Microelectronic Engineering 87 (2010) 20–29

Contents lists available at ScienceDirect

Microelectronic Engineering

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

Three-dimensional level set based Bosch process simulations using ray tracing for flux calculation

Otmar Ertl *, Siegfried Selberherr

Institute for Microelectronics, TU Wien, Gußhausstraße 27-29/E360, A-1040 Wien, Austria

ARTICLE INFO

Article history:
Received 9 December 2008
Received in revised form 14 April 2009
Accepted 8 May 2009
Available online 19 May 2009

Keywords:
Bosch process
Deep reactive ion etching
Level set method
Monte Carlo
Ray tracing

ABSTRACT

This paper presents three-dimensional simulations of deep reactive ion etching processes, also known as Bosch processes. A Monte Carlo method, accelerated by ray tracing algorithms, is used to solve the transport equation, while advanced level set techniques are applied to describe the movement of the surface. With multiple level sets it is possible to describe accurately the different material layers which are involved in the process. All used algorithms are optimized in such a way, that the costs of computation time and memory scale more like with the surface size rather than with the size of the simulation domain. Finally the presented simulation techniques are used to simulate the etching of holes, whereas the influence of passivation/etching cycle times and hole diameters on the final profile is investigated.

1. Introduction

The invention of the Bosch process [1] enabled high aspect ratio etching by alternation of passivation and etching cycles and is used in semiconductor devices and microelectromechanical systems (MEMS) fabrication. In each cycle a chemically inert polymer layer is uniformly deposited using fluorocarbon gases. This passivation layer prevents the sidewalls from being attacked in the subsequent etching step (Fig. 1). By feeding a high frequency plasma with etch gases like SF_6, CF_4, or NF_3, a superposition of physical (directional) and chemical (isotropic) etching is obtained. This leads to a faster removal of the passivation layer at the bottom of the trench compared to the sidewalls due to the additional sputtering of the directional ions. After uncovering the substrate at the bottom chemical etching is dominant. Hence, in simple terms, in each cycle an isotropic etching process is started at the bottom of the trench. After many iterations profiles with high aspect ratios can be obtained. For optimal processing the passivation and etching cycle times have to be balanced. If the deposited passivation layer is too thin, the process time for the etching cycle has to be smaller to avoid the corrosion of the sidewalls increasing the number of required iterations. If the layer is too thick, the etching duration has to be increased resulting in a longer total process time. The choice of the process times has also an influence on the undulation of the sidewalls caused by the two-phase procedure. Computer simulations help to study parameter variations in order to optimize the process. Several simulators have been developed and applied to the Bosch process in the past [2–8].

A two-dimensional simulator using a string-cell hybrid method for surface evolution was presented in [2,3]. Therein a simplified model for the particle transport is used. Etching is modeled by a constant etching rate superposed by a directional etching term which is proportional to the incident ion flux. For the passivation cycle a perfect conformal deposition is assumed, which is equivalent to a constant surface velocity. However, this model is not able to describe the lag effect [9] appropriately. Therefore, a geometric shape factor was introduced [4], accounting for different trench widths.

A simulation with a more sophisticated transport model is presented in [5], where different sticking probabilities and higher order re-emissions of neutral particles are incorporated using the ballistic transport-reaction model (BTRM) [10,11]. Since the transport of neutrals to the surface is taken into account, the lag effect is inherently incorporated. The surface evolution is calculated using the level set method [12], which allows easy handling of topographic changes, while sub-grid resolution of the surface can be achieved. The transport equations, which result in surface integral equations, are solved by conventional integration techniques [13].

Another approach to calculate the particle transport is based on the Monte Carlo method which was first applied to Bosch process simulation in combination with a string-cell method for surface evolution in [6]. Many particle trajectories and their surface reactions are calculated to determine the surface rates.

Three-dimensional simulations of the Bosch process were recently reported in [7,8]. Both use simplified transport models and

* Corresponding author. Fax: +43 1 58801 36099.
E-mail addresses: ertl@iue.tuwien.ac.at (O. Ertl), selberherr@iue.tuwien.ac.at (S. Selberherr).

0167-9317/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.mee.2009.05.011

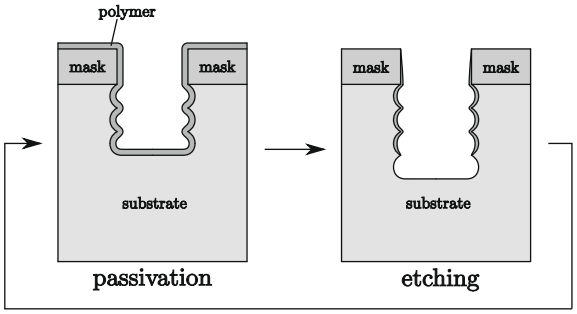

Fig. 1. A schematic illustration of the Bosch process. The deposition of a passivation layer protects the sidewalls during the subsequent etching cycle.

do not incorporate higher order re-emissions of neutrals. Instead, a uniform surface rate is assumed. The particle transport is calculated using conventional integral methods. For surface evolution a voxel-based method and the level set method are used, respectively.

In the following we describe a new approach for three-dimensional Bosch process simulations. We use advanced level set techniques to represent the geometry and also the different material regions. To determine the reaction rates on the surface we apply a Monte Carlo method, accelerated by ray tracing algorithms and parallelization.

2. Model

The scope of this paper is the demonstration of three-dimensional Bosch process simulations by means of fast computation techniques. For this purpose we use the model as given in [5], where a Bosch process with alternating flows of SF_6 and fluorocarbon gases is described, including a full set of parameters. In the following we summarize the model and discuss the solution of the governing equations.

2.1. Particle transport

The model is based on the BTRM [10,11], where the mean free path of particles is assumed to be much larger than the typical structure sizes of the geometry. Hence, particle–particle interactions can be neglected at feature scale. The arrival angle distributions of all particles are given at a certain plane $\mathscr{P}$, called source plane, just above the surface $\mathscr{S}$ (Fig. 2). For neutral particles a cosine-like arrival angle distribution (flux per solid angle)

$$\Gamma_{\mathrm{n}}^{\mathrm{src}}(\vec{t}) = F_{\mathrm{n}}^{\mathrm{src}} \frac{1}{\pi}(\vec{t}\cdot\vec{n}_{\mathscr{P}}), \tag{1}$$

is assumed, while a more directional distribution is used for ions

$$\Gamma_{\mathrm{i}}^{\mathrm{src}}(\vec{t}) = F_{\mathrm{i}}^{\mathrm{src}} \frac{\kappa+1}{2\pi}(\vec{t}\cdot\vec{n}_{\mathscr{P}})^{\kappa}. \tag{2}$$

Here $\vec{t}$ denotes the direction and $\vec{n}_{\mathscr{P}}$ is the normal on the source plane pointing to the surface. $F_{\mathrm{n}}^{\mathrm{src}}$ and $F_{\mathrm{i}}^{\mathrm{src}}$ are the total incoming fluxes of neutrals and ions, respectively. The parameter κ is used to model the narrow angular distribution of ions [14]. For $\kappa \gg 1$ this distribution is equivalent to a normal distribution for the arrival angles with a standard deviation of $\sigma = \frac{1}{\sqrt{\kappa}}$.

