Shape-from-Shading and Simulation of SEM Images Using Surface Slope and Curvature

Adam Seeger[1*], Horst Haussecker[1]
[1]Computational Nano-Vision Research, Intel Corporation, SC12-303, 2200 Mission College Blvd., Santa Clara, CA 95054
[*]Correspondence to: Adam Seeger, adam.a.seeger@intel.com

Abstract

The Monte Carlo method is useful for modeling the physics of scanning electron microscopes but is limited in interactive applications because it can take hours or days to simulate a single SEM image. We have developed an alternative approach to simulating SEM images that mimics the behavior of Monte Carlo simulations and real SEM images but is several orders of magnitude faster to compute, simulating an image in a fraction of a second. Furthermore, this approach enables one to solve the inverse problem of reconstructing surface topography from SEM images within minutes to hours. Our method is limited to surfaces that are height fields and that are composed of only a single material. Also, we assume there are no charging affects and that the shading of the surface is rotation invariant (no detector asymmetry).

We represent the input surface to our simulator as a height image with the same resolution as the desired SEM image. An SEM image is simulated from the height image by first convolving it with each of a set of separable filter kernels. Each of these filters kernels is designed to extract local information about slope and curvature at different scales. After these convolutions, the resulting filtered images are added together with different weighting factors to produce the simulated image.

The weighting factors are found by a fitting procedure. We take examples of height images and corresponding SEM images either generated synthetically using the Monte Carlo method or acquired from an actual SEM. In the case of real SEM images, the corresponding height image must be known either from design data or measured independently (for example, using an AFM). These example pairs of height and SEM images are used to determine the weighting factors which may then be used to simulate an SEM image from any other height image.

Keywords: Shape-from-shading, surface reconstruction, SEM, Monte Carlo, image simulation

Introduction

Jones and Taylor[10] describe a method for reconstructing surfaces from SEM images using a shape-from-shading approach where the intensity in the SEM image is assumed to be a function of height gradient magnitude or slope of the surface. At sub-micrometer scale this approximation breaks down as the finite size of the electron-specimen interaction volume becomes significant. This paper describes a way to extend this approach to smaller scales by taking into account both the slope and curvature of the surface computed at multiple scales.

The optimization approach used by Jones and Taylor[10] could in theory be adapted to use Monte Carlo simulation but such an algorithm would be impractical because it would require the simulation of thousands of SEM images and thousands of hours of computation. In an effort to

bridge the gap in performance/accuracy between the Monte Carlo model and slope-based models of SEM intensity, we developed a generalization of the slope-based approach that uses combinations of additional convolution-based filters (besides those computing slope) to improve accuracy. This approach provides a more powerful phenomenological model that more closely mimics the output of Monte Carlo simulation for nanometer-scale topography but is fast enough to be part of an iterative optimization algorithm similar to what has been used for shape-from-shading. In the next section we explain some of the inspiration for this approach by giving an intuitive description of what is measured by an SEM and how it relates to surface shape. Next we describe the new simulation method and provide performance measurements and simulation examples. After this, we describe how this model can be used for surface reconstruction and provide some examples.

SEM Signal Components and Rationale for Our Approach

For each pixel in an SEM image, the electron beam is focused at a corresponding point on the specimen surface. Electrons from the beam are scattered within the volume of the specimen and some escape to hit a detector where they contribute to the measured signal that determines the brightness of the pixel. The electrons detected in an SEM can be roughly divided into two groups by energy because there is a distinct division in the energy spectrum between high energy backscattered electrons (BSE) and low energy secondary electrons (SE). The SE component can be considered as having three subcomponents labeled SE-I, SE-II, SE-III [6]. Because secondary electrons (SE) have much lower energy they can only travel a short distance before being stopped and as a result only escape the specimen if they are generated within a few nanometers of the surface. BSE on the other hand can travel much farther before escaping. The distance an electron travels in the specimen before it escapes determines in some sense the resolution of shape information carried by that electron. The SE-I signal represents SE escaping very near the point where the electron beam enters the specimen surface and is considered to carry the highest resolution information. The BSE signal represents high energy back-scattered electrons that have traveled much farther in the specimen and escape from a much larger region of the surface. The SE-II signal represents low energy secondary electrons produced near the surface by escaping BSE and therefore provides a sort of amplifying effect that depends on the topography near the point of escape in the same way that the SE-I signal depends on the topography near the point of beam entry. The SE-III signal represents secondary electrons produced when BSE strike various parts of the SEM other than the specimen. The SE-III signal provides an amplifying effect for BSE that depends on the shape and materials of the SEM specimen chamber.

In the shape-from-shading literature, the models for SEM are all based on the slope of the surface with no notion of the scale for measuring slope[10]. If the SEM signal is modeled as a function of slope, then the slope should be measured at a scale (or scales) comparable to the electron-specimen interaction volume. This concept is illustrated in Figure 1 showing constant slope approximations to a surface about a point where electrons hit the surface. The approximating surfaces are fit only to an area comparable to the area from which electrons escape the specimen.

The SEM signal consists of multiple components, each representing interaction with the specimen at different scales so we use estimates of the slope at multiple scales to predict the SEM signal. Although slopes at multiple scales would certainly be better than a single slope measurement for estimating the SEM signal, slope by itself is insufficient for modeling the

appearance of sharp bumps, corners and pits on the surface so we also make use of local curvature estimated at multiple scales. We call this SEM model based on slope and curvature the filter bank model because it is expressed in terms of the outputs of a set of intermediate filters (the filter bank). Each filter computes either a slope or curvature value at a particular scale.

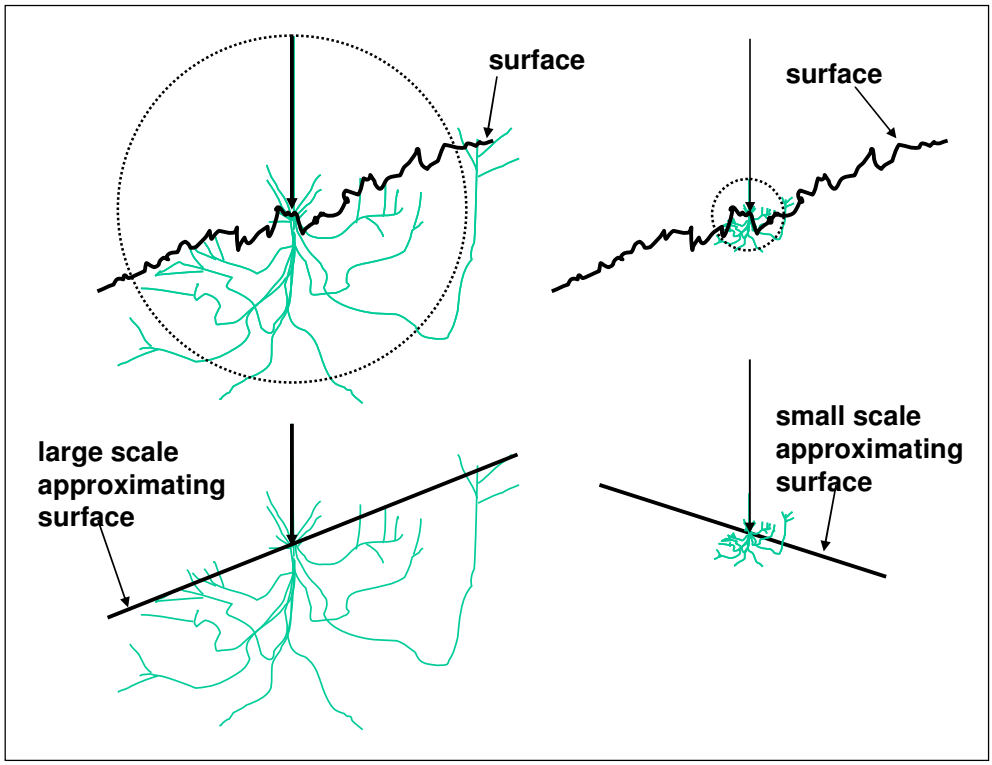

Figure 1: Taking into account the scale of the interaction volume when modeling the number of escaped electrons as a function of the local slope. When the interaction volume is large (left) the slope should represent the surface over a large neighborhood. When the interaction volume is small (right) the slope should represent the surface over a small neighborhood.

SEM Modeling Overview

The filter bank model consists of a set of filters (each computing either a slope or curvature value) and a function that maps the slope and curvature values to SEM intensity. A schematic of how the filter bank model computes an SEM image from a height image is illustrated in Figure 2. The mapping from slope and curvature values to SEM intensity is determined automatically by fitting to example Monte Carlo simulated images and real SEM images. Synthetic surfaces and corresponding Monte Carlo simulated images were used to determine a pair of filter bank models that mimic the BSE and SE outputs of the Monte Carlo model. Experimental AFM and SEM data were then used to determine how to combine the BSE and SE images to best match the real SEM image. An outline of the method including the fitting procedure is illustrated in Figure 3.

