# Implementation and evaluation of the Level Set method: Towards efficient and accurate simulation of wet etching for microengineering applications

C. Montoliu[a], N. Ferrando[b], M. A. Gosálvez[c], J. Cerdá[a], R. J. Colom[a]

[a]*Instituto de Instrumentación para Imagen Molecular (I3M) centro mixto CSIC – Universitat Politècnica de València – CIEMAT, Camino de Vera s/n, 46022 Valencia, Spain*

[b]*Centro de Física de Materiales, centro mixto CSIC – UPV/EHU and Donostia International Physics Center (DIPC), 20018 Donostia-San Sebastian, Spain*

[c]*Department of Material Physics, University of the Basque Country UPV/EHU and Donostia International Physics Center (DIPC), 20018 Donostia-San Sebastian, Spain*

## Abstract

The use of atomistic methods, such as the Continuous Cellular Automaton (CCA), is currently regarded as a computationally efficient and experimentally accurate approach for the simulation of anisotropic etching of various substrates in the manufacture of Micro-electro-mechanical Systems (MEMS). However, when the features of the chemical process are modified, a time-consuming calibration process needs to be used to transform the new macroscopic etch rates into a corresponding set of atomistic rates. Furthermore, changing the substrate requires a labor-intensive effort to reclassify most atomistic neighborhoods. In this context, the Level Set (LS) method provides an alternative approach where the macroscopic forces affecting the front evolution are directly applied at the discrete level, thus avoiding the need for reclassification and/or calibration. Correspondingly, we present a fully-operational Sparse Field Method (SFM) implementation of the LS approach, discussing in detail the algorithm and providing a thorough characterization of the computational cost and simulation accuracy, including a comparison to the performance by the most recent CCA model. We conclude that the SFM implementation achieves similar accuracy as the CCA method with less fluctuations in the etch front and requiring roughly 4 times less memory. Although SFM can be up to 2 times slower than CCA for the simulation of anisotropic etchants, it can also be up to 10 times faster than CCA for isotropic etchants. In addition, we present a parallel, GPU-based implementation (gSFM) and compare it to an optimized, multi-core CPU version (cSFM), demonstrating that the SFM algorithm can be successfully parallelized and the simulation times consequently reduced, while keeping the accuracy of the simulations. Although modern multicore CPUs provide an acceptable option, the massively parallel architecture of modern GPUs is more suitable, as reflected by computational times for gSFM up to 7.4 times faster than for cSFM.

*Keywords:* Level Set Method, Sparse Field Method, Anisotropic wet chemical etching, Microengineering, Cellular Automata, Micro-electro-mechanical systems, MEMS, KOH, TMAH, Parallel computing, GPU, CUDA

## 1. Introduction

Anisotropic wet chemical etching is one of the most popular bulk micromachining methods for the fabrication of Micro-Electro-Mechanical Systems (MEMS). This process is technologically relevant due to the low cost and ability to generate smooth/flat surfaces and release suspended structures. Nevertheless, the resulting three-dimensional shapes are difficult to predict due to multiple dependencies, such as the crystal orientation of the surface [1, 2, 3, 4, 5, 6], the composition of the applied etchant –*e.g.* potassium hydroxide (KOH) [7, 8], tetramethylammonium hydroxide (TMAH) [9, 10] or various other solutions [11]– the etchant concentration [4, 6], the temperature [2, 4, 6] or the use of additives, such as Triton X-100 [12, 13] or isopropyl alcohol [4, 8, 14]. Accordingly, an important effort has been dedicated through the years to model accurately the process.

The first simulators were based on geometrical models focused on describing the etching front as a set of moving facets [15, 16]. This posed various challenges, such as the efficient implementation of the intersection of two or more planes in 3D and, correspondingly, the description of wafer perforation. On the other hand, the use of atomistic approaches, such as Cellular Automata (CA) [17, 18, 19, 20, 21, 22, 23, 24] and Kinetic Monte Carlo (KMC) [25, 26, 27, 28], became gradually successful. By treating the evolving surface as a collection of atomistic sites within a lattice, the surface atoms are progressively removed according to simple microscopic rules, thus resulting in the emergence of the neighbor sites into the surface. This enables a natural description of topological changes in the front (*e.g.* coalescing/splitting of contiguous regions) without any further computational/programming effort. While KMC has been traditionally used to model the surface roughness on different substrate orientations [26, 27], there exist successful examples of its use for microstructure prediction [29, 28]. In turn, although CA is best suited for modeling the formation of 3D microstructures, it has also been used to describe a wide range of surface morphologies [30].

The first CA models were based on stochastic approaches

*Email address:* carmonal@upv.es (C. Montoliu)

[17], eventually giving way to various deterministic procedures [18, 19, 20]. By taking into account several physical phenomena, such as the *step flow* nature of wet etching [5, 22], the Continuous CA (CCA) method has demonstrated to be the most accurate, simulating reliably wet etching for a wide range of MEMS structures and conditions [21, 31]. In addition, algorithmic accelerations for sequential and parallel computational environments have been presented [23, 24] and commercial simulators, such as IntelliEtch [32], are able to execute the CCA calculations on Graphics Processing Units (GPUs), drastically reducing the computation times to just a few seconds per simulation. Another advantage over the classical geometrical methods is the possibility of calibrating the atomistic model parameters by using a reduced dataset of experimental etch rates [33]. Originally requiring careful supervision by the user, the calibration process has been recently automated by using an evolutionary algorithm for both KMC [28] and CCA [6, 34]. Despite the good results, new calibrations are needed each time the experimental conditions are changed, requiring several hours or even days for calibration completion. In this context, the search of an alternative method, which is simultaneously capable of using the experimental data without prior calibration while remaining computationally-efficient, has the potential to influence significantly the future of MEMS design.