The arriving flux for neutrals can be obtained by solving the surface integral equation

$$F_{\mathrm{n}}(\vec{x}) = \int_{\mathscr{P}} \Gamma_{\mathrm{n}}^{\mathrm{src}}(\vec{t}) \frac{\mathrm{vis}(\vec{x},\vec{x}')(-\vec{t}\cdot\vec{n})}{\|\vec{x}-\vec{x}'\|^2} dA' + \int_{\mathscr{S}} F_{\mathrm{n}}(\vec{x}') \frac{1}{\pi}(\vec{t}\cdot\vec{n}')(1-\theta(\vec{x}')) \times \frac{\mathrm{vis}(\vec{x},\vec{x}')(-\vec{t}\cdot\vec{n})}{\|\vec{x}-\vec{x}'\|^2} dA', \tag{3}$$

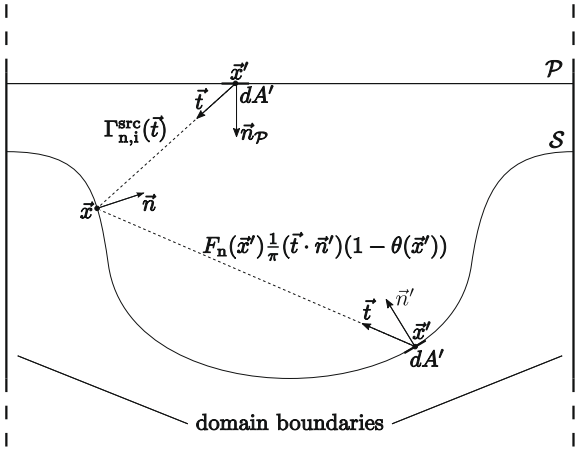

Fig. 2. The arriving flux at point $\vec{x}$ on the surface $\mathscr{S}$ is the sum of the flux coming directly from the source plane $\mathscr{P}$ and the re-emitted flux originating from the surface itself. In case of reflective or periodic domain boundaries the regions of integration $\mathscr{P}$ and $\mathscr{S}$ are finite.

with $\vec{t} = \frac{(\vec{x}-\vec{x}')}{\|\vec{x}-\vec{x}'\|}$. The first term describes the direct flux from the source. The second term is the flux which origins from the surface itself due to re-emission. $\mathrm{vis}(\vec{x},\vec{x}')$ is the visibility function which returns 1 or 0, if the surface points $\vec{x}$ and $\vec{x}'$ are in line of sight or not, respectively. For neutrals diffusive re-emission with a sticking probability $\theta(\vec{x})$ is assumed. During the passivation cycle the sticking probability is uniform, because the whole surface gets covered with the same type of material. During the etching cycle the sticking probability depends on the local material on the surface. For ions a constant sticking probability of 1 is assumed. Therefore, the ion flux can be written as

$$F_{\mathrm{i}}(\vec{x}) = \int_{\mathscr{P}} \Gamma_{\mathrm{i}}^{\mathrm{src}}(\vec{t}) \frac{\mathrm{vis}(\vec{x},\vec{x}')(-\vec{t}\cdot\vec{n})}{\|\vec{x}-\vec{x}'\|^2} dA'. \tag{4}$$

2.2. Surface kinetics

The deposition and etching rates in both cycles of the Bosch process are simply modeled by linear combinations of neutral and ion fluxes

$$R(\vec{x}) = \alpha F_{\mathrm{i}}(\vec{x}) + \beta F_{\mathrm{n}}(\vec{x}). \tag{5}$$

The coefficients α and β are model parameters, and in case of etching they depend on the exposed material. The model assumes that three different types of material are involved in the Bosch process: the mask, the substrate, and the passivation (polymer) layer.

The numeric values of all parameters for the passivation and the etching cycle, which we used for all simulations, are listed in Tables 1 and 2, respectively. Contrary to [5] we also consider mask etching by assuming a mask/substrate etch selectivity of 1:20. The coefficients α and β for the mask are adjusted accordingly. Furthermore, a spread of the arrival angles of ions is assumed ($\sigma = 2^\circ$).

To solve the above-described equations two different methods are necessary. One for tracking the surface and the different material regions over time and a second to determine the fluxes on the surface. In the following two sections the numerical framework is presented to accomplish these tasks.

3. Surface evolution

This section addresses the description of the geometry and of its evolution over time. For Bosch process simulation it is important that the profile evolution algorithm can handle different material regions. We use the level set method, since it allows a sub-cell

Table 1
The numeric values of the parameters used for the simulation of the passivation cycle.

Parameter	Value
σ	2°
F_n^{src}	2×10^{18} atoms/(cm^2 s)
F_i^{src}	3.125×10^{15} atoms/(cm^2 s)
α	10 Å^3/atom
β	0.5 Å^3/atom
θ	0.1

Table 2
The numeric values of the parameters used for the simulation of the etching cycle.

Parameter	Value
σ	2°
F_n^{src}	10^{19} atoms/(cm^2 s)
F_i^{src}	4.375×10^{15} atoms/(cm^2 s)
$\alpha_{polymer}$	125 Å^3/atom
$\alpha_{substrate}$	270 Å^3/atom
α_{mask}	13.5Å^3/atom
$\beta_{polymer}$	0.03 Å^3/atom
$\beta_{substrate}$	0.9 Å^3/atom
β_{mask}	0.045 Å^3/atom
$\theta_{polymer}$	0.1
$\theta_{substrate}$	0.2
θ_{mask}	0.2

accurate representation of the surface, while topographic changes are handled inherently.

3.1. Level set method

The basic idea of the level set method is to describe a boundary by means of a continuous function [12]. For a given surface $\mathscr{S}$ a level set function Φ is initialized in such a way that $\mathscr{S}$ can be obtained as its zero level set

$$\mathscr{S} = \{\vec{x} : \Phi(\vec{x}) = 0\}. \tag{6}$$

The advantage of this representation is that the propagation of a boundary driven by a given velocity field $V(\vec{x})$ can be easily determined by solving the level set equation

$$\frac{\partial \Phi}{\partial t} + V(\vec{x})\|\nabla \Phi\| = 0. \tag{7}$$

If discretized on a Cartesian grid, this equation can be easily solved by means of simple finite difference schemes. To guarantee a stable time integration a Courant–Friedrichs–Lewy (CFL) condition has to be fulfilled, which restricts the maximum advancement of the surface per time step. For our calculations we used a maximum step size of 0.1 grid spacings.

In topography simulations the surface velocities are only defined on the surface. Therefore, to get the required surface velocity field an extension technique has to be applied [15]. To keep the level set function a signed distance function, it was proposed to take for each grid point the surface velocity of its closest surface point [16]. Later, we will discuss this mapping and more generally, how to couple the transport equation solver with the level set method.

3.2. Sparse field level set method

The original level set method shows a non-linear scaling of computation time and memory consumption with the surface size, since the level set function is stored and integrated over time for all grid points of the simulation domain. A linear scaling law for the computation time was achieved by the narrow band level set method which only considers active grid points close to the surface for time integration. The approach makes use of the fact that the level set values of grid points far away do not influence the actual position of the surface. A further enhancement of this method is the sparse field level set method [17], which has the advantage that only a single layer of active grid points, namely those with an absolute level set value less than 0.5, must be considered for time integration. Only for those grid points the surface velocity field has to be known, making the mapping from the surface very easy. Another advantage of the sparse field level set method is that periodic re-initializations of the level set function as needed for conventional level set methods are not necessary. The level set values at neighboring grid points, which are required to determine the derivatives of the level set function, are obtained by a simple and fast update scheme.

3.3. Run-length-encoding

To reduce the memory consumption for storing a level set function the recently developed hierarchical run-length-encoding (HRLE) data structure [18] was implemented. Only for grid points close to the surface the explicit level set values are stored. For all other grid points only the signs of the level set function are stored using run-length compression. The availability of the signs of all grid points is very useful, since the sign of the level set reveals on which side of the level set a grid point is located.

The HRLE data structure enables fast sequential and random access to grid points with constant and sub-logarithmic complexity, respectively. In combination with the sparse field level set method a perfect linearly scaling level set algorithm in terms of surface size can be realized.

Boolean operations like union or intersection of two regions play a role in multi-level-set methods, where different material regions are represented by more than one level set. These operations can be expressed as minimum or maximum of the corresponding level set functions [19], which can be computed with an optimal linear complexity using the HRLE data structure.