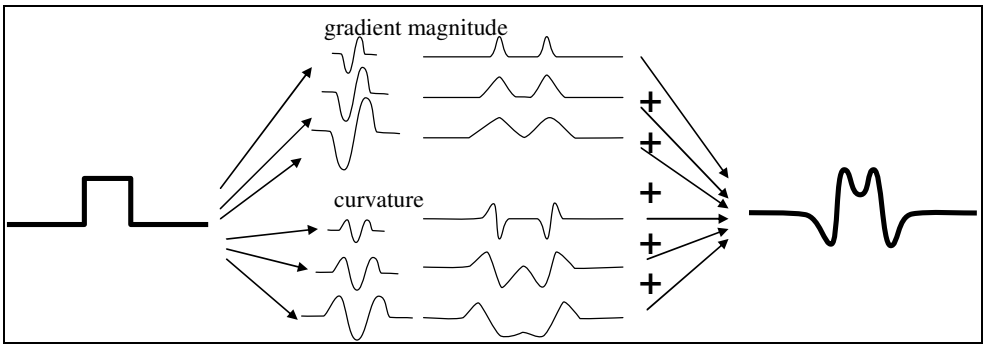

Figure 2: Filter bank model applied to simulating the SEM signal. Surface height (left) is filtered by each of a set of slope and curvature convolution filters to generate corresponding intermediate images that are summed to compute the simulated SEM signal (right). Parameters of the model control weighting of the intermediate images in the sum.

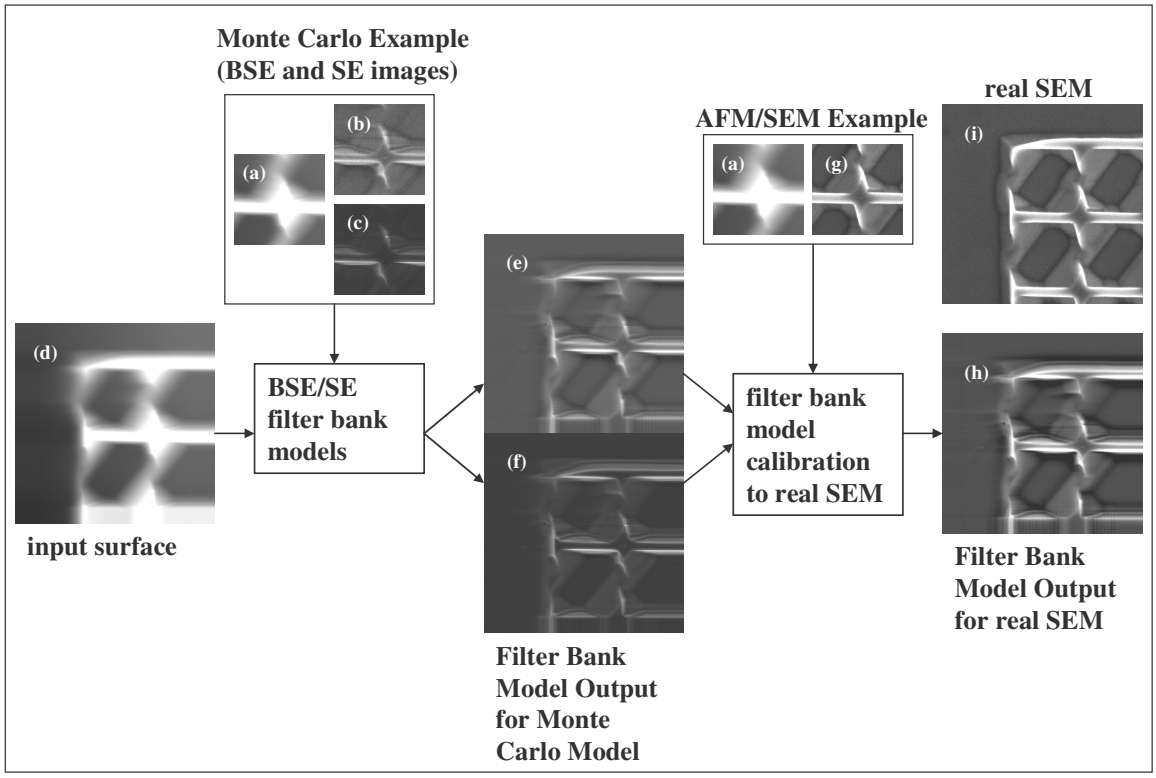

Figure 3: SEM model-fitting framework for filter bank model. A surface reconstruction from AFM (a) along with Monte Carlo simulated BSE (b) and SE (c) images made from this reconstruction are used to determine models that mimic the Monte Carlo simulation (BSE/SE filter bank models). The same AFM-based surface (a) and a real SEM image of that surface (g) are used to determine how the BSE and SE images (e) and (f) should be combined to best match the real SEM image. Given an input surface (d) (in this case a larger area of the same surface used to construct the filter bank model but in general a novel surface), the filter bank model computes an output (h) that is comparable to a real SEM image of that surface (i).

Estimating First and Second Derivatives of Height Using the Cubic Facet Model

A facet model is used to estimate slope and curvature as described by Haralick and Watson[7]. For the facet model, a *facet* is not just a planar face (as in the common definition) but can be any

approximating function over a subregion of an image. One common choice for the approximating function, and the one used here, is a two-dimensional cubic polynomial fit to a local neighborhood at each pixel in the image. Each pixel has a different facet represented by the cubic polynomial fit about the pixel. A local neighborhood for a pixel (*x*,*y*) in an image *H* is approximated by a two-dimensional cubic polynomial

$$H\left(x+t_x, y+t_y\right) \approx$$
$$f\left(t_x, t_y\right) = K_1 + K_2 t_y + K_3 t_x + K_4 t_y^2 + K_5 t_y t_x + K_6 t_x^2 + K_7 t_y^3 + K_8 t_y^2 t_x + K_9 t_y t_x^2 + K_{10} t_x^3$$

where $t_y \in Y$ and $t_x \in X$ represent row and column indices for a rectangular-shaped neighborhood with center at (0,0). For example, for a 5x5 neighborhood, $X = Y = \{-2, 1, 0, 1, 2\}$. K_1, K_2, K_3... K_{10} are functions of *H* (the image being fit) and the pixel location (*x*,*y*). The first and second derivatives of *H* are estimated as

$$H_x\left(x, y\right) \approx K_3\left(x, y\right)$$
$$H_y\left(x, y\right) \approx K_2\left(x, y\right)$$
$$H_{xx}\left(x, y\right) \approx 2K_6\left(x, y\right)$$
$$H_{xy}\left(x, y\right) = H_{yx}\left(x, y\right) \approx K_5\left(x, y\right)$$
$$H_{yy}\left(x, y\right) \approx 2K_4\left(x, y\right)$$

A detailed description of how the cubic polynomial coefficients (K_1, K_2, K_3... K_{10}) are computed is described in a previous publication[13]. First and second derivatives are computed for 10 different neighborhoods centered about each pixel with sizes (in pixels) 5x5, 7x7, 9x9, 11x11, 15x15, 21x21, 29x29, 41x41, 59x59, and 83x83.

Estimating the SEM Signal from First and Second Derivatives of the Height

After computing the first and second derivatives of the height for all the neighborhood sizes, the set of derivative values for each pixel is transformed to two different values: the estimated BSE and SE yields (ratio of BSE and SE emitted to the number of incident electrons). Given examples of height field images and corresponding BSE and SE yield images computed by Monte Carlo simulation, we optimize a pair of functions that estimate the BSE and SE yields from the derivatives of the height field at each pixel. The BSE and SE yields computed by our Monte Carlo simulation are (in the limit as the number of simulated electrons goes to infinity) circularly symmetric. This means that the first derivatives can be first transformed to the gradient magnitude without loss of information needed to predict the BSE or SE yield. We also convert the second derivatives into a rotationally invariant form: the minimum and maximum principal curvatures. For each pixel (*x*,*y*) in the height field and each neighborhood size *n*, we compute gradient magnitude ($G_n\left(x, y\right)$) and two principal curvatures ($\kappa_{+,n}\left(x, y\right), \kappa_{-,n}\left(x, y\right)$) as

$$G_n(x,y) = \sqrt{\left(K_{2,n}(x,y)\right)^2 + \left(K_{3,n}(x,y)\right)^2}$$

$$\kappa_{+,n}(x,y) = K_{6,n}(x,y) + K_{4,n}(x,y) + \sqrt{\left(K_{6,n}(x,y) - K_{4,n}(x,y)\right)^2 + \left(K_{5,n}(x,y)\right)^2}$$

$$\kappa_{-,n}(x,y) = K_{6,n}(x,y) + K_{4,n}(x,y) - \sqrt{\left(K_{6,n}(x,y) - K_{4,n}(x,y)\right)^2 + \left(K_{5,n}(x,y)\right)^2}$$