In this study we consider the use of the Level Set (LS) method, introduced in the 80's for capturing moving fronts [35] and developed further for related problems, such as image reconstruction [36], image segmentation [37], moving liquid interfaces [38, 39] as well as chemical etching [40, 41, 42, 43] and Deep Reactive Ion Etching (DRIE)/plasma etching [44, 45]. While LS has been previously used to simulate anisotropic etching [46, 47, 48, 49], the computational efficiency and relative accuracy in comparison to the existing methods have not been addressed before. In addition, there is some freedom to perform the LS calculations over the whole domain or just in the vicinity of the evolving front [50]. Thus, the computational efficiency and relative accuracy of different approaches need to be compared to the benchmark results by the CCA method.

Based on the Sparse Field Method (SFM) [50], which concentrates the processing effort on the active points around the evolving front, we present an efficient sequential implementation of the LS method and compare it to an equivalent sequential implementation of the CCA method. This way, we evaluate the viability of LS as a simulation tool for MEMS design in practice. In addition, two parallel versions of the SFM are developed: (i) cSFM, which takes advantage of the multicore nature of modern Central Processing Units (CPUs) and the Single Instruction Multiple Data (SIMD) functionality, enabling the computation of the same instruction over multiple data in a single clock cycle; and (ii) gSFM, which benefits from the massively parallel architecture of modern Graphics Processing Units (GPUs), enabling the simultaneous processing of many thousands of threads and, thus, reducing drastically the computational time. Both cSFM and gSFM implementations are compared in terms of their computational efficiency.

The rest of the paper is organized as follows. The LS method as well as the numerical techniques needed to adapt it to anisotropic etching are presented in Section 2, including the resulting algorithm in detail. This is followed by the results from several examples in Section 3, demonstrating the reliability of our LS simulations by direct comparison to experimental microphotographies. This also includes a comparison of the computational time, memory usage and relative accuracy between LS and CCA. Finally, the parallel cSFM and gSFM implementations are presented and compared in Section 4.

## 2. Level Set Method applied to anisotropic etching

The LS method was introduced by Osher and Sethian in order to track moving fronts [35]. The main idea consists in embedding the front, $x(t)$, inside a signed distance function, $\phi$, such that the distance right at the front at any time is zero by construction: $\phi(x(t), t) = 0$. This enables to apply to the function $\phi$ itself the partial derivatives in time and space required to describe the propagation of the front. In the specific case where the front evolves in the normal direction, using the chain rule the propagation equation (or *LS equation*) is:

$$\phi_t + R(x(t), t)\,|\nabla\phi| = 0\ , \tag{1}$$

where $R(x(t), t)$ is defined over the whole domain and denotes the velocity of a point in the normal direction, as determined by the physical laws of the process being simulated. The LS equation can also be written in Hamilton-Jacobi form:

$$\phi_t + H(\nabla\phi(x(t), t)) = 0\ , \tag{2}$$

where $H = R(x(t), t)\,|\nabla\phi|$ is the Hamiltonian. For anisotropic etching, where the rate of propagation is a function of the local normal vector, the Hamiltonian depends on the geometry of the front (*non-convex* Hamiltonian), in which case there are several techniques to discretize Eq. 2, such as the Lax-Friedrichs (LF) [51], Roe-Fix [52] or Godunov [53] schemes. Based on central difference derivatives, LF is the simplest scheme that preserves monotonicity. Applied to Eq. 2, one obtains the following space-time discretization:

$$\phi^{n+1}_{i,j,k} = \phi^{n}_{i,j,k} - \Delta t\Big[H\left(\beta_x^+, \beta_y^+, \beta_z^+\right) - \alpha_F\left(\alpha_x\beta_x^- + \alpha_y\beta_y^- + \alpha_z\beta_z^-\right)\Big], \tag{3}$$

$$\beta_l^{\pm} = \frac{1}{2}\left[\left(a^{+l}_{i,j,k}\right)^n \pm \left(a^{-l}_{i,j,k}\right)^n\right], \tag{4}$$

$$\left(a^{\pm l}_{i,j,k}\right)^n = \frac{\pm\phi^n_{(i,j,k)\pm1} \mp \phi^n_{(i,j,k)}}{\Delta l}, \tag{5}$$

$$\alpha_l = max\left|\frac{\partial H}{\partial \phi_l}\right|, \tag{6}$$

where $l = x, y, z$ and the operation $(i, j, k) \pm 1$ only affects the index $i$, $j$ or $k$ corresponding to the selected dimension $l$. Although $\Delta l$ is the spatial resolution of the grid in a particular dimension, from now on the grid resolution will be considered to be the same in all dimensions and referred to as $\Delta x$. Finally, the $\alpha_l$ coefficients are the artificial viscosity factors inherent to the LF method and $\alpha_F$ is an overall viscosity coefficient which we introduce to guarantee the stability of the front by modifying simultaneously the three $\alpha_l$ values. Larger $\alpha_F$ results in smoother surfaces while smaller $\alpha_F$ generates sharp features/discontinuities, which should be handled carefully to avoid unrealistic and/or diverging solutions. We have found that $\alpha_F = 0.48$ allows the front to converge with every tested etchant.