3.4. Multiple materials

The simulation of the Bosch process requires accurate handling of all three involved material regions: The substrate, the mask, and the passivation layer, labeled by $\Omega_{substrate}$, Ω_{mask}, and $\Omega_{polymer}$, respectively. Three level sets are used to represent the whole geometry. They are defined as follows

$$\begin{aligned} \Phi_1(\vec{x}) \leqslant 0 \quad &\Longleftrightarrow \quad \vec{x} \in \Omega_{substrate}, \\ \Phi_2(\vec{x}) \leqslant 0 \quad &\Longleftrightarrow \quad \vec{x} \in \Omega_{substrate} \cup \Omega_{mask}, \\ \Phi(\vec{x}) \leqslant 0 \quad &\Longleftrightarrow \quad \vec{x} \in \Omega_{substrate} \cup \Omega_{mask} \cup \Omega_{polymer}. \end{aligned} \tag{8}$$

Here the zero level set of Φ is equal to the surface, while those of Φ_1 and Φ_2 can be assigned to interfaces. The representation of a structure consisting of three different material regions is illustrated in Fig. 3. If the level set functions are initialized using a metric function, the inequation

$$\Phi_1(\vec{x}) \geqslant \Phi_2(\vec{x}) \geqslant \Phi(\vec{x}) \tag{9}$$

holds.

Obviously there are other alternatives to choose the level sets to represent this structure. For example, it is possible to describe each material region by one enclosing level set. However, by nature, if the level set functions are discretized on a Cartesian grid, it is not possible to resolve layers accurately which are thinner than one grid spacing. Therefore, it is possible that the passivation layer suddenly disappears, if a certain thinness is reached during the etching cycle. Consequently, the etching of the underlying

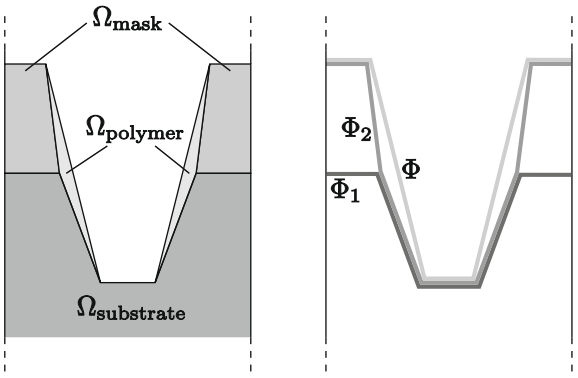

Fig. 3. The different material regions which have to be considered during a Bosch process simulation (left) and their representation using level sets (right).

substrate starts too early, leading to wrong profiles. This effect is intensified due to the etch rate ratio and due to the multiple repetitions during the Bosch process. Therefore, it is very important to resolve the passivation layer accurately. With the level set configuration as defined in (8) also very thin layers can be resolved.

A time integration step consists of solving the level set equation for the surface level set function Φ and subsequent adapting the interface level sets Φ_1 and Φ_2 using the boolean operation

$$\Phi_i^{(t+\Delta t)}(\vec{x}) = \max\left(\Phi_i^{(t)}(\vec{x}), \Phi^{(t+\Delta t)}(\vec{x})\right). \tag{10}$$

It should be noted that this adaption rule maintains inequation (9). As mentioned previously, the maximum of two level set functions can be constructed very efficiently using the HRLE data structure.

The type of material at a certain surface point $\vec{x}$ can be obtained from the level set functions as follows:

$$\Phi_1(\vec{x}) = \Phi_2(\vec{x}) = \Phi(\vec{x}) \Rightarrow \text{substrate}, \tag{11}$$
$$\Phi_1(\vec{x}) > \Phi_2(\vec{x}) = \Phi(\vec{x}) \Rightarrow \text{mask}, \tag{12}$$
$$\Phi_1(\vec{x}) \geqslant \Phi_2(\vec{x}) > \Phi(\vec{x}) \Rightarrow \text{polymer}. \tag{13}$$

The surface velocities of different materials are taken into account during time integration. If the surface front reaches another material within a time step (during the etching cycle), the different surface rates are incorporated adequately. A detailed description of this methodology can be found in [20].

4. Flux calculation

Every time step the surface rates have to be determined to enable the profile evolution calculation using the level set method. For this purpose the flux Eq. (3) has to be solved. Especially in three dimensions it is crucial to use fast techniques and algorithms to speed up the whole topography simulation.

Conventionally, this surface integral equation is solved by discretization of the surface using triangle [21] or voxel elements [22], resulting in a system of linear equations. The system matrix contains the visibility factors which have to determined for each pair of elements. If they are visible from each other, the corresponding system matrix entry is non-zero. Generally the system matrix is dense, which to set up and to solve is computational intensive, since at least a quadratic scaling law with surface size can be expected. The visibility check can lead to an even worse scaling [23].

The particle fluxes are often calculated using an explicit representation of the surface. However, surface extraction algorithms like the marching cubes algorithm [24] result in a huge number of surface elements, revealing the importance of a well scaling algorithm. A way to reduce the number of elements is coarsening of the resulting surface mesh [25]. However, this approach does not only reduce the number of elements and hence the computation time, it also reduces the resolution of the flux. This is a problem, since even on plane regions of the surface the flux can change abruptly due to shadowing. Therefore, coarsening is limited and the unfavorable scaling law is maintained.

4.1. Ray tracing

Since ballistic transport of particles is assumed, the flux calculation is quite analogous to rendering a scene in computer graphics. Due to the ballistic transport of particles the propagation is linear like that of light rays. A widely applied technique to get a realistic picture of a three-dimensional scene is ray tracing [26], a Monte Carlo technique, where a huge number of light rays is simulated. Applied correspondingly to our problem, many particle trajectories are calculated. Whenever, a particle hits the surface it contributes locally to the surface. Thus, the main task is to find the first intersections of rays with the surface. Spatial subdivision can reduce the complexity of finding the first intersection to an expected logarithmic scaling $\mathcal{O}(\log N)$ [27], where N is the number of surface discretization elements, or in our case the number of surface grid cells. Grid cells having corners with different signed level set values contain parts of the surface, which consequently have to be checked for intersection. To optimize the data structure for fast traversals we use binary subdivision along grid planes with simultaneous consideration of a cost function based on surface area heuristics (SAH) as described in [28]. As exemplified in Fig. 4 only a small number of boxes have to be traversed to find the intersection with the surface. Ray tracing can be directly applied to the level set surface representation. The intersection can be found by tri-linear interpolation of the level set function within grid cells and finding the zero-crossing along the particle ray [29].

Since ray tracing is a statistical method, its accuracy strongly depends on the number of simulated rays. To obtain a desired accuracy the number of simulated particle trajectories has to scale with the surface size, to keep the statistical events per unit area constant. Therefore an overall complexity of $\mathcal{O}(N \log N)$ can be achieved using ray tracing, which is a much better scaling law than that for solving the flux balance equation directly.