We used 10 neighborhoods with sizes (in pixels) (5x5, 7x7, 9x9, 11x11, 15x15, 21x21, 29x29, 41x41, 59x59, 83x83) and standard deviations for the weighting function (in pixels) (0.6, 0.8485, 1.2, 1.697, 2.4, 3.394, 4.8, 6.788, 9.6, 13.58) so the BSE and SE yields for each pixel were each represented by functions of 10x3=30 different values. We chose to use linear functions to combine the gradient and curvature values into a BSE yield (FB_{BSE}) and SE yield (FB_{SE}):

$$FB_{BSE}(x,y) = d_{BSE} + \sum_{n=1..N} a_{BSE,n} G_n(x,y) + b_{BSE,n}\kappa_{+,n}(x,y) + c_{BSE,n}\kappa_{-,n}(x,y)$$

$$FB_{SE}(x,y) = d_{SE} + \sum_{n=1..N} a_{SE,n} G_n(x,y) + b_{SE,n}\kappa_{+,n}(x,y) + c_{SE,n}\kappa_{-,n}(x,y)$$

Equation 1: Filter bank models for simulating BSE and SE images

where N is the number of neighborhood sizes, and $G_n(x,y), \kappa_{+,n}(x,y), \kappa_{-,n}(x,y)$ are the gradient magnitude, maximum principal curvature and minimum principal curvature for neighborhood size n computed from the cubic fit at each pixel in the height field image. Using the method described in the next section, the parameters d_{BSE}, $a_{BSE,n}$, $b_{BSE,n}$, $c_{BSE,n}$ and d_{SE}, $a_{SE,n}$, $b_{SE,n}$, $c_{SE,n}$ (n=1..N) are determined automatically from training examples generated using Monte Carlo SEM simulation. These parameters will vary depending on the physical dimension of the pixels, accelerating voltage, and the beam shape.

Optimizing the Parameters of a Filter Bank Model

This section describes the fitting procedure used to find the parameters of the filter bank model from example input/output images computed by Monte Carlo simulation. The procedure takes as input a height image (H_{train}) and a corresponding simulated image (I_{train}) computed by Monte Carlo simulation[12]. First the H_{train} image is filtered by the filter bank to generate a set of filter bank output images $\{F_i, i=1..N_f\}$. Next, a principal component analysis (PCA) of the output images is used to eliminate redundant dimensions in the filter basis[9]. The principal components are computed as the eigenvectors of the N_fxN_f correlation matrix of filter bank outputs. The eigenvalues of the correlation matrix measure the amount of variation in the filter bank outputs explained by the corresponding principal components. It is common to ignore those principal components with eigenvalues less than 1.0 but to provide sufficient accuracy we had to use components with eigenvalues down to 0.1. The risk in including components with small eigenvalues is that the problem of determining the optimal combination of these components can become poorly conditioned and accuracy may suffer due to numerical errors. The result of the PCA is a smaller set of linear transformations of the outputs of the filter bank:

$$F_{PC,i}(x,y) = \sum_{j=1..N_f} PC_i[j] \cdot F_j(x,y), \; i = 1..M$$

where $PC_i[j]$ is the jth element of the ith principal component and M is the number of principal components with eigenvalues greater than 0.1 ($M \le N_f$).

The next step is to find parameters D and $\{B_i, i=1..M\}$ such that

$$I_{\text{train}}(x, y) = D + \sum_{i=1..M} B_i F_{PC,i}(x, y)$$

To do this we solve the linear least squares problem

$$\begin{bmatrix} I_{\text{train}}(1,1) \\ I_{\text{train}}(1,2) \\ \vdots \\ I_{\text{train}}(N_x, N_y) \end{bmatrix} = \begin{bmatrix} F_{PC,1}(1,1) & F_{PC,2}(1,1) & \cdots & F_{PC,M}(1,1) & 1 \\ F_{PC,1}(1,2) & F_{PC,2}(1,2) & \cdots & F_{PC,M}(1,2) & 1 \\ & & \vdots & & \\ F_{PC,1}(N_x, N_y) & F_{PC,2}(N_x, N_y) & \cdots & F_{PC,M}(N_x, N_y) & 1 \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_M \\ D \end{bmatrix}$$

Finally, we transform the B_i into the A_i as

$$A_i = \sum_{j=1..M} PC_j[i] \cdot B_j$$

The offset D and the filter bank coefficients A_i are then used in Equation 1.

For example, for the BSE filter bank model in Equation 1 the training input would be a height field and the training output would be the BSE image output from the Monte Carlo simulation with that height field. In practice, one must choose a particular order for the filter bank outputs $G_n(x, y), \kappa_{+,n}(x, y), \kappa_{-,n}(x, y)$ so we assume the order

$$\begin{aligned} &\left(F_1(x, y), F_2(x, y), \ldots, F_{N_f}(x, y)\right) = \\ &\quad \left(\begin{array}{l} G_1(x, y), \kappa_{+,1}(x, y), \kappa_{-,1}(x, y), \\ G_2(x, y), \kappa_{+,2}(x, y), \kappa_{-,2}(x, y), \ldots, G_N(x, y), \kappa_{+,N}(x, y), \kappa_{-,N}(x, y) \end{array}\right) \end{aligned}$$

With this order, the parameters in Equation 1 would be given by

$$d_{BSE} = D$$

$$\left(a_{BSE,1}, b_{BSE,1}, c_{BSE,1}, a_{BSE,2}, b_{BSE,2}, c_{BSE,2}, \ldots, a_{BSE,N}, b_{BSE,N}, c_{BSE,N}\right) = \left(A_1, A_2, \ldots, A_{N_f}\right)$$

Calibrating an SEM Model to a Real SEM Image

The Monte Carlo simulation image does not quantitatively match experimental data so an additional transformation is applied to the simulated signal to make it match the experimental data. We first review a previously described approach and then describe our approach for matching Monte Carlo simulation image to experimental images. Because the filter bank model is constructed to mimic the Monte Carlo simulation, the same calibration procedure is used to match this model to experimental images.

Calibrating Monte Carlo Simulation to Experimental Data

Differences between Monte Carlo simulation and experimental data have previously been handled by only comparing the simulation to the experimental data up to a slowly varying multiplicative factor[2]:

$$E(x) = MC(x) \cdot F(x) + \varepsilon(x)$$

where E is the experimental data, MC is the Monte Carlo simulation data, ε represents the residual error between the simulation and experimental data, and F is a slowly varying function defined by

$$F(x) = \frac{\int_{x-R}^{x+R} E(x')\,dx'}{\int_{x-R}^{x+R} MC(x')\,dx'}$$

where R is an arbitrary range parameter. If one wishes to optimize a surface, minimizing ε, this approach could easily generate additional local minima that would make local optimization methods fail. For example, a simulated image that matches the experimental data up to multiplication by 0.5 would be scored the same as a simulated image that matches up to multiplication by 1.0. This approach makes sense for the exhaustive search used previously[2] but because it creates multiple local minima it does not make sense for local optimization.

We scale and offset the simulated image to match it to a real image. We model the experimental image as

$$E(x) = MC(x) \cdot \text{gain} + \text{offset} + \varepsilon(x)$$

where the gain and offset are constants for an image. This makes sense given the fact that the real signal undergoes such a transformation that is set when the SEM user adjusts brightness and contrast controls. The lack of variation in the gain and offset across the image means that this approach cannot take into account non-uniformities (making one part of the image look brighter than another part) due to the SEM detector geometry, charging or surface contamination but we assume that such non-uniformities are not significant in our experimental data. The Monte Carlo simulation that we use actually gives two different signals, one for BSE and one for SE and we combine these to approximate the experimental signal using two different gain parameters:

$$E(x) = MC_{\text{BSE}}(x) \cdot \text{gain}_{\text{BSE}} + MC_{\text{SE}}(x) \cdot \text{gain}_{\text{SE}} + \text{offset} + \varepsilon(x)$$

where MC_{BSE} is the simulated BSE signal and MC_{SE} is the simulated SE signal.