In order to apply the LF scheme, it is necessary to express the viscosity factors of Eq. 6 in terms of $R(x(t), t)$. In practice, the experimental etch rates are constant in time, depending only on the local crystallographic orientation of the surface. Correspondingly, the etch rates are organized as a 180x45 matrix, $R(\theta, \Phi)$, as a function of the azimuth ($\Phi$) and latitude ($\theta$) of 8100 normal vectors in spherical coordinates with a resolution of 2 degrees [6]. Thus, applying the chain rule to Eq. 6 we obtain:

$$\alpha_l = max\left|\frac{\partial R(\theta,\Phi)}{\partial N_l}\frac{\phi_p^2+\phi_q^2}{|\nabla\phi|^2} + R(\theta,\Phi)\cdot N_l\right| \tag{7}$$

where $N_l$ is the $l$-component of the unit normal vector with $l$, $p$ and $q$ any of $x$, $y$ and $z$, such that $p \neq l$, $q \neq l$ and $p \neq q$. Here, $\frac{\partial R(\theta,\Phi)}{\partial N_l}$ denotes the variation of the etch rate with respect to the $l$-component of the normal vector, which is numerically determined by discretizing each component with a resolution of $\Delta N_l = 0.01$:

$$\frac{\partial R(\theta,\Phi)}{\partial N_l} = \frac{R(N_l+\Delta N_l,N_p,N_q)-R(N_l-\Delta N_l,N_p,N_q)}{2\cdot\Delta N_l} \tag{8}$$

It is important to notice that the cost to evaluate Eq. 3 has four contributions: (i) the computation of the spatial derivatives according to Eqs. 4 and 5, which involves the processing of neighbor $\phi$ values, (ii) the update of the Hamiltonian, which requires determining the value of the etch rate corresponding to the normal vector, (iii) the determination of the artificial viscosities according to Eq. 7, which requires searching for the maximum value over the whole domain, and (iv) the update of the new $\phi$ values. As described below, contribution (iii) represents the largest computational burden for the implementation of the SFM method, taking about 55% of the total time required to evaluate Eq. 3, while the other contributions represent: (i) 20-28% (ii) 11-13% and (iv) 7-12% .

### 2.1. *Sparse Field Method*

Due to the need to update the value of $\phi$ at every point of the grid, the LS method has a computational cost of $O(N^3)$, where $N$ is the number of grid points in one dimension. In order to reduce the cost Adalsteinsson *et al.* introduced the Narrow Band (NB) method [54], which updates only the points located within $k$ layers counted from the front itself, thus reducing the computational cost to $O(kN^2)$. However, this approach is not completely optimal because more points are still updated than strictly necessary and, in addition, the signed distance function needs to be rebuilt periodically every time the front reaches the $k$-th layer. The Sparse Field Method (SFM) was introduced by Whitaker [50] in order to reduce the active region to only the strictly necessary points while efficiently updating the signed distance in each time step [55].

Based on the use of backward and forward first order derivatives, the SFM focuses on maintaining only 3 groups of points: those at the front ($L0$), those at distance $\Delta x$ from the front ($Lp1$), and those at distance $-\Delta x$ ($Ln1$). The rest of the points are not included in any list since they do not need to be updated. The active points are labelled as 1, 0 or -1, depending on whether they belong to $Lp1$, $L0$ or $Ln1$, respectively, while the rest of the points are labelled as 2 or $-2$, depending on whether they are located below or above the front. If higher $k$-order derivatives are used (*e.g.* to increase the accuracy), the points at distance $\leq |\pm(k)\Delta x|$ from the front should be maintained in similar lists. Fig. 1 shows a two-dimensional example of a wet etching interface and the necessary lists. For each time step, only the values of $\phi$ for the points in $L0$ are updated using Eq. 3. For the points in $Lp1$ ($Ln1$), we search for the nearest point belonging to $L0$ and add (subtract) $\Delta x$ to the found distance value. After propagating the front, the neighboring points that are not included in any list are added to the $Lp1$ or $Ln1$ list, depending on the sign of their distance value. This is described in detail in Section 2.2.

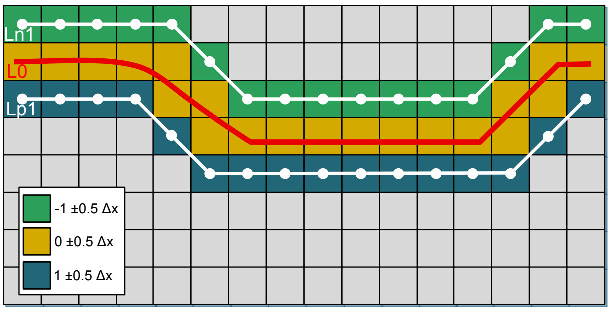


Figure 1: Two-dimensional example of an interface for the simulation of wet etching. Green, orange and blue voxels represent the $Ln1$, $L0$ and $Lp1$ lists used in SFM, respectively, while grey voxels do not need to be updated. Positive/negative distances are associated to the points located below/above the front. Signed distance ranges are shown.

### 2.2. *Pseudocode: SFM implementation*

The pseudocode of our SFM implementation is as follows:

1. Load the mask pattern and apply it to the substrate. This simulates the lithographic process used to mask the wafer.
2. Choose the substrate type (silicon or quartz), crystallographic orientation, grid dimensions, etchant concentration, temperature and the etching time ($t_{\max}$, in minutes). The depth of the grid is calculated to ensure that the front does not reach the bottom of the grid during $t_{\max}$:

   $$Num_z = 4 + t_{\max}\cdot R(\theta = 0, \Phi = 0)\ , \tag{9}$$

   where $R(\theta = 0, \Phi = 0)$ is the vertical etch rate. To reduce memory, dynamic allocation and an octree data structure can be used [23].
3. The three-dimensional grid is built and the exposed surface is defined.
4. Calculate the time step $\Delta t$ according to the Courant-Friedrichs-Levy (CFL) condition [56]. To avoid front divergence we use: $\Delta t = \frac{0.3\Delta x}{max\{R(\theta,\Phi)\}}$.