To be able to determine the incident flux for a certain surface point a reference area has to be defined to relate the number of incidences to the local fluxes. Each particle hitting a reference area

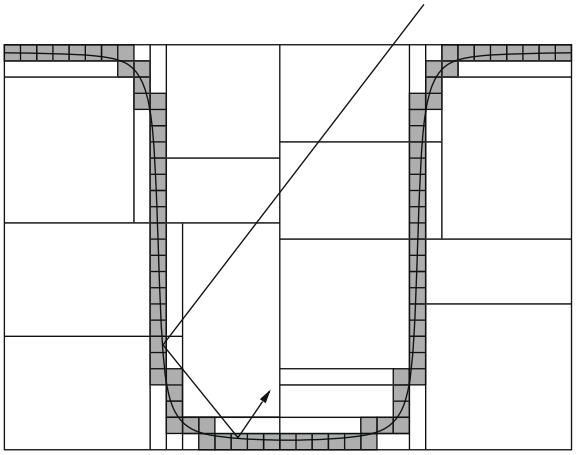

Fig. 4. Spatial subdivision accelerates the calculation of particle trajectories. Within the surface cells (gray) tri-linear interpolation is used to find the intersection with the surface.

of size A^{ref} contributes to the local flux of the corresponding surface point following

$$\Delta F = \frac{F^{\text{src}}}{n \cdot A^{\text{ref}}}. \tag{14}$$

Here n denotes the number of simulated particles which are launched per unit area from the source plane. In principle, these reference areas can be arbitrarily shaped plain areas. For example, the triangles of a surface mesh, or, as we will describe in the next section, tangential disks can be used as reference areas. It is only important that they are localized around the surface point for which the flux has to be determined. However, it is not necessary that the sum of all reference areas equals the real physical area of the surface. In particular, it is even possible that they overlap. In this case an incident particle can contribute to the fluxes of various reference areas following (14).

According to our model neutral particles have a sticking probability much less than 1. Hence, also higher order re-emissions have to be incorporated. This can be performed by continuing the particle trajectory calculation in compliance with the applied re-emission law. The particle trajectory is stopped with a probability equal to the sticking probability. Elsewise, a new direction is randomly chosen in accordance to the used re-emission model, and the particle is re-emitted. In contrast to reality where a particle only contributes to the surface velocity at the point where it finally remains sticking, a particle trajectory contributes to the flux each time it reaches the surface, independent from being re-emitted or not. Hence, more statistical events are generated and a better accuracy is obtained.

Alternatively, instead of re-emitting a particle following the complementary sticking probability, it is also possible to assign a weight factor w to the particle as described in [30]. Starting with an initial value $w^{(0)} = 1$ the particle contributes to the local flux according to its weight factor

$$\Delta F = w \cdot \frac{F^{\text{src}}}{n \cdot A^{\text{ref}}}. \tag{15}$$

In contrast to the first method the particle is always re-emitted, however, with a reduced weight factor

$$w^{(k+1)} = w^{(k)} \cdot (1 - \theta_{\text{stick}}). \tag{16}$$

This method is equivalent with the first one, because the expected contribution to the local flux of a particle which is re-emitted k times is the same in both cases

$$\langle \Delta F \rangle = \rho_k \cdot \frac{F^{\text{src}}}{n \cdot A^{\text{ref}}} = w^{(k)} \cdot \frac{F^{\text{src}}}{n \cdot A^{\text{ref}}}. \tag{17}$$

Here $\rho_k = (1 - \theta_{\text{stick}})^k$ denotes the probability that a particle is re-emitted k times. The trajectory calculation is stopped, if the weight factor falls under a certain fraction $w < w_{\text{limit}}$, or, if the particle leaves the simulation domain upwards. In our simulations we used $w_{\text{limit}} = 10^{-3}$. The error introduced by aborting the particle trajectory is given by w_{limit}. Usually, the error is smaller, because the particle leaves the simulation domain after a couple of re-emissions before reaching this critical weight. For the latter method a better accuracy can be expected especially at regions which are unlikely reached by lower order particles.

4.2. Coupling with surface evolution

In the following we describe how to link the ray tracing algorithm for flux calculation with the level set method. On the one hand side the surface velocities at all active grid points have to be determined as needed for the sparse field level set method. On the other hand side reference areas for the flux calculation using ray tracing have to be defined. In [31] it was proposed to choose for each active grid point an environment around its closest surface point. However, this approach requires a triangulation of the surface.

As already mentioned ray tracing can be performed directly using the implicit level set surface representation. To avoid an explicit surface representation at all, which increases not only the memory requirements but also the calculation time due to the surface extraction algorithm, a disk with predefined radius ρ is set up for each active grid point. These disks serve as reference areas ($A^{\text{ref}} = \pi\rho^2$) for the calculation of the fluxes for the corresponding active grid points. Their positions are chosen in such a way that they are tangential to the surface at the closest surface point of the corresponding grid point. The closest surface point of a grid point $\vec{p}$ can be approximated by

$$\vec{p}' = \vec{p} + d \cdot \vec{n} = \vec{p} + \frac{\Phi(\vec{p})}{\|\nabla\Phi(\vec{p})\|} \cdot \frac{\nabla\Phi(\vec{p})}{\|\nabla\Phi(\vec{p})\|}. \tag{18}$$

d denotes the distance to the closest surface point and $\vec{n}$ is the normal vector. As applied, both expressions can be estimated from the surface level set function Φ [17]. Fig. 5 shows the tangential disk for an active grid point. Whenever a particle hits the disk, it contributes to the flux of the corresponding grid point according to (15). As shown it might be necessary to continue the trajectory calculation after finding the intersection with the surface to ensure a proper calculation of the fluxes. In our simulations the particle rays are extended for 3 grid spacings from the intersection point. Then, in case of a neutral particle, for which diffusive re-emission is assumed, the trajectory is continued from the memorized intersection point. The direction is randomized in accordance with diffusive re-emission. The surface normal is obtained from the tri-linear interpolated level set function in the corresponding grid cell.

Since for all active grid points $\vec{p}$, $|\Phi(\vec{p})| \leqslant \frac{1}{2}$ and for the gradient $\|\nabla\Phi(\vec{p})\| \geqslant 1$ holds except for some special cases, the distance to the surface is always within a half grid spacing $|d| \leqslant \frac{1}{2}$. Thus, if the radius is chosen in such a manner that

$$\rho \leqslant \sqrt{1 - \left(\frac{1}{2}\right)^2} \approx 0.866, \tag{19}$$

the disk is almost always within the 8 grid cells which are adjacent to the corresponding active grid point. In very rare cases the distance d has to be reduced to fit the disk into these cells. Hence, the same data structure can be used as for the tri-linear interpolation, which requires for each surface cell links to all its corners in

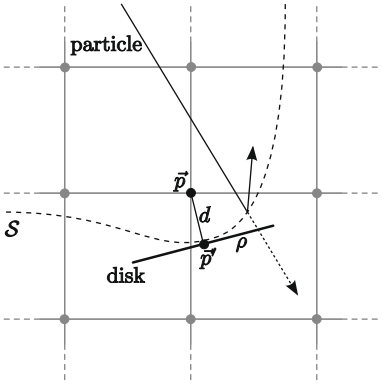

Fig. 5. The tangential disk for an active grid point $\vec{p}$. All particles hitting the disk contribute to the local flux of the grid point. Due to the curvature of the surface it can be necessary to continue the trajectory calculation for a couple of grid spacings (dashed) to calculate the flux correctly. However, re-emission takes place at the surface intersection point.

order to access the corresponding level set values. Therefore, it is sufficient within a grid cell to check its 8 corners, if they are active and if their corresponding tangential disks are hit by the particle.

The choice of the disk size is a compromise between statistical and spatial accuracy. If the disk size is too large, the calculated fluxes are spatially averaged. In case of disk radii much larger than the grid spacing the spatial resolution of the flux, and consequently that of the surface velocity, might be not sufficient for an accurate time evolution of the surface. Larger disks also intensify the previously mentioned problem at surface regions with larger curvature, resulting in additional errors. Furthermore, if (19) is not satisfied, the disks of much more grid points have to be checked for intersections, which slows down the ray tracing algorithm and also requires additional data structures. Otherwise, if the disk size is too small, only a few particle rays hit the disk leading to a poor statistical accuracy of the fluxes. The statistical errors are inversely proportional to the chosen disk radius. A good choice is a value close to the upper limit in (19). We compared simulations with $\rho = 0.4$ and $\rho = 0.8$, where 4 times more particles are used for the first case to obtain the same statistical accuracy. However, we could not observe an improvement of the simulation result for the smaller radius. Consequently, it does not make sense to further decrease the radius. In our simulations $\rho = 0.8$ is used which seems to be a good choice.

