stopping Power Relationship for Low-Energy Electrons," Scanning **11**, pp. 176-180 (1989).

22. H. Bethe, "Zur Theorie des Durchgangs schneller Korpuskularstrahlen durch Materie," Materie Ann. Phys. **5**, 325-400 (1930).

23. H. A. Bethe and J. Ashkin, "Passage of Radiations Through Matter," in Emilio Segrè (ed.), Exp. Nucl. Phys., **1** pp. 166–357 (1953).

24. R. M. Nieminen, "Stopping Power for Low-Energy Electrons," Scanning Microscopy **2**, pp. 1917-1926 (1988).

25. M. Gryzinski, "Classical theory of atomic collisions I. Theory of inelastic collisions," Phys. rev. A**138**, p. 336 (1965).

26. D. C. Joy, "A database on electron-solid interactions," Scanning **17**, p. 270-275 (1995). Also http://web.utk.edu/~srcutk/.

27. H. Drescher, L. Reimer, H. Seidel, "Ruckstreukoeffizient und Sekundarelektronen-Ausbeute von 10-100 keV Elektronen und Beziehungen zur Raster-Elektronenmikroskopie," Z. Angew. Phys. **29**, 331 (1970).

28. L. Reimer, Scanning Electron Microscopy, *Physics of Image Formation and Microanalysis, Springer Series in Optical Sciences* Vol. **45** (Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, 1985). Material in Fig. 7 of this work was reproduced from Fig. 4.18a-c of this reference, with the kind permission of Springer Science and Business Media.

29. A. P. Hibbins, J. R. Sambles, and C. R. Lawrence, "Surface plasmon-polariton study of the optical dielectric function of titanium nitride," J. Mod. Optics **45**, pp. 2051-2062 (1998).