We assume that gain_{BSE}, gain_{SE} and offset are constants that can be determined for a particular SEM and operating conditions including the brightness/contrast setting. One of the difficulties in using an SEM for quantitative analysis is that the amplifier gain and offset set by brightness/contrast settings are not typically stored or even accessible to the user except indirectly by visual inspection of the position of analog knobs. However, given an experimental SEM image of a specimen with known shape and materials, one could estimate the gain_{BSE}, gain_{SE} and offset parameters by simulating BSE and SE images (MC_{BSE} and MC_{SE}) and minimizing the cost function

$$f(\text{gain}_{\text{BSE}}, \text{gain}_{\text{SE}}, \text{offset}) = \sum_x \left[MC_{\text{BSE}}(x) \cdot \text{gain}_{\text{BSE}} + MC_{\text{SE}}(x) \cdot \text{gain}_{\text{SE}} + \text{offset} - E(x) \right]^2$$

Equation 2: cost function minimized to find calibration gain and offset parameters

We use this approach except instead of using a specimen with known shape we estimate the shape from an AFM image. Consequently, MC_{BSE} and MC_{SE} are only estimates of the simulated images for the actual specimen. We minimize Equation 2 using the least squares solver DGELS in the LAPACK library[1].

Calibrating the Filter Bank Model to Experimental Data

To calibrate the filter bank model we followed the same procedure from the end of the previous section but replaced MC_{BSE} with FB_{BSE} and MC_{SE} with FB_{SE}. The formula for the final simulated image (that quantitatively approximates the experimental SEM image) is also just a linear combination of the same filter bank outputs

$$S_{FB}(x,y) = d + \sum_{n=1..N} a_n G_n(x,y) + b_n \kappa_{+,n}(x,y) + c_n \kappa_{-,n}(x,y)$$

Equation 3: Calibrated filter bank model

where

$$\begin{aligned} a_n &= \mathrm{gain}_{\mathrm{BSE}} a_{BSE,n} + \mathrm{gain}_{\mathrm{SE}} a_{SE,n} \\ b_n &= \mathrm{gain}_{\mathrm{BSE}} b_{BSE,n} + \mathrm{gain}_{\mathrm{SE}} b_{SE,n} \\ c_n &= \mathrm{gain}_{\mathrm{BSE}} c_{BSE,n} + \mathrm{gain}_{\mathrm{SE}} c_{SE,n} \\ d &= \mathrm{gain}_{\mathrm{BSE}} d_{BSE} + \mathrm{gain}_{\mathrm{SE}} d_{SE} + \mathrm{offset} \end{aligned}$$

This calibrated filter bank model can be used to estimate the experimental SEM image for an arbitrary surface and this is the model that we used later for surface reconstruction.

Acceleration Methods and Performance Measurements

Sharing the Filter Bank between SE and BSE

The same filter bank is used to estimate both the SE and BSE outputs of the Monte Carlo simulation. While the SE signal tends to have higher spatial frequencies than the BSE signal, there is a significant amount of overlap because the SE-II component is approximately half of the total SE signal and the SE-II component is a sort of amplification of the BSE signal. Because almost all of the running time for the filter bank model is in the convolutions and square roots required to compute the slope and curvature estimates, by sharing this computation we can compute both the BSE and SE images in nearly the same time as it takes to compute either one, yielding a nearly 2x speedup.

Separability

For a 300x300 pixel input image and a variety of filter sizes ranging from 5x5 to 83x83, separating the coefficient filters into 1D convolutions in x and y gave a speedup of 12x over an implementation that did not take advantage of separability. The filters for K_4, K_5, and K_6 are directly separable and K_2 and K_3 can be decomposed into a sum of two parts and each part is then separable[13].

Vector Acceleration

The most time consuming operation is the 1D convolution required to compute the separable filter outputs. An additional speedup of 5x was achieved by using SIMD CPU instructions (MMX on Intel Pentium M, 1.5 GHz) that help to parallelize 1D convolutions and were accessed through the Intel Performance Primitives library.

Multi-resolution Gaussian-weighted Facet Model

Cubic fits over large neighborhood sizes require large filter kernels and convolution with these kernels becomes the main performance bottleneck. A subsampling scheme was developed to help speed up the calculation of cubic fits over large neighborhoods. For example, the cubic fit over a large neighborhood with a Gaussian weighting function with standard deviation 13.58 pixels may be approximated by subsampling the input height field by a factor of 16, computing the cubic fit with a Gaussian weighting function with a standard deviation of 13.58/16=0.84375 and then upsampling the result by a factor of 16. Subsampling was implemented efficiently by recursively subsampling by a factor of 2. The subsampling version of a cubic fit is in general not the same as the non-subsampling version because it is only an approximation based on a subsampled image. These differences are somewhat compensated by the optimization procedure used to find the parameters of the filter bank model. We did not use this optimization in practice because among all the optimizations this is the only one that significantly affected accuracy and the additional speedup was not critical for our purpose.

Summary of Performance Gains

Here we compare the performance benefits of the various optimizations. The test input image (Figure 4) has 300x300 pixels. The output (Figure 5) is an SE and a BSE image both with 300x300 pixels. The filter bank consists of 3 filters for each neighborhood: gradient and the two principal curvatures. These are computed from the 5 linear and quadratic coefficients of the cubic polynomial fit for each neighborhood. There are 10 neighborhood sizes: 5x5, 7x7, 9x9, 11x11, 15x15, 21x21, 29x29, 41x41, 59x59, and 83x83. Thus, in total there are 50 2D convolutions for the coefficient filters and the output of these convolutions gets converted to 30 slope/curvature basis images. The initial implementation with no optimization took 110 seconds to run. After all optimizations were implemented, the running time was 0.12 seconds giving a speedup of about 912x. Compared with Monte Carlo simulation using 1000 electrons per pixel, this represents a speedup of about 150,000x. The performance gains and running time after each optimization technique was implemented are summarized in Table 1 in the order they were added.

acceleration technique	incremental speedup	cumulative speedup	running time
sharing filter bank between SE and BSE images	1.9x	1.9	58 seconds
separability of filters	12x	22.8	4.83 seconds
SIMD hardware	5x	114	0.97 second
miscellaneous optimization using profiler (Intel VTune)	2x	228	0.48 seconds
subsampled filtering	4x	912	0.12 seconds

Table 1: Summary of performance gains. The incremental speedup refers to the speedup when using an acceleration technique compared with not using that acceleration technique but keeping the rest of the implementation the same.

Simulation Results

Comparison with slope function

Both the filter bank model and a slope function described previously[13] were used to simulate SEM images from the input height field shown in Figure 4.

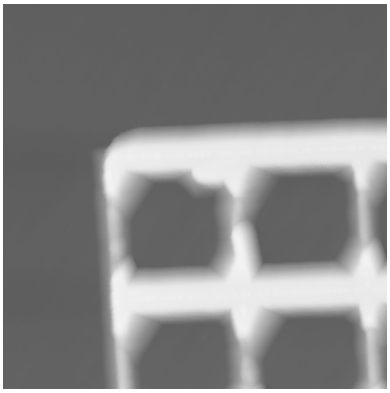

Figure 4: Test input height field

Figure 5 shows BSE and SE SEM images predicted by Monte Carlo simulation, the piecewise linear function of slope constructed by fitting to the Monte Carlo simulation using the two-plane method described in the appendix, and filter bank models fit to the Monte Carlo simulation.