5. Simulation

Assuming that small changes in geometry have only a small impact on the local fluxes, which is also known as pseudo-steady state assumption [10], the flux can be considered constant during the whole time step. Therefore a simulator can simply pass the surface velocities obtained from the calculated fluxes to the profile evolution algorithm.

5.1. Algorithm

An overview of the whole algorithm is shown in Fig. 6. After reading the initial geometry a distance transformation initializes the level set functions. Then a loop over the flux calculation and the profile evolution modules is started, until the final time is reached.

Within the flux calculation part the tangential disks are set up first. Then all cells are determined which contain parts of the surface or parts of the tangential disks. Links to their corner grid points are stored, since they are necessary for the tri-linear interpolation and for the ray-disk intersection tests. Subsequently, the simulation domain is subdivided into boxes in such a manner that all surface cells represent individual boxes. This additional data structure speeds up the ray tracer which calculates the particle fluxes for all active grid points. Within the profile evolution module the surface velocities are computed from the fluxes. Then the maximum possible time step according to the CFL-condition is determined and used for integrating the level set Eq. (7) over time. Afterwards the interface level sets Φ_1 and Φ_2 are adjusted accordingly (10).

After the final time is reached, the marching cubes algorithm is applied to extract explicit representations of the surface and the interface level sets, which are used for visualization.

5.2. Parallelization

For good statistical accuracy a huge number of particles has to be simulated each time step. Despite the application of fast algorithms, the simulator spends most of the time with ray tracing. To resolve this bottle neck we use parallelization. Since individual

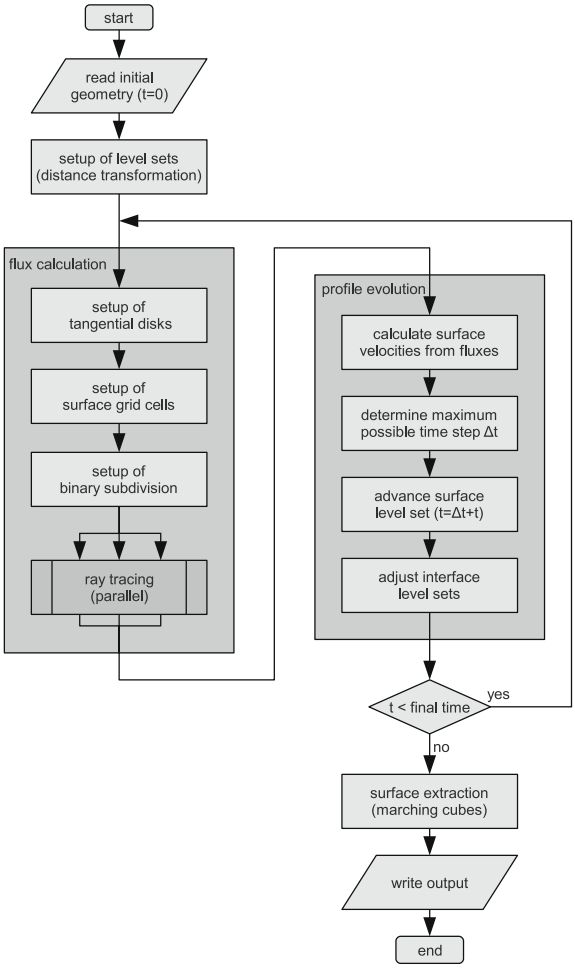

Fig. 6. The simulation algorithm.

trajectories are independent from each other due to ballistic transport, their calculation can be simply distributed among multiple cores. Especially on shared memory architectures, which are getting more and more popular due to the increasing number of processor cores, the parallelization is straightforward using OpenMP [32]. To get for all threads independent streams of random numbers, which are required for ray tracing, we used the Scalable Parallel Random Number Generators Library (SPRNG) [33]. The ray tracing speedup shows a very good scaling with the number of applied CPUs (Fig. 7).

6. Results and discussion

For all in the following presented simulations we use the same parameters, as described in Section 2 for the passivation and the etching cycle. For all calculations reflective boundary conditions are used for both lateral directions. If not mentioned differently, the grid spacing is 25 nm. The radii of the tangential disks are set to 0.8 grid spacings.

6.1. Process time variations

The effect of different passivation and etching cycle durations is studied on a structure existing of a substrate and a 1 μm thick mask, which has a cylindrical hole with diameter 2.5 μm. Despite the rotational symmetry this problem can not be straightforwardly reduced to two dimensions. The introduction of cylindrical coordi-

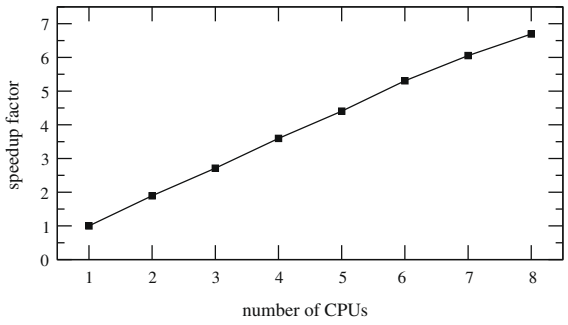

Fig. 7. The speedup of ray tracing versus the number of used CPUs.

nates leads to non-linear particle trajectories, which makes the determination of the visibilities in the particle transport equation (3) much more difficult. For convex holes, where all points are visible from each other, the solution of the transport equation using cylindrical coordinates was demonstrated in [13]. However, due to the rippled, non-convex side walls of the hole, which evolve during the Bosch process, this method can not be applied.

In three dimensions the simulation domain can be reduced to a quarter due to the reflective boundary conditions and the twofold reflection symmetry of the hole. However, to proof the symmetry of the solution and to avoid reflections to generate our final visualizations the process is simulated on half of the domain, which was discretized using a grid with lateral extensions 140×70. 100 particles for each involved species are launched per grid unit area each time step from the open boundary ($n = 100$). Hence, in total 1.96 million particle trajectories are calculated.

The final profiles after 20 cycles with different process times for deposition (5 s and 8 s) and etching (11 s and 13 s) are given in Fig. 8. The results show the influence of the process time on the depths of the holes, tilt angles of the side walls, and the resulting polymer layers. Since also mask etching is incorporated, its final thickness can also be studied. Such simulations can help to find the optimal process parameters.

6.2. Lag effect

Next the influence of the hole diameter on the final profile is investigated. A Bosch process with 6 s passivation followed by 12 s etching cycles is applied on a 1 μm thick perforated mask with cylindrical holes of varying diameters (2.5 μm, 2 μm, 1.5 μm, 1 μm, and 0.5 μm).

The simulation domain is resolved on a grid with extensions 500×140 proving the practicability of the applied techniques on larger geometries. Despite this large simulation domain the total memory consumption does not exceed 300 MB during the whole simulation thanks to the applied adaptive memory saving data structures.

For each of both species 100 particle trajectories are calculated per grid unit area ($n = 100$), which gives 12.5 millions in total for each time step. Using 8 cores of AMD Opteron 2222 processors (3 GHz) the total computation time is about 2 days. 6480 time steps are necessary to simulate all 20 cycles of the Bosch process. The sequential part of the algorithm takes 3.4 s and the parallelized ray tracing takes 24 s in average. The runtimes increase continuously during the whole simulation due to the increasing depths of the holes and the increasing surface area. However, the runtime of these simulation can be reduced drastically by lowering the accuracy, which is described in the next section.

Fig. 9 shows the final profile after 20 cycles. The different etching depths due to the lag effect can be clearly seen. With increasing aspect ratio the effective etching rate decreases.