5. Calculate the signed distance $\phi$ for the points located near the front and assign a constant value to all other points.
6. Add the points to the lists $L0$, $Lp1$ or $Ln1$, depending on the distance value.
7. Main loop:
    1) For every point in $L0$, determine:
        - i. The forward and backward derivatives using Eq. 5.
        - ii. The normal components $N_l = \phi_l / |\nabla\phi|$, where $l = x, y$ and $z$.
        - iii. The etch rate (convert the cartesian components to spherical coordinates and access the corresponding matrix element).
        - iv. The maximum value according to Eq. 7.
    2) Define temporary lists $S0$, $Sp1$ and $Sn1$.
    3) For every point $(i, j, k)$ in $L0$:
        - Update $\phi$ using Eq. 3.
        - If $\phi^{n+1}_{i,j,k} < -0.5\Delta x$: remove $(i, j, k)$ from $L0$; add it to $Sn1$.
        - If $\phi^{n+1}_{i,j,k} > 0.5\Delta x$: remove $(i, j, k)$ from $L0$; add it to $Sp1$.
    4) For every point $(i, j, k)$ in $Lp1$:
        - i. Find the neighbor $(i_n, j_n, k_n)$ from $L0$ that has the smallest distance value $\phi^{n+1}_{i_n,j_n,k_n}$. Then update: $\phi^{n+1}_{i,j,k} = \phi^{n+1}_{i_n,j_n,k_n} + \Delta x$.
        - ii. If $|\phi^{n+1}_{i,j,k}| \leq 0.5\Delta x$: remove $(i, j, k)$ from $Lp1$; add it to $S0$.
        - iii. If no $L0$ neighbor is found, remove $(i, j, k)$ from $Lp1$.
    5) For every point $(i, j, k)$ in $Ln1$:
        - i. Find the neighbor $(i_n, j_n, k_n)$ from $L0$ that has the maximum distance value $\phi^{n+1}_{i_n,j_n,k_n}$. Then update: $\phi^{n+1}_{i,j,k} = \phi^{n+1}_{i_n,j_n,k_n} - \Delta x$.
        - ii. If $|\phi^{n+1}_{i,j,k}| \leq 0.5\Delta x$: remove $(i, j, k)$ from $Ln1$; add it to $S0$.
        - iii. If no $L0$ neighbor is found, remove $(i, j, k)$ from $Ln1$.
    6) For every point in $Sp1$:
        - i. Add it to $Lp1$.
        - ii. Change the state to 1.
        - iii. Remove it from $Sp1$.
    7) For every point in $Sn1$:
        - i. Add it to $Ln1$.
        - ii. Change the state to $-1$.
        - iii. Remove it from $Sn1$.
    8) For every point in $S0$:
        - i. Add it to $L0$.
        - ii. Change the state to 0.
        - iii. Remove it from $S0$.
    9) For every point $(i, j, k)$ in $L0$:
        - i. Search for neighbor points $(i_{np}, j_{np}, k_{np})$ with *state* = 2 and:
            - Change the state of $(i_{np}, j_{np}, k_{np})$ to 1.
            - Update: $\phi^{n+1}_{i_{np},j_{np},k_{np}} = \phi^{n+1}_{i,j,k} + \Delta x$.
            - Add $(i_{np}, j_{np}, k_{np})$ to $Lp1$.
        - ii. Search for neighbor points $(i_{np}, j_{np}, k_{np})$ with *state* = $-2$ and:
            - Change the state of $(i_{np}, j_{np}, k_{np})$ to $-1$.
            - Update: $\phi^{n+1}_{i_{np},j_{np},k_{np}} = \phi^{n+1}_{i,j,k} - \Delta x$.
            - Add $(i_{np}, j_{np}, k_{np})$ to $Ln1$.
    10) Update the time: $t^{n+1} = t^n + \Delta t$. If $t^{n+1} < t_{\max}$, go to step 7.
8. Extract the final surface from $\phi$ (zero level set) and visualize it. We use the marching cubes method [57] to solve this step.

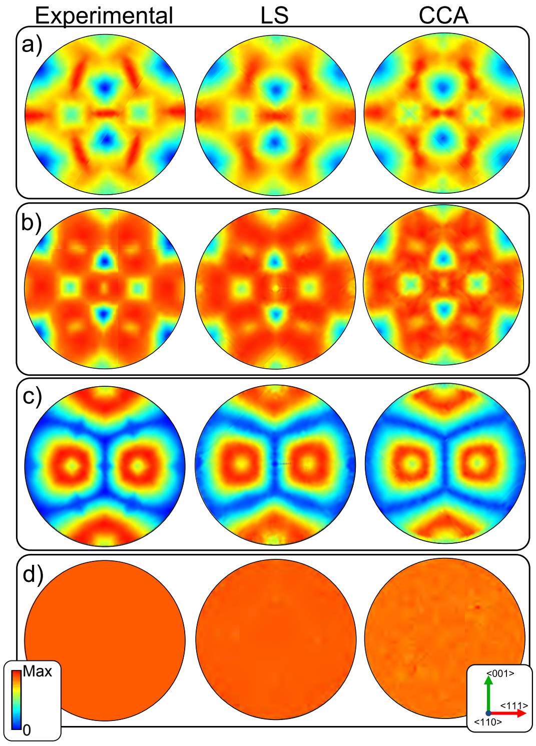


Figure 2: Comparison between experimental and simulated etch rate distributions using the SFM and CCA methods: (a) KOH 40 wt% at 70°C, (b) TMAH 25 wt% at 80°C, (c) TMAH 25 wt% + Triton 0.1 v/v at 80°C and (d) an isotropic etchant.