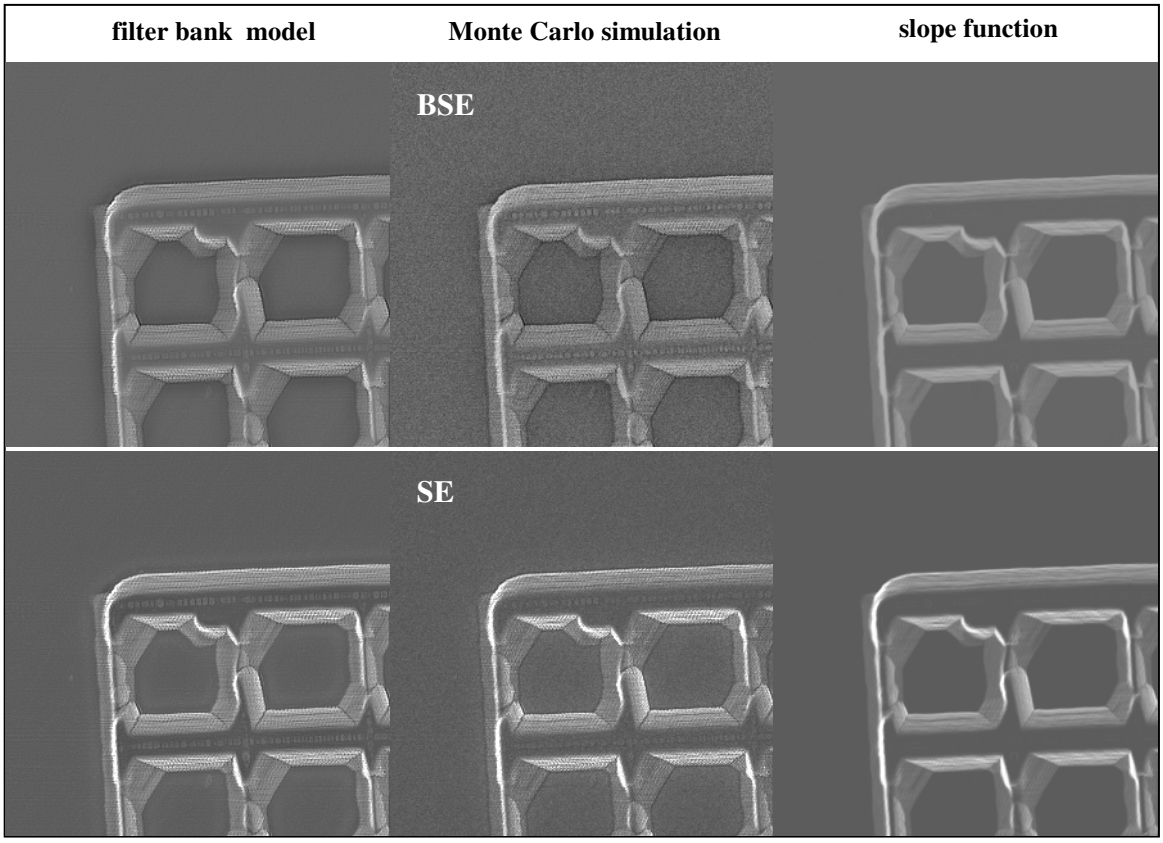

Figure 5: Comparison of simulated SEM images output by Monte Carlo model, a function of slope (two-plane slope model - see the appendix), and the filter bank model. A single intensity scale is used for all the BSE images and another intensity scale is used for all the SE images. Note especially the darkening in crevices and the bright high-curvature bumps present in the filter bank and MC simulations that are missing in the slope model.

Figure 6 shows the difference in error between the slope function and the filter bank model. Large errors for the slope function in areas of high curvature are significantly reduced for the filter bank model. The differences shown in Figure 6 are described more quantitatively in Table 2. The errors listed in Table 2 are relative errors in units of the true signal value as estimated by the Monte Carlo simulation. The signal values are SE or BSE yield which refers to the ratio between the number of SE or BSE and the number of incident electrons. When averaged over the whole image, the relative errors for the filter bank model are about half those for the slope function but because the error is concentrated in a relatively small area where the surface is highly concave or convex (as can be seen in Figure 6), the difference must be significantly larger in these parts of the image.

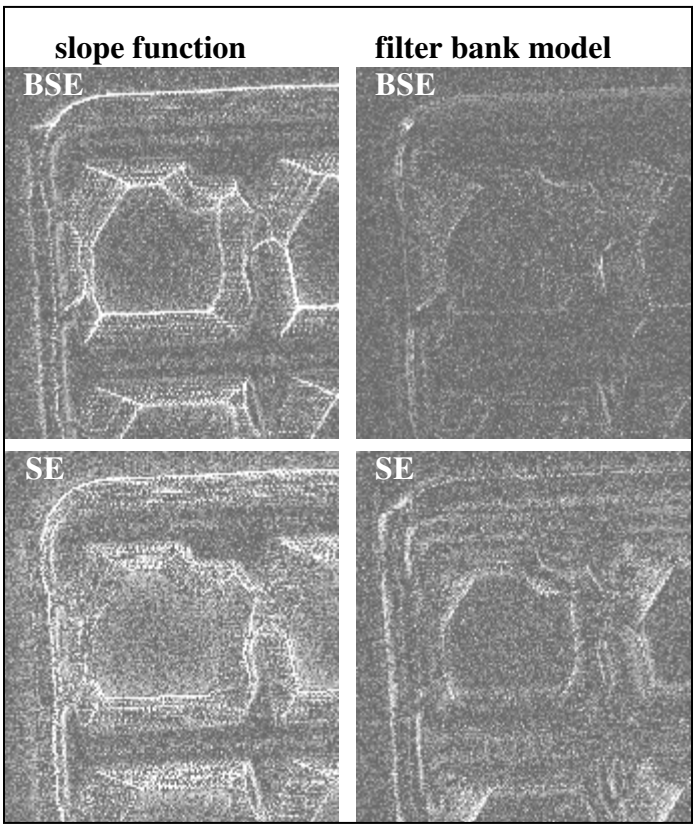

Figure 6: Estimates of absolute value of relative error in intensity from slope function (left) and for the filter bank model (right). The intensity scales are identical between the two BSE images and between the two SE images. The relative error is with respect to the Monte Carlo images.

	mean	max	median	std. dev.
$\left\|\left(BSE_{slope} - BSE_{MC}\right)/BSE_{MC}\right\|$	0.0986	2.89	0.0731	0.104
$\left\|\left(BSE_{FB} - BSE_{MC}\right)/BSE_{MC}\right\|$	0.0545	0.480	0.0446	0.0440
$\left\|\left(SE_{slope} - SE_{MC}\right)/SE_{MC}\right\|$	0.110	2.32	0.0872	0.109
$\left\|\left(SE_{FB} - SE_{MC}\right)/SE_{MC}\right\|$	0.0714	1.22	0.0589	0.0569

Table 2: Some statistics from the images shown in Figure 6. The slope subscript signifies the image computed using the function of slope, the FB subscript signifies the image computed using the filter bank model, and the MC subscript signifies the image computed by Monte Carlo simulation.

The superior ability of the filter bank model to emulate Monte Carlo simulation compared with a function of slope becomes more evident as the structures get smaller. This was tested using two synthetic surfaces shown as the "input" grayscale height images in Figure 9 and Figure 10. Before doing this test, the filter bank model had to be trained using images with the same pixel size as the images to which it would be applied. The training images for the two pixel sizes are shown in Figure 7 and Figure 8. The results from applying the filter bank models to the test structures are shown along with the slope function and Monte Carlo output in Figure 9 and Figure 10. These results demonstrate that the filter bank captures qualitative aspects of the Monte

Carlo simulation that the slope function cannot. For example, there is no way for the slope function to predict a lower intensity for highly concave parts of the surface than for flat parts of the surface as the filter bank model does.

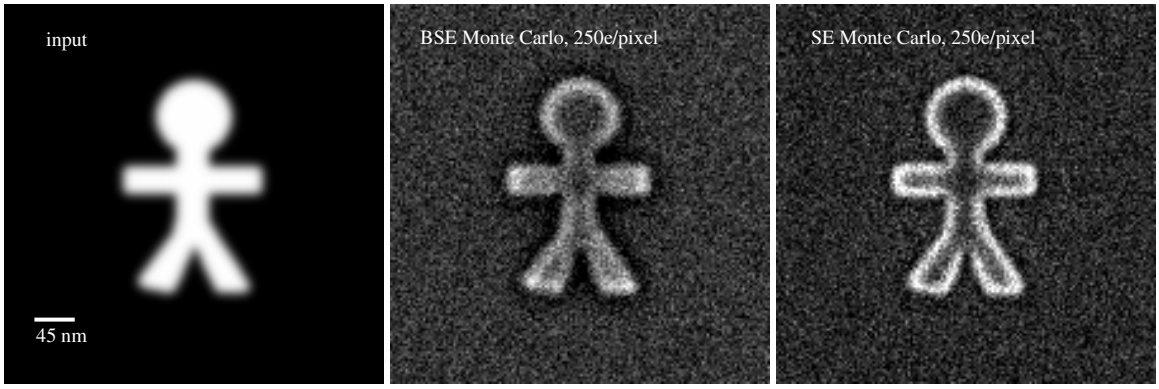

Figure 7: Filter bank model training data for 3nm/pixel resolution. Images are 150x150 pixels. 250 electrons/pixel were used to generate the training output. The input structure is 20nm high.

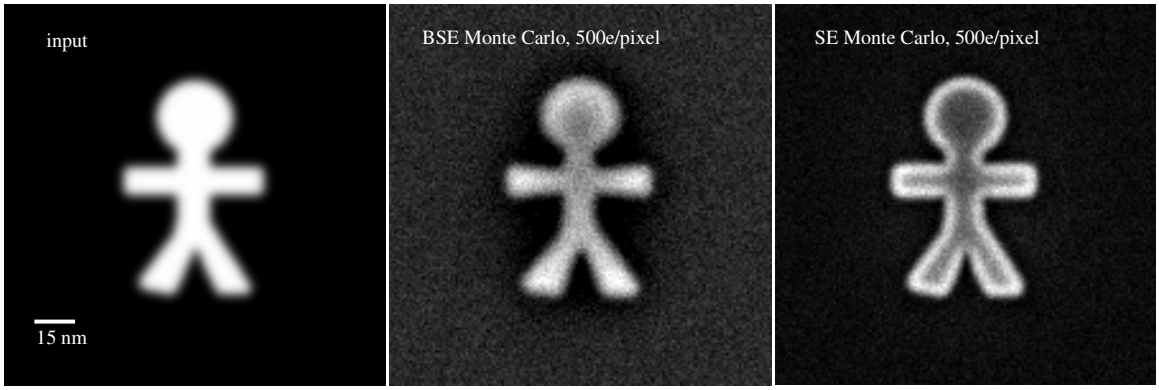

Figure 8: Filter bank model training data for 1nm/pixel resolution. Images are 150x150 pixels. 500 electrons/pixel were used to generate the training output. The input structure is 20nm high.