To analyze the reason of the lag effect in more detail, the ion and neutral fluxes are calculated at the bottom center of cylindrical holes for various aspect ratios $x = d/2r$. Here d denotes the depth and r the radius of the hole. The ion fluxes obtained by ray tracing are in very good agreement with those calculated analytically (Fig. 10). The analytical expression

$$F_\mathrm{i} = F_\mathrm{i}^\mathrm{src}\left(1 - \left(\frac{2x}{\sqrt{1+4x^2}}\right)^{\kappa+1}\right) \tag{20}$$

can be derived from (2) by integration over the open solid angle. For the calculation of the neutral flux the sticking coefficient is set to

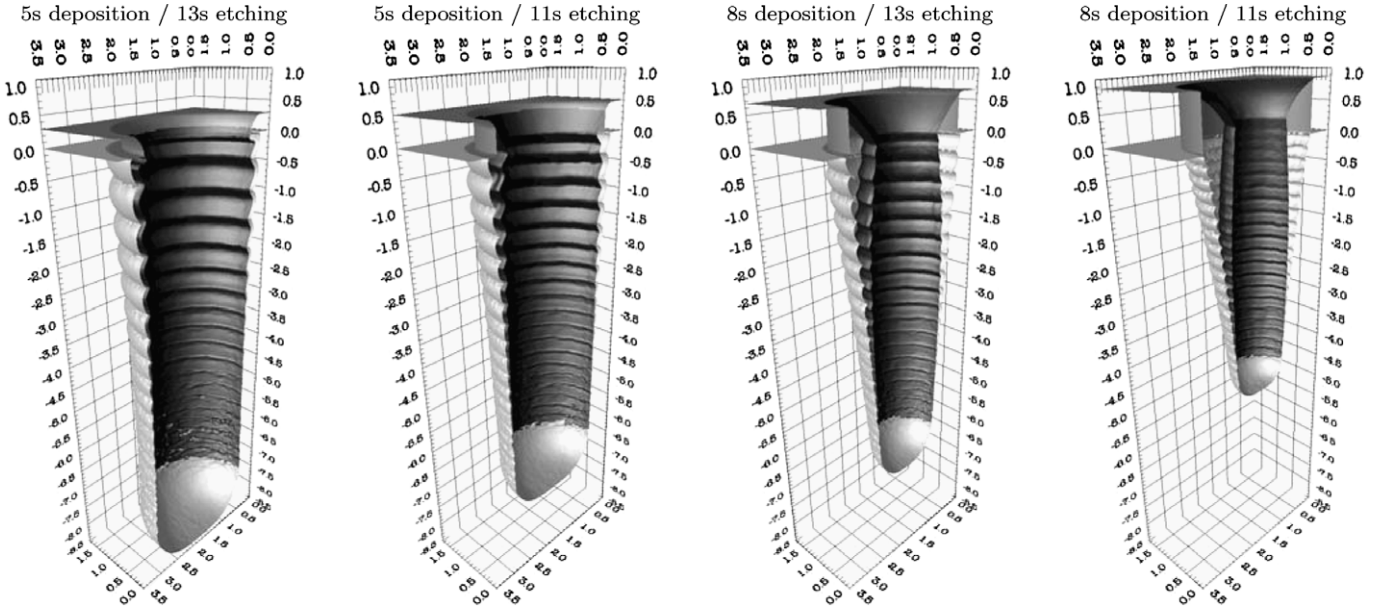

Fig. 8. The final profiles after 20 cycles for different combinations of deposition and etching process times. The zero level sets of the functions Φ_1 (light gray), Φ_2 (dark gray), and Φ (black) are visualized. Lengths are given in μm. The grid spacing is 25 nm, which corresponds to a grid with lateral extensions 140×70.

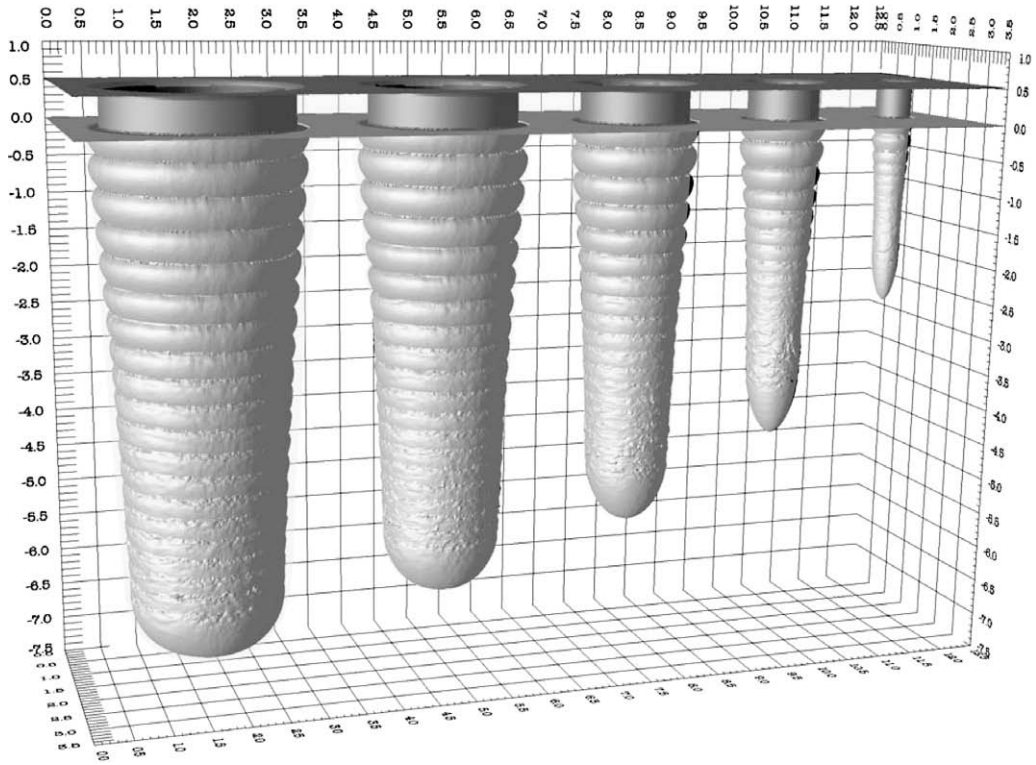

Fig. 9. Deep reactive ion etching of holes with varying diameters (2.5 μm, 2 μm, 1.5 μm, 1 μm, and 0.5 μm). The different depths are a result of the lag effect. The structure is resolved on a grid with lateral extensions 500×140.

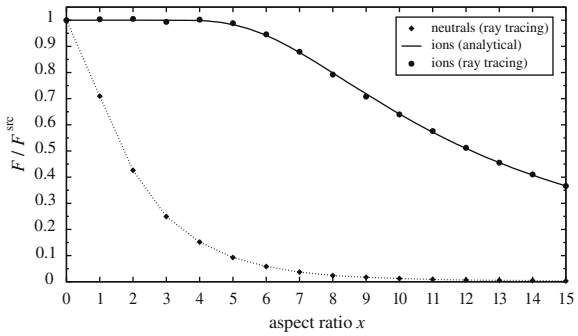

Fig. 10. The characteristic dependence of the neutral and ion fluxes at the bottom center on the aspect ratio.

0.1, which corresponds to the sticking probability of neutrals on the passivation layer as used in our model. The results show that the neutral flux is much more affected by the aspect ratio than the directional ion flux ($\sigma = 2^\circ$). With increasing depth the hole surface area increases, leading to a smaller fraction of particles, which remain sticking at the bottom and not at the sidewalls.