## 3. Results

This section presents an extensive collection of results. For the LS approach, the SFM implementation is used. For comparison, we use the Constant-Time-Stepping (CTS) implementation of the Continuous Cellular Automaton (CCA) presented in [23] in order to simulate the same experiments. The parameters used in the LS simulations are as follows: $Num_z$ = Eq. 9, $\Delta t = \frac{0.3\Delta x}{max\{R(\theta,\Phi)\}}$, $\alpha_F = 0.48$ and $\Delta N_l = 0.01$. Both SFM and CCA implementations are written using sequential Java, similar data structures and single precision, since it has been demonstrated that this level of precision is sufficient to produce accurate results [24]. The testing machine consists on an Intel Core i7 at 2.8 GHz with 4 GB of RAM using 64 bit Windows-based

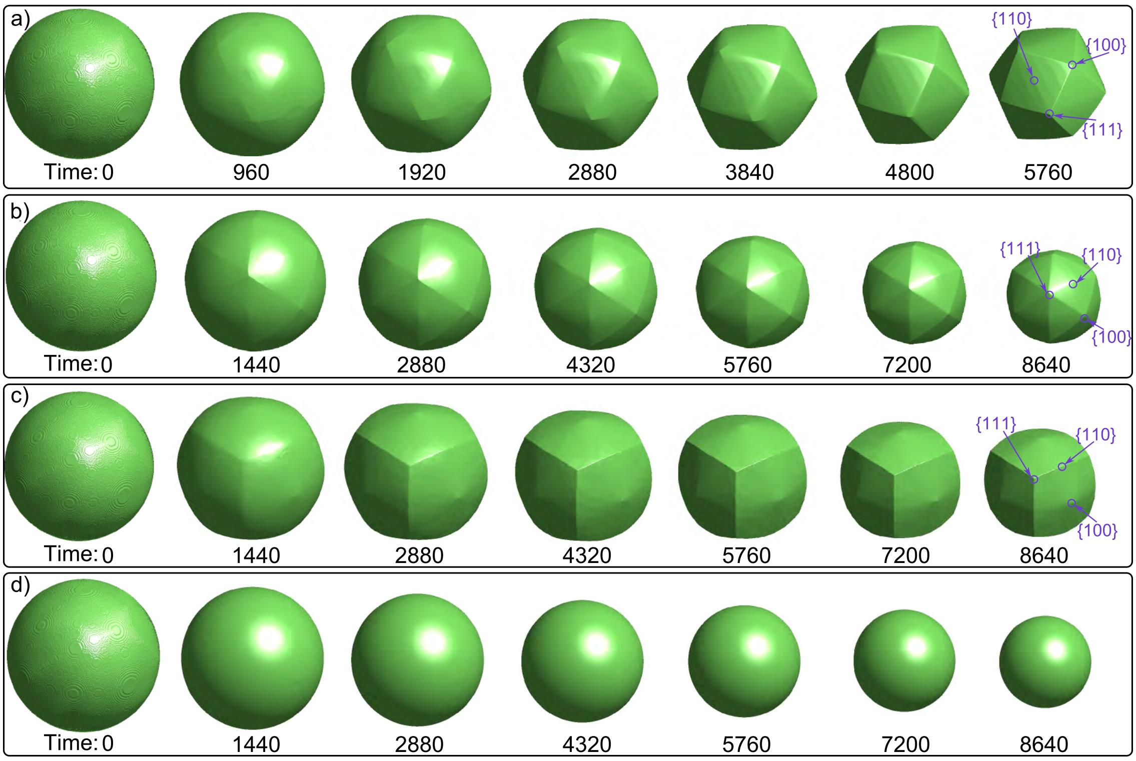


Figure 3: SFM-simulated evolution of a typical sphere of 22 mm of radius etched in: a) KOH 40 wt% at 70°C, b) TMAH 25 wt% at 80°C, c) TMAH 25 wt% + Triton 0.1 v/v at 80°C and d) an isotropic etchant. The numbers represent the etch time (in minutes).

server Java Virtual Machine (version 1.7.0_03). The purpose of this section is to compare the computational cost, memory use and accuracy of similar implementations of both methods.

### *3.1. Orientation-dependence of the etch rate*

From an experimental point of view the traditional way to represent the anisotropy of an etchant is to show an stereographic projection of the unit sphere while presenting the etch rates at different locations ($\theta$, $\Phi$) using isolines and/or a collection of colors. Since a sphere contains all possible substrate orientations, it provides a complete representation of the etchant anisotropy in a compact way. The procedure originated in early experimental reports, where real hemispherical samples were actually etched [7, 10]. By mechanically probing the hemisphere surface before and after etching at different latitude/longitude locations, the etched distance is determined as a function of the orientation ($\theta$, $\Phi$), thus obtaining the etch rate by dividing by the etch time. Since the procedure can be easily mimicked computationally, in this section we compare the simulated etch rate distributions to those obtained in an extensive, experimental study by Gosálvez *et al.* [6], where ∼ 8100 etch rates were determined for each of a total of 33 different etchants.

Fig. 2 compares the experimental and simulated stereographic projections for four different etching conditions, including a) KOH 40 wt% at 70°C, b) TMAH 25 wt% at 80°C, c) TMAH 25 wt% + Triton 0.1 v/v at 80°C and d) an isotropic etchant. There is much similarity between the experimental and SFM-simulated etch rate distributions, confirming the correctness of our implementation. In comparison, the results from the atomistic CCA method seem more noisy in all the cases, specially for the isotropic etchant. For very long etch times, Fig. 3 shows the actual shape evolution simulated with the SFM for typical spheres etched in the same etchants. As expected, the lower the local etch rate the sharper the corresponding protruding region, which reflects the underlying symmetry of the etch rate: two-, four- and six/three-fold around the {110}, {100} and {111} orientations, respectively.