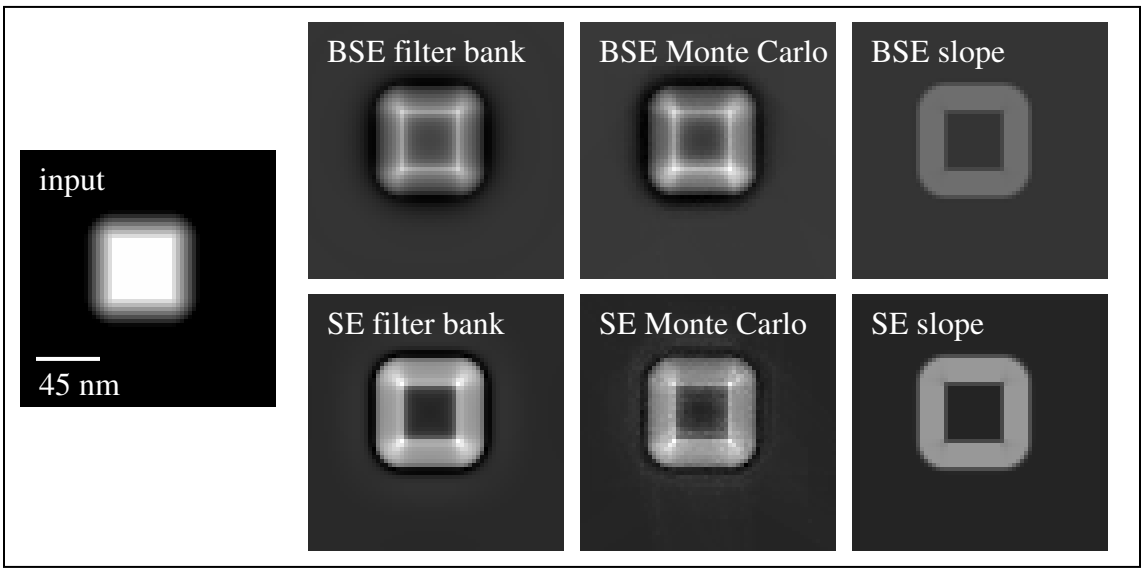

Figure 9: Monte Carlo simulation compared with a function of slope and the filter bank model on a 20nm high raised square 45nm wide at the top. Images are 64x64 pixels and 3nm/pixel.

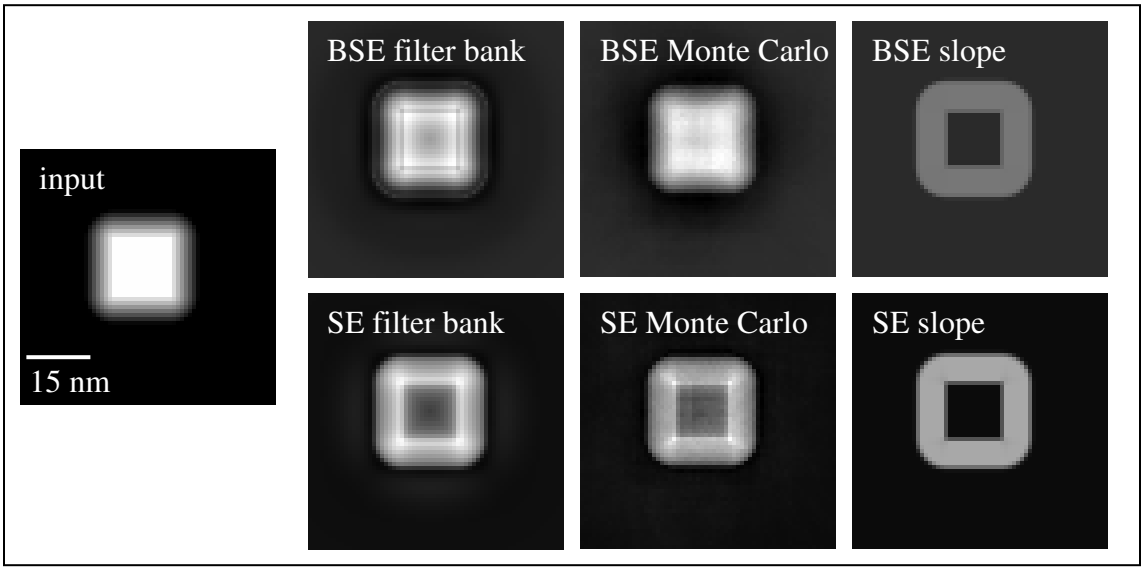

Figure 10: Monte Carlo simulation compared with a function of slope and the filter bank model on a 20nm high raised square 15nm wide at the top. Images are 64x64 pixels and 1nm/pixel.

Surface Optimization

Surface Model and Scale Space Reconstruction

Jones and Taylor introduced an approach called scale-space reconstruction in which a surface is represented by Gaussian basis functions and is constructed in a coarse to fine sequence. They also computed the SEM objective function gradient using convolution. Though the basic framework for our approach closely follows that described by Jones and Taylor [10], we generalize

the SEM shading function from one that depends only on slope with no notion of the scale of measurement to one that depends on slope and curvature at multiple scales.

The scale space representation of a surface, $H(x, y; \sigma)$, gives the height of the surface as a function of a position in the plane (x,y) for x=1..N_x and y=1..N_y and a scale parameter σ. The scale-space reconstruction algorithm computes the surface at each of a set of discrete scales in decreasing order $\sigma_{N-1}, \sigma_{N-2}, \ldots \sigma_0$ from the largest scale to the smallest scale. At any scale σ_k (k<N) the reconstruction is computed as the sum of the reconstruction at the next larger scale and an incremental update $h(x, y; \sigma_k)$ such that

$$H(x, y; \sigma_k) = \sum_{i=k..N} h(x, y; \sigma_i)$$

In the following descriptions, we abbreviate $h(x, y; \sigma_k)$ as h_k or $h_k(x, y)$ and $H(x, y; \sigma_k)$ as H_k or $H_k(x, y)$.

Building the surface up from large scale to small scale ensures that a smooth surface is constructed that explains the observed data. Although the solution found by this algorithm is not necessarily the smoothest among all surfaces that optimally explains the data, an upper bound for a lack-of-smoothness measure given by Jones and Taylor [10] suggests that the solution is nearly as smooth as possible.

Each function h_k is the convolution of a coefficient image C_k with a Gaussian basis function G_k^F and is defined as

$$h_k = C_k \otimes G_k^F = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} C_k(i, j) \cdot G_k^F(x - i, y - j)$$

$$\text{where } G_k^F(x, y) = \frac{1}{2\pi\sigma_k^2} e^{-(x^2+y^2)/(2\sigma_k^2)}$$

We define $H_k(x, y)$ in terms of the same (x,y) coordinates as the SEM image.

Objective Function and Its Gradient

Previously, the sum of squared differences between the SEM intensity and the intensity predicted by the candidate surface reconstruction was used as the objective function[10]. Assuming identical and independent Gaussian noise at each pixel in the SEM image, the surface that minimizes this objective function is equivalent to the maximum likelihood surface. We adopt this approach and define our objective function as

$$f(H_k) = \frac{1}{2} \sum_{x=1}^{N_x} \sum_{y=1}^{N_y} \left[\left(S(x, y) - S_{\text{FB}}(x, y) \right)^2 \right]$$

Because we use a gradient-based optimization method, it is necessary to calculate the gradient of the objective function in addition to the value of the objective function. The gradient of the

objective function (∇f) is the direction in parameter space in which the objective function increases the most. It is defined as

$$(\nabla f)_{i,j} = \frac{\partial f(H_k)}{\partial C_k(i,j)}$$

where (i,j) i=1..N_x, j=1..N_y index over the components of the gradient vector. Intuitively, this tells which Gaussian coefficients should go up, and which should go down (and by how much) to most quickly increase the likelihood of the surface estimate.

The method used to compute the gradient of the objective function relies on the fact that the facet model coefficients can be computed by a convolution[7]. We first give a mathematical derivation of a formula for the gradient. After this we describe the algorithm used to compute this formula. Throughout these descriptions we use $K^F_{m,n}$ to represent the convolution kernel used to compute a facet model coefficient at each pixel and $K_{m,n}(x,y)$ to represent the actual coefficient computed at (x,y) by the convolution. G^F_k represents the normalized 2-dimensional Gaussian kernel centered at 0 with width the same as that used for the surface basis functions at scale k (the superscript F in G^F_k is to avoid confusion with the gradient magnitude image G_n representing the gradient magnitude estimated for neighborhood size n).