According to our Bosch process model (5) and Table 1 the neutral flux is the main contribution to the deposition rate of the passivation layer. Hence, with increasing aspect ratio the thickness of the deposited passivation layer decreases due to the smaller neutral fluxes. However, the ion flux is not alike reduced. As consequence, the passivation layer is faster etched through. The ion flux countervails the lag effect, because the substrate is attacked earlier in the etching cycle for larger aspect ratios. However, this head start is more than compensated by the larger substrate etch rate for smaller aspect ratios due to the larger neutral fluxes.

6.3. Accuracy vs. runtime

Inaccuracies of the final profiles are mainly caused by statistical and spatial discretization errors. Both can be reduced at the expense of the runtime.

Statistical errors can be reduced by simulating more particles. Principally, the statistical accuracy of the final profiles strongly depends on the total number of particles, which are calculated during the whole simulation. Therefore, larger time steps also require the simulation of more particles each time step to obtain final profiles of similar quality. Time steps can only be increased by choosing a weaker CFL-criterion, hence allowing larger advancements of the surface, at the expense of the time resolution. However, if the total number of simulated particles is kept constant, the runtime is only marginally reduced, because for good accuracies the simulator spends most of the time with ray tracing anyway. As trade-off we limit the maximum advancement of the surface by 0.1 grid spacings, as already mentioned earlier.

One way to improve the total runtime considerably apart from using more and faster CPUs is to simulate less particles at the expense of the statistical accuracy. The influence of the number of particles n, which are launched per unit area from the top of the simulation domain, on accuracy and runtime is studied on the basis of a 6 s/12 s Bosch process simulation. Fig. 11 shows the final profiles for $n = 1$, $n = 10$, $n = 100$, and $n = 1000$ after 20 cycles. Interestingly, even for the least accurate case quite good results are obtained. Although the final surface is very rough the qualitative characteristics are maintained. The etched hole is only 3.8% deeper than for $n = 1000$.

From measurements of the total runtime T for different n and different number of CPUs we obtain the relation

$$T \approx N_{\text{steps}} \cdot (n/p \cdot 0.31\text{s} + 0.52\text{s}). \tag{21}$$

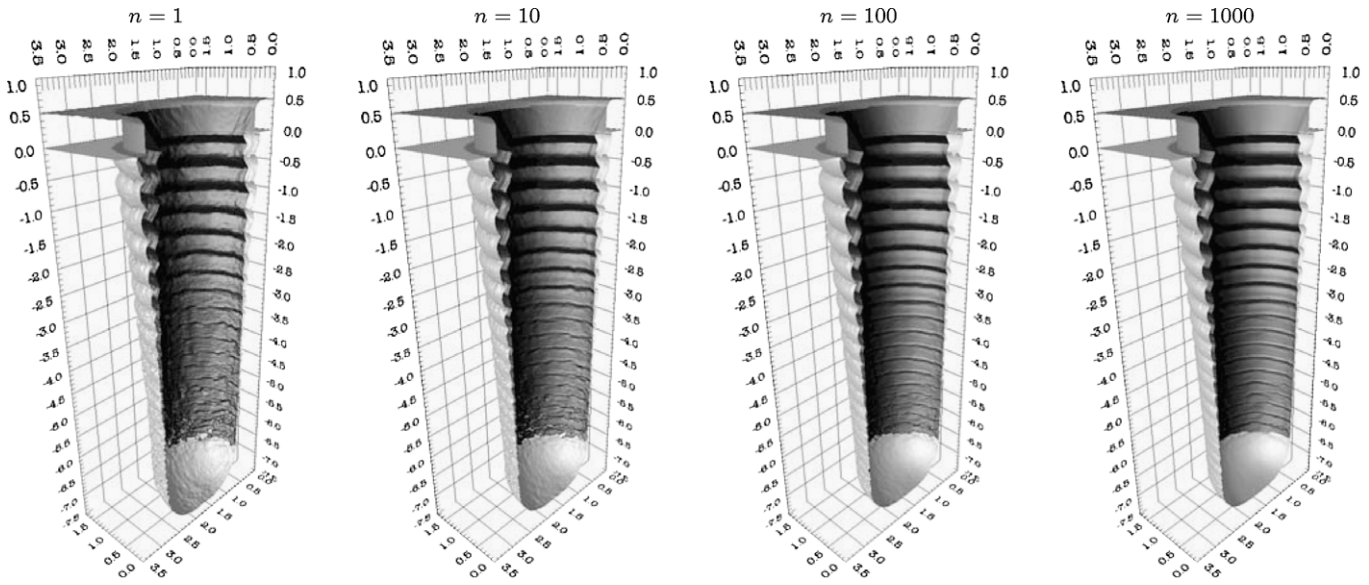

Fig. 11. The final profiles for different n. Apart from the roughness of the surface the results are very similar, although n and hence the computation time for ray tracing varies over 3 orders of magnitude.

Here p denotes the speedup due to parallelization as given in Fig. 7. Fig. 12 shows plots of T and the number of time steps N_{steps}, which are necessary for the whole simulation. The times are referred to 4.8h, which is the runtime for $n = 1$ on a single CPU. For small n the number of time steps increases due to the poorer statistics. The expected maximum surface velocity of all grid points is larger for smaller n due to the larger variation of the velocity distribution. Hence, according to the CFL-criterion the maximum allowed time step must be smaller, leading to more time steps. This on the other hand side implies that the total number of simulated particles during the whole calculation increases, which improves the accuracy again. Consequently, a minimum exists for the total runtime as can be seen in Fig. 12. Choosing a value for n smaller than that of the minimum is not favorable.

Another way to speed up the calculation is the usage of coarser spatial discretizations. In Fig. 13 the simulation results are compared for $n = 100$ and grid spacings of 70 nm and 14 nm, which correspond to grids with lateral extensions 50×25 and 250×125, respectively. The runtime between both calculations differs by a factor more than $5^3 = 125$. This comes from the fact that the number of surface discretization elements N scales inversely quadratically with the grid spacing. Furthermore, due to the CFL-condition, the number of required time steps scales inversely with the grid spacing.

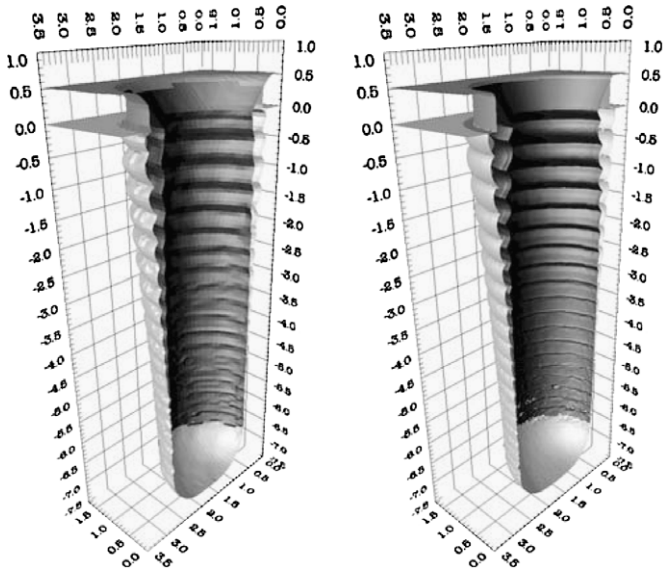

Fig. 13. The final results of calculations on grids with grid spacing 70 nm (left) and 14 nm (right).

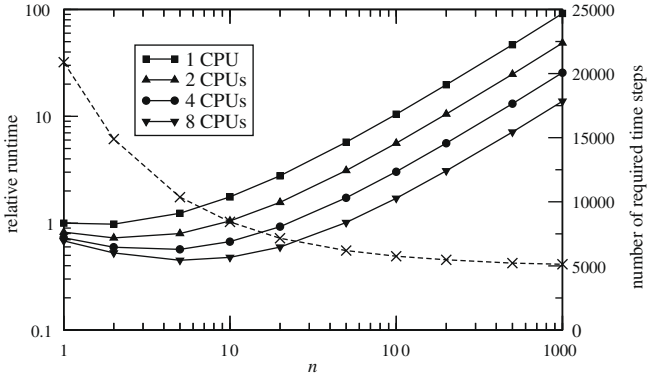

Fig. 12. The relative runtimes for different accuracies n and different numbers of used CPUs. In addition the number of required time steps is also given (dashed).