### *3.2. Grid resolution*

The resolution of the grid used to represent the substrate affects the final accuracy of the simulated microstructures. Typically, the larger the number of grid points the better the accuracy, a phenomenon that is also observed in the atomistic models. As an example, Fig. 4 shows how the number of grid points affects a particular length measurement in both the SFM and CCA simulated fronts for the same etching process: the underetching of the convex corners of a square-shaped mask pattern on a 128x128 $\mu m^2$ substrate etched in KOH 40 wt% at 70°C for 30 minutes. As shown in Fig. 5, the measured width of the obtained mesa structure decreases as the number of grid points is increased (voxels in SFM and Unit Cells (UCs) in CCA, each cell containing 4 silicon atoms), following a negative exponential behavior that essentially saturates at the 512*x*512 grid size. Considering a discrepancy of 2% with respect to this value,

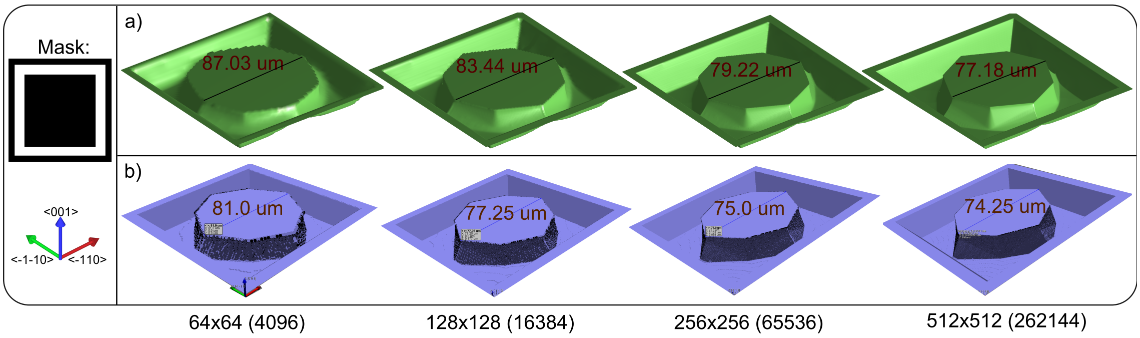


Figure 4: Dependence of simulated features on the resolution of the grid: (a) SFM and (b) CCA. (Etching in KOH 40 wt% at 70°C for 30 minutes. Substrate size: 128x128 $\mu m^2$.)

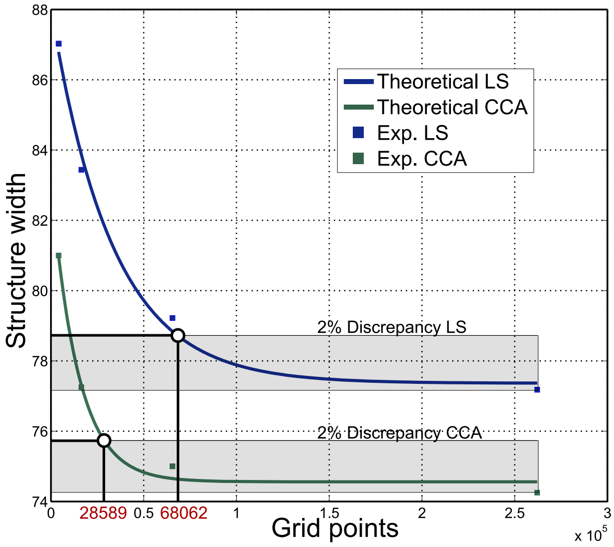


Figure 5: Width of the obtained mesa structures shown in Fig. 4 as a function of the grid size. Exponential decay fits are shown.

the fitted exponentials indicate that the minimal grid resolution to achieve this error is 68062 voxels and 28589 UCs for SFM and CCA, respectively. Thus, the SFM method requires about double the number of points to reach similar accuracy as CCA. Nevertheless, the discrepancy between SFM and CCA for 512$x$512 grid points is only 3.8%, thus concluding that the SFM method seems suitable for the simulation of anisotropic etching of real structures.

### 3.3. Etching of real structures

We now compare the relative performance of the SFM and CCA methods in terms of computational time, memory use and relative error with respect to the experiments. For this purpose, we consider several microengineering structures obtained by the etching of patterned Si{100} wafers in KOH 30 wt% at 80°C [58]. The SFM/CCA-simulated etchant is actually KOH 35 wt% at 80°C, which has a very similar etch rate distribution [6]. Fig. 6 compares all the etched shapes. From left to right, the different columns correspond to (i) the used mask patterns, (ii) the experimental results [58], (iii) the SFM-simulated fronts and (iv) the CCA-based results. The grid sizes, chosen for each simulation to satisfy the 2% error described in the context of Fig. 5, are collected in Table 1, which also presents the etched depth, memory use and computational time for both computational methods. According to the figure, the simulated microstructures by both SFM and CCA are very similar to the experimental shapes. This is confirmed by the values of the simulated etched depths, which are in close proximity to the experimental values. These results demonstrate the reliability of the SFM simulator, achieving similar accuracy and producing less noise than the CCA model.

As shown in Table 1, the required computational times for both methods are within the same order of magnitude for all tests, although CCA can be up to 2 times faster. The larger computational cost for SFM is assigned primarily to the need to determine the maximum in Eq. 7 over all active points, which represents 36-42% of the simulation time. Comparatively, the computation of the spatial derivatives by Eqs. 4 and 5 takes 14-22% of the simulation time. This involves on average more operations than the processing of the atomistic neighborhood in the CCA. Finally, the task of updating the content of the lists (*i.e.*, steps 4) through 9) of the SFM algorithm) is also relevant, representing 13-26% of the whole time. In turn, the reported *main* memory refers to the main variables stored during the whole simulation, which directly reflects the grid size. Additionally, the memory referred to as *lists* corresponds to the points stored in the SFM lists and similar structures for the CCA model. The lists use much less memory than the grid since only the necessary information to access to the main grid is stored. Although the SFM implementation typically requires a finer grid, Table 1 shows that the CCA method needs to store more atoms, requiring between 3.7 and 4.8 times more memory.