Optimization of Surface at Each Scale

At each scale indexed by k, we minimize $f(H_k(x,y))$ using the non-linear Polak-Ribière conjugate gradient method as implemented in a program called CG+ [5,11]. Before the optimization, the Gaussian basis function coefficients for scale k are all initialized to 0.

Reconstruction Results

We tested the reconstruction algorithm using both synthetic and real SEM images. Example images from a reconstruction using a synthetic SEM image are shown in Figure 11. Images from the test using a real image are shown in Figure 12. The experimental SEM image, taken using a landing energy of 3keV, is of a specimen constructed of silicon by dip-pen lithography described previously[3]. It is difficult to evaluate the accuracy of the method in the case of real data because of the lack of ground truth but comparison with AFM data shows that the result is qualitatively correct. As shown in Figure 13, the reconstruction provides a much more accurate measurement within the plane of the SEM image than in the depth dimension.

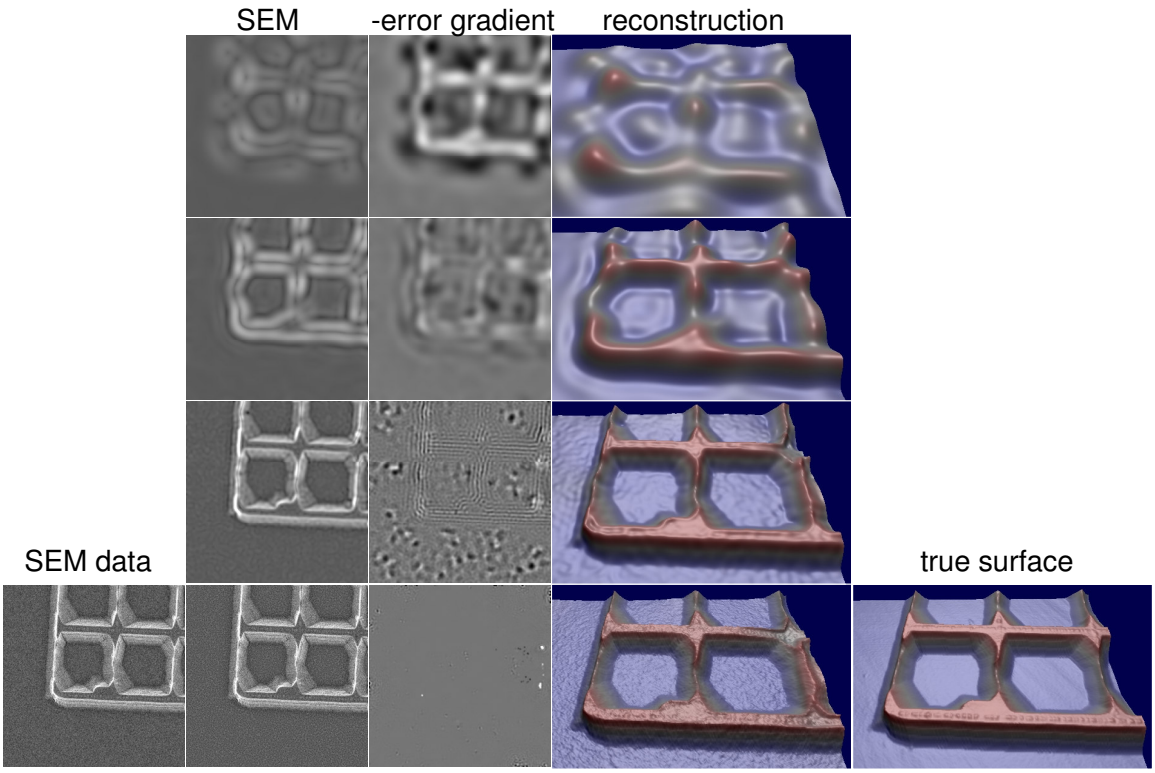

Figure 11: Example showing sequence of intermediate solutions for a test using synthetic data.