Despite the large difference in computation time the results look again very similar. Due to the sub-time step resolution of the material dependent etch rates within our multi-level-set framework [20] the error introduced by the coarse grid is kept small. The final depths differ only by 2.1%. However, the coarse grid is not able to represent the rippled sidewalls accurately.

7. Conclusion

We applied modern techniques like the sparse field level set method and the HRLE data structure for the profile evolution as well as ray tracing algorithms for the flux calculation to three-dimensional simulations of the Bosch process. The presented

multi-level-set approach allows an accurate, robust, and memory efficient handling of different material regions, including thin layers.

Due to its simple parallelization and due to the increasing number of processor cores, ray tracing becomes an alternative to common direct integration methods for particle transport calculation. Although the Monte Carlo method was only demonstrated with a relative simple model, it is capable to solve more complex ones, where for example energy dependent sputter rates or specular reflexions of ions are incorporated. In contrast to direct integration methods such effects can be straightforwardly implemented without increasing the algorithmic complexity. A three-dimensional simulation of a more complex reactive ion etching model was already demonstrated in [34].

At the expense of accuracy the whole calculation can be drastically accelerated by reducing the number of simulated particles or the grid resolution. The results still reflect the qualitative characteristics. Therefore, the Monte Carlo method is useful for the fast examination of parameter variations. After finding the optimized set of parameters a final simulation can be carried out to get an appropriately accurate profile.

For future work it may be interesting to incorporate mask charging effects [35], where the emerging electric field leads to non-linear trajectories of ions, which complicates ray tracing.

References

[1] F. Lärmer, A. Schilp, Patent No. DE4241045 (Germany, issued 5 December 1992), US5,501,893 (U.S., issued 26 March 1996).
[2] R. Zhou, H. Zhang, Y. Hao, D. Zhang, Y. Wang, Simulation of profile evolution in etching-polymerization alternation in DRIE of silicon with SF_6/C_4F_8, in: Proceedings of the 16th IEEE International Micro Electro Mechanical Systems Conference (MEMS), 2003, pp. 161–164.
[3] R. Zhou, H. Zhang, Y. Hao, Y. Wang, J. Micromech. Microeng. 14 (7) (2004) 851–858.
[4] Y. Tan, R. Zhou, H. Zhang, G. Lu, Z. Li, J. Micromech. Microeng. 16 (12) (2006) 2570–2575.
[5] G. Kokkoris, A. Tserepi, A.G. Boudouvis, E. Gogolides, J. Vac. Sci. Technol. A 22 (4) (2004) 1896–1902.
[6] A. Shumilov, I. Amirov, Russ. Microelectron. 36 (4) (2007) 241–250.
[7] G. Sun, X. Zhao, H. Zhang, L. Wang, G. Lu, 3-d simulation of Bosch process with voxel-based method, in: Proceedings of the second IEEE International Conference on Nano/Micro Engineered and Molecular Systems (IEEE-NEMS), 2007, pp. 45–49.
[8] A. Hössinger, Z. Djurić, A. Babayan, Modeling of deep reactive ion etching in a three-dimensional simulation environment, in: Proceedings of the International Conference on Simulation of Semiconductor Processes and Devices (SISPAD), 2007, pp. 53–56.
[9] R.A. Gottscho, C.W. Jurgensen, D.J. Vitkavage, J. Vac. Sci. Technol. B 10 (5) (1992) 2133–2147.
[10] T.S. Cale, G.B. Raupp, J. Vac. Sci. Technol. B 8 (6) (1990) 1242–1248.
[11] T.S. Cale, T.P. Merchant, L.J. Borucki, A.H. Labun, Thin Solid Films 365 (2) (2000) 152–175.
[12] J.A. Sethian, Level Set Methods and Fast Marching Methods, Cambridge University Press, 1999.
[13] G. Kokkoris, A.G. Boudouvis, E. Gogolides, J. Vac. Sci. Technol. A 24 (6) (2006) 2008–2020.
[14] R.A. Gottscho, J. Vac. Sci. Technol. B 11 (5) (1993) 1884–1889.
[15] D. Adalsteinsson, J.A. Sethian, J. Comput. Phys. 148 (1) (1999) 2–22.
[16] R. Malladi, J. Sethian, B. Vemuri, IEEE T. Pattern. Anal. 17 (2) (1995) 158–175.
[17] R.T. Whitaker, Int. J. Comput. Vision 29 (3) (1998) 203–231.
[18] B. Houston, M.B. Nielsen, C. Batty, O. Nilsson, K. Museth, ACM Trans. Graph. 25 (1) (2006) 151–175.
[19] A. Pasko, V. Adzhiev, A. Sourin, V. Savchenko, The Visual Comput. 11 (8) (1995) 429–446.
[20] O. Ertl, S. Selberherr, Comput. Phys. Commun. (2009), doi:10.1016/j.cpc.2009.02.002.
[21] H. Liao, T.S. Cale, Thin Solid Films 236 (1–2) (1993) 352–358.
[22] D. Adalsteinsson, J.A. Sethian, J. Comput. Phys. 138 (1) (1997) 193–223.
[23] P.L. O'Sullivan, F.H. Baumann, G.H. Gilmer, J. Appl. Phys. 88 (7) (2000) 4061–4068.
[24] W.E. Lorensen, H.E. Cline, SIGGRAPH Comput. Graph. 21 (4) (1987) 163–169.
[25] C. Heitzinger, A. Sheikholeslami, F. Badrieh, H. Puchner, S. Selberherr, IEEE Trans. Electron. Dev. 51 (7) (2004) 1129–1134.
[26] J. Arvo, D. Kirk, A survey of ray tracing acceleration techniques, in: An Introduction to Ray Tracing, Academic Press Ltd., 1989, pp. 201–262.
[27] V. Havran, Heuristic ray shooting algorithms, Ph.D. thesis, Department of Computer Science and Engineering, Faculty of Electrical Engineering, Czech Technical University in Prague (November 2000).
[28] I. Wald, V. Havran, On building fast kd-trees for ray tracing, and on doing that in O(N log N), in: Proceeding of the IEEE Symposium on Interactive Ray Tracing, 2006, pp. 61–69.
[29] G. Marmitt, A. Kleer, I. Wald, H. Friedrich, P. Slusallek, Fast and accurate ray-voxel intersection techniques for iso-surface ray tracing, in: Proceedings of the 9th International Fall Workshop Vision, Modeling, and Visualization (VMV), 2004, pp. 429–435.
[30] T. Smy, S.K. Dew, R.V. Joshi, J. Vac. Sci. Technol. A 19 (1) (2001) 251–261.
[31] O. Ertl, C. Heitzinger, S. Selberherr, Efficient coupling of Monte Carlo and level set methods for topography simulation, in: Proceedings of the International Conference on Simulation of Semiconductor Processes and Devices (SISPAD), 2007, pp. 417–420.
[32] OpenMP C and C++ application program interface. URL <http://www.openmp.org>.
[33] M. Mascagni, A. Srinivasan, ACM Trans. Math. Softw. 26 (3) (2000) 436–461.
[34] O. Ertl, S. Selberherr, Three-dimensional topography simulation using advanced level set and ray tracing methods, in: Proceedings of the International Conference on Simulation of Semiconductor Processes and Devices (SISPAD), 2008, pp. 325–328.
[35] K.P. Giapis, G.S. Hwang, O. Joubert, Microelectron. Eng. 61–62 (2002) 835–847.