### 3.4. Isotropic etching

Isotropic etching is used as a complementary process in connection with anisotropic etching and/or DRIE [59]. For isotropic etchants, the etch rate remains essentially constant along any direction. This behavior can be tested by using a mask with a small circular opening. If the etching process is truly isotropic, a perfect hemispherical cavity will be developed

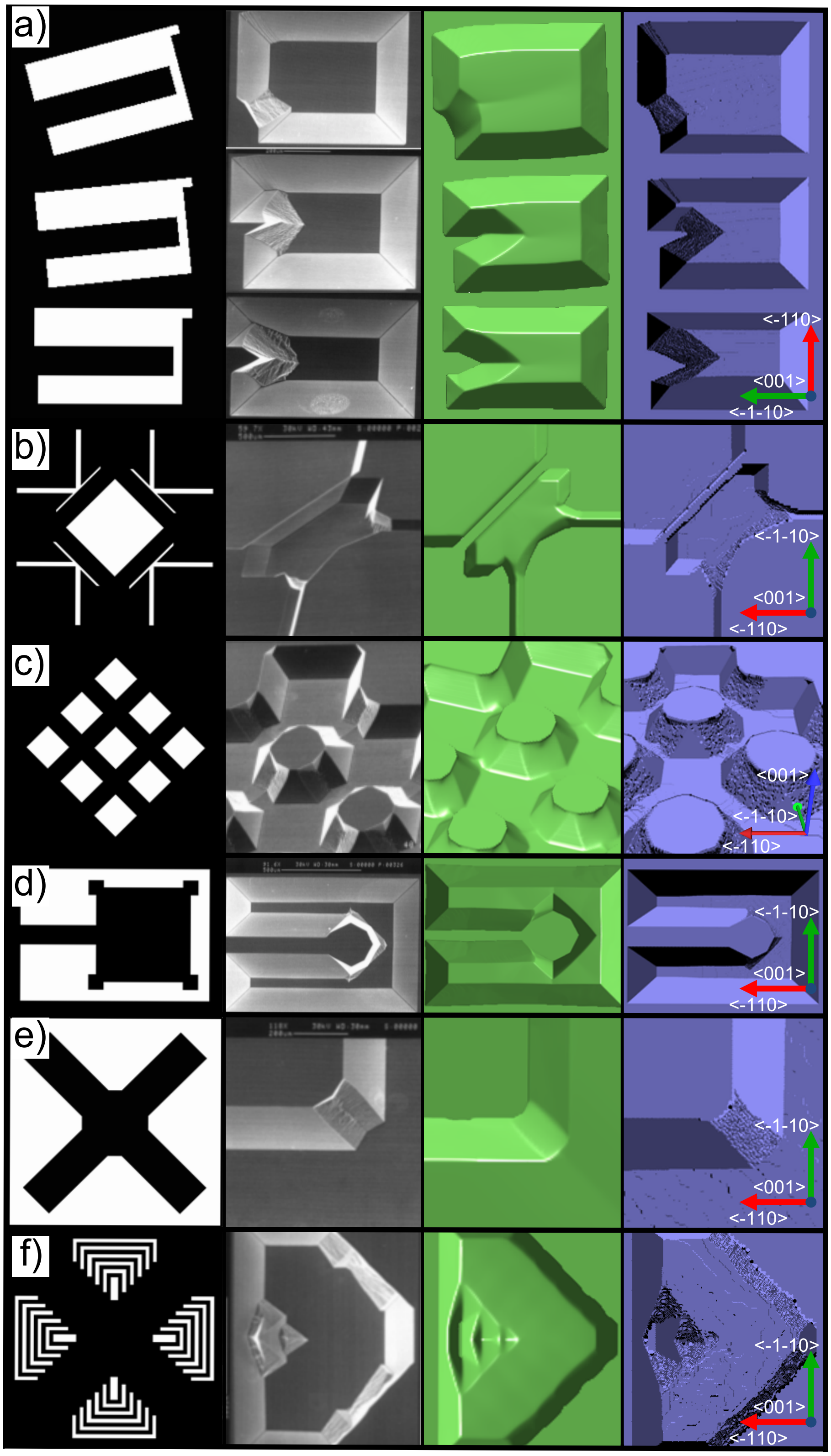
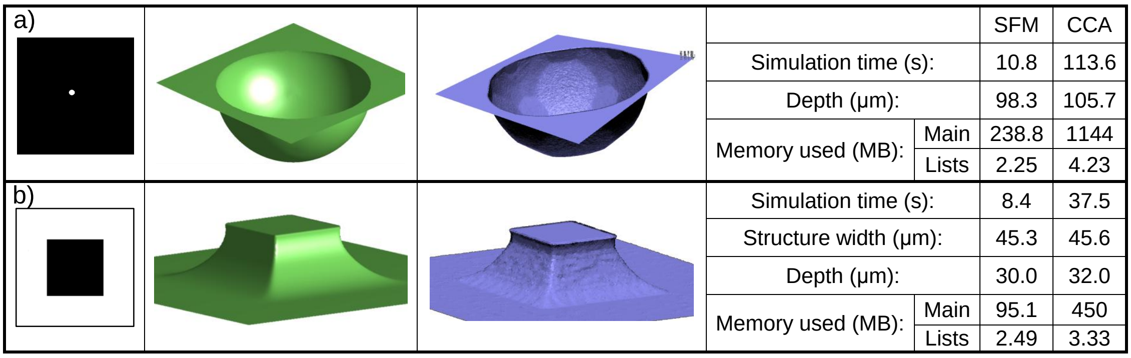