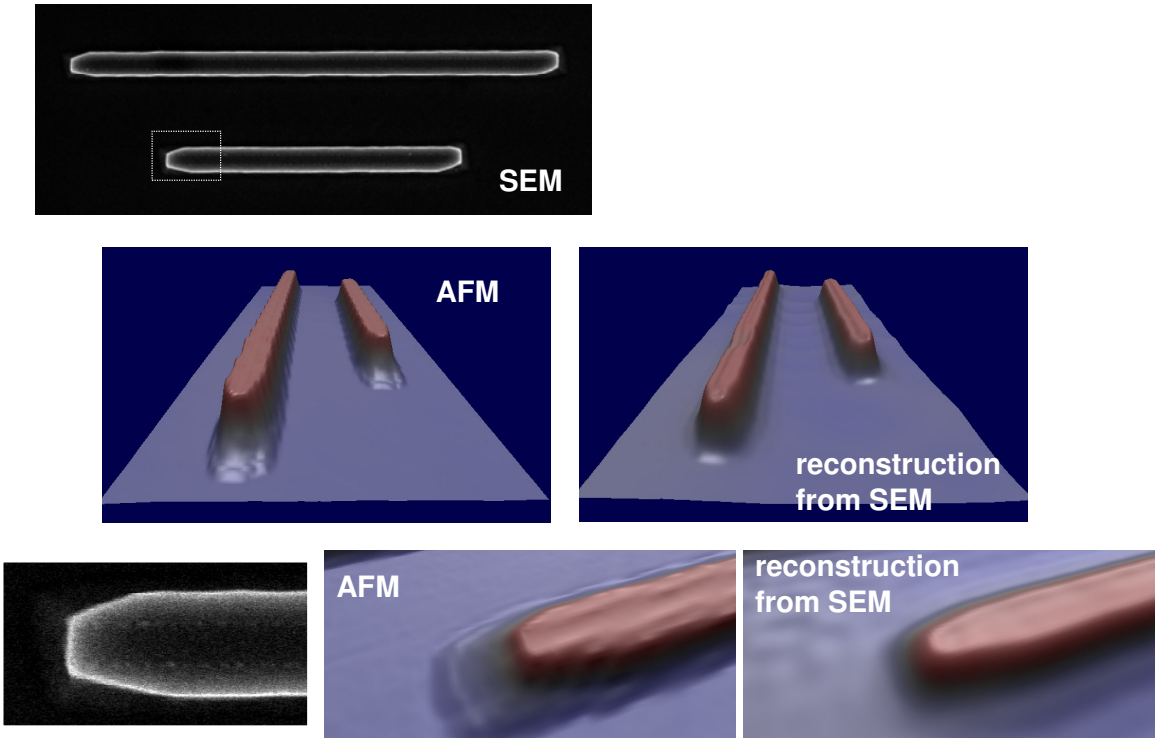

Figure 12: Reconstruction from a real SEM image

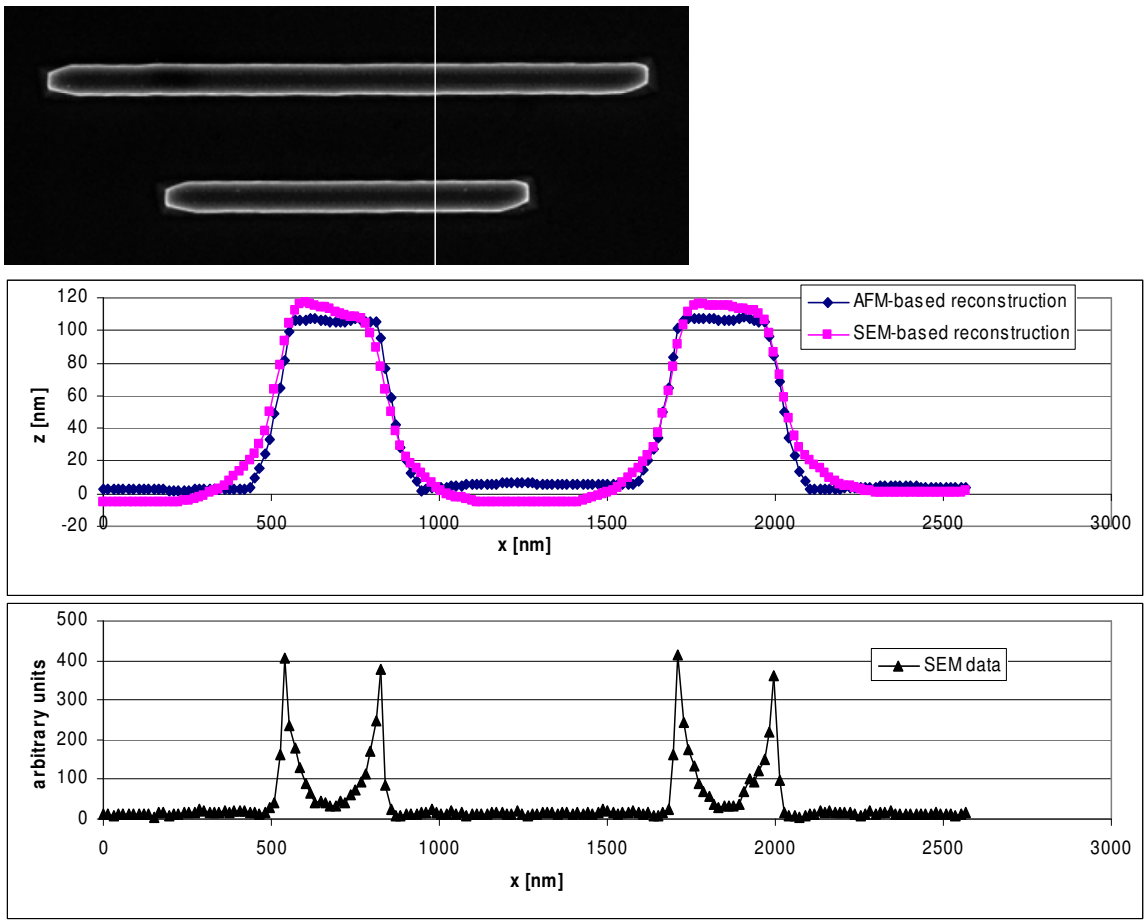

Figure 13: Cross section of reconstruction from SEM, reconstruction from AFM, and SEM data

Discussion

In our reconstruction method, the surface must be representable as a height field and this means that in general, surfaces with undercuts cannot be reconstructed. The SEM can sample below the surface of a specimen because electrons penetrate some distance before escaping and being detected. Also, electrons escaping from one surface may interact with other surfaces that are hidden from the incident electron beam. Thus there may be some information in an SEM image that is representative of such hidden surfaces. Allowing hidden surfaces is likely to make the reconstruction problem severely under-constrained. However, if it is known that hidden parts of the surface are constrained such that they are predictable from the visible surface, then the actual surface might be deduced from the reconstructed visible surface. For example, the profile of undercut edges might be the same everywhere, a reasonable assumption for surfaces generated by some etching processes.

The algorithm as implemented for this project is limited to specimens composed of a single material. This limitation simplifies the problem of modeling the SEM images because different materials will in general appear differently in the SEM and one would need to either be given or determine as part of the reconstruction the distribution of the different materials in the specimen. This algorithm could be extended to handle specimens composed of multiple materials by

extending the filter bank model to include filters that respond differently to different material types. A material map could be provided by an x-ray detector or considered as a free parameter of the specimen model. For manufactured samples consisting of multiple materials it is common for the different materials to be arranged in layers where each layer is contained in a unique range of heights. In reconstructing such a sample, one could use the height as a stand-in for knowing the material. The height of the surface model would automatically translate into a different material and this would essentially switch the behavior of the SEM model. Accurate simulation of the SEM image at material boundaries could still be a challenge but this may not be critical or there may be a simple interpolation scheme that would work.

Determining the height of the surface from an SEM image alone is expected to be an under-constrained problem. Even in the simpler case of shape-from-shading it has been proven that the height of certain critical points (local minima and maxima) on the surface must be known in order to sufficiently constrain the problem. We have not determined what would be a sufficient set of constraints but it is reasonable to expect similar requirements to those for shape-from-shading. For applications involving manufactured surfaces, there is usually some prior knowledge (such as CAD data) that could be incorporated into the algorithm to improve accuracy. Knowledge of the manufacturing process, including etching and deposition processes, could also be helpful in constraining the reconstructed surface.

We assume no charging effects in the SEM images but specimen charging typically occurs for specimens composed of an insulating material. Some SEMs reduce charging effects by introducing a gas into the specimen chamber or by placing a conducting grid just above the specimen but these are not commonly available. The charging results in electric fields that modify the trajectories of electrons and can create complicated variations in intensity. In order to overcome this limitation, it would be necessary to model the charging in an efficient way. Some subtle variations in intensity due to charging can be compensated for by the calibration procedure (determining the best fit combination of simulated SE and BSE images to match the real SEM image) and previous work describes a more flexible calibration method that might be used to reduce sensitivity to charging effects but this method is also restricted to subtle charging effects[2]. Existing methods for simulating charging effects require an impractical amount of time so, unless a fast approximation can be developed, it is difficult to see how our reconstruction approach could work in the case of severe charging.

The SEM shading is assumed to be rotation invariant. This is not a strict requirement of the method but in the described implementation, the SEM intensity is assumed to be a function of rotation invariant shape characteristics: gradient magnitude and the two principal curvatures estimated over a number of local neighborhood sizes. The assumption of rotation invariance is reasonable given the lack of any information about the specific detector geometry in the SEM. A secondary electron detector approximately measures SE escaping directly from the surface in all directions and BSE escaping in all directions that have generated additional SE through interactions with the walls of the specimen chamber. The detector may be more sensitive to electrons escaping in a particular direction due to asymmetry in the detector geometry, specimen chamber or electric field geometry but this information was not readily available. If a specimen with a known shape were available, it could be used to characterize the asymmetry in the SEM shading. Instead of training the filter bank model from a Monte Carlo simulation, the model could be trained from an actual SEM image of the known shape.

In our opinion, the most important obstacle to making this approach practical is that SEMs are not designed to make quantitative measurements of intensity. This is reflected in the fact that while the SEM detector amplifier settings may be manually set to adjust brightness and contrast, the actual values are not output by the control software making it very difficult to convert images from an SEM into meaningful physical units for later analysis. Images taken at different times typically will usually have different amplifier settings but without knowing these parameters it is difficult to compare them. Also, characterization of the detector sensitivity to electrons escaping in different directions with different energies is not typically available making it difficult to relate the output of a Monte Carlo simulation to a real image. This is only a technical problem and one could probably solve it by modifying an SEM to read out the amplifier settings and by doing many careful measurements of calibration specimens to characterize the detector. It may be useful to develop a calibration procedure that would work on multiple SEMs enabling quantitative analysis of image contrast and comparison of images across different tools.

REFERENCES

1. Anderson, E., Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, and D. Sorenson (1994). LAPACK Users' Guide - Release 2.0. Philadelphia, SIAM.

2. Davidson, M. P. and A. E. Vladar (1999). "An Inverse Scattering Approach to SEM Line Width Measurements." SPIE Conference on Metrology, Inspection, and Process Control for Microlithography XIII **3677**: 640-649.

3. Chien, F. S.-S., W.-F. Hsieh, S. Gwo, A.E. Vladar, and J.A. Dagata (2002). "Silicon nanostructures fabricated by scanning probe oxidation and tetra-methyl ammonium hydroxide etching." Journal of Applied Physics **91**(12): 10044-10050.

4. Firsova, A. A., L. Reimer, N. G. Ushakov, S. I. Zaitzez (1991). "Comparison of a Simple Model of BSE Signal Formation and Surface Reconstruction with Monte Carlo Calculations." Scanning **13**: 363-368.

5. Gilbert, J. C. and J. Nocedal (1992). "Global Convergence Properties of Conjugate Gradient Methods for Optimization." SIAM Journal on Optimization **2**: 21-42.

6. Goldstein, J. I., D. E. Newbury and P. Echlin (1992). Scanning Electron Microscopy and X-Ray Microanalysis, Plenum Pub Corp.

7. Haralick, R. M. and L. Watson (1981). "A Facet Model for Image Data." Computer Graphics and Image Processing **15**: 113-129.

8. Ikeuchi, K. and B. K. P. Horn (1981). "Numerical Shape from Shading and Occluding Boundaries." Artificial Intelligence **17**: 141-184.

9. Jollife, I. T. (1986). Principal Components Analysis. New York, Springer-Verlag.

10. Jones, A. G. and C. J. Taylor (1994). “Robust shape from shading.” Image and Vision Computing **12**(7): 411-421.

11. Liu, G., J. Nocedal and R. Waltz (2000). CG+ software. URL: http://www.ece.northwestern.edu/%7Enocedal/Software/CG+.1.1.tar.gz

12. Seeger, A., C. Fretzagias and R. M. Taylor II (2003). “Software Acceleration Techniques for the Simulation of SEM Images.” Scanning **25**(5): 264-273.

13. Seeger, A. (2004). “Surface Reconstruction from AFM and SEM Images.” UNC-Chapel Hill Computer Science Thesis, URL: http://www.cs.unc.edu/~seeger/publications/2004seeger_dissertation_sm.pdf