Figure 6: Comparison between experimental and simulated etch structures: (left column) applied masks, (center-left) experiments [58], (center-right) SFM results, (right) CCA results.

| | Experiment features | | | Surface grid size | | Depth ($\mu m$) | | Used memory (MB) | | | | Simulation time (s) | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Substrate size ($\mu m$) | Etching time (min) | Depth ($\mu m$) | SFM (voxels) | CCA (UCs) | SFM | CCA | SFM | | CCA | | SFM | CCA |
| | | | | | | | | Main | Lists | Main | Lists | | |
| a) | 2370x1416 | 200 | 225 | 338x201 | 219x130 | 231.4 | 229.6 | 79.4 | 2.12 | 347.3 | 2.84 | 23.1 | 14.7 |
| b) | 4720x4750 | 150 | 166 | 261x262 | 167x169 | 174.4 | 173.9 | 28.9 | 0.9 | 106.4 | 1.15 | 3.6 | 2.1 |
| c) | 3782x3782 | 200 | 225 | 261x261 | 169x169 | 232.5 | 233.4 | 42.0 | 1.14 | 175.6 | 1.57 | 7.7 | 3.8 |
| d) | 1564x1148 | 200 | 225 | 304x223 | 198x145 | 231.5 | 231.8 | 105.6 | 2.52 | 481.2 | 3.39 | 35.6 | 28.9 |
| e) | 2305x2305 | 200 | 225 | 261x261 | 169x169 | 231.0 | 233.9 | 66.3 | 2.54 | 280.4 | 3.0 | 20.6 | 20.0 |
| f) | 4170x4170 | 200 | 225 | 261x261 | 169x169 | 225.9 | 231.2 | 39.8 | 1.13 | 158.6 | 1.49 | 6.6 | 4.9 |

Table 1: Experimental and simulation details for the structures shown in Fig. 6.



| | | | SFM | CCA |
|---|---|---|---|---|
| a) | Simulation time (s): | | 10.8 | 113.6 |
| | Depth (μm): | | 98.3 | 105.7 |
| | Memory used (MB): | Main | 238.8 | 1144 |
| | | Lists | 2.25 | 4.23 |
| b) | Simulation time (s): | | 8.4 | 37.5 |
| | Structure width (μm): | | 45.3 | 45.6 |
| | Depth (μm): | | 30.0 | 32.0 |
| | Memory used (MB): | Main | 95.1 | 450 |
| | | Lists | 2.49 | 3.33 |

Figure 7: Comparison between SFM and CCA simulations for isotropic etching (etch rate: 1 $\mu m/min$). Substrate size ($\mu m^2$)/etch time (min): (a) 250x250/100; (b) 200x200/30.

into the substrate after prolonged etching. Fig. 7(a) compares the results of such a computational experiment for the SFM and CCA methods. It is concluded that the CCA results are noisier and less isotropic, which is consistent with the etch rate distribution for the isotropic etchant shown in Fig. 2. A similar conclusion is obtained from Fig. 7(b), where a square mask pattern leads to the formation of a mesa structure. In both examples the grid size has been chosen by the same criteria explained in the context of Fig. 5, leading to a surface grid of $261x261$ voxels for SFM and $169x169$ UCs for CCA.

An interesting feature of the SFM method applied to isotropic etching is that it is not necessary to search the values for the artificial viscosity of Eq. 3 since the etch rate is always constant and, thus, $\alpha_l = 1$ in all directions. The spherical coordinates of the normal vector are not required either, thus resulting in a faster algorithm for isotropic etchants. This is reflected in the simulation times shown in Fig. 7, where SFM is 10.5 and 4.5 times faster than CCA, respectively, while the memory used by SFM remains lower, as for the anisotropic examples.

## 4. Exploratory parallel implementations

Although our SFM implementation is able to achieve similar accuracy as the CCA method with less noise in the etch front and roughly 4 times less memory, the computational performance of SFM remains up to 2 times slower than that of CCA for the anisotropic simulations, as reflected in Table 1. To accelerate the performance we present in this Section two parallel-oriented versions of the SFM: (i) gSFM, a truly parallel implementation based on the Nvidia CUDA architecture of modern GPUs, which enables the simultaneous processing of thousands of threads and, thus, substantial savings in the overall computational time, and (ii) cSFM, a parallel-oriented version based on the multi-threading capabilities of modern multicore CPUs, enhanced by the use compilation options, such as the Streaming SIMD Extension (SSE) instruction set, which optimizes the computation of the same instruction over multiple data in a single clock cycle for single/multi-core CPUs. Although the LS method has been successfully parallelized previously, leading to substantially faster performance [60, 37], such procedures have never been applied to the simulation of wet etching for improved MEMS design.

### 4.1. *GPU implementation: gSFM*

The use of linked lists does not fit well the GPU computing philosophy. Thus, we have developed a SFM implementation without this data structure. When launching a GPU algorithm (or kernel), several hundreds/thousands of threads are executed in parallel, each targeting a portion of the whole task. In our implementation, each thread is in charge of evaluating all the grid points (voxels) of a vertical column, as recommended by previous CUDA optimizations for similar data access patterns [61]. In order to mimmic the SFM behavior, each thread keeps track of the upper and lower spatial boundaries of the active region of a column, thus processing only a few voxels instead of the whole column. This requires the use of an auxiliary state variable (*aux_state*) to avoid modifying the current state until required. In addition, only the points labelled as *state=0* (or *state* = ±1, depending on the algorithm step) are processed, while the points labeled as *state* = ±2 are ignored. Using 32-

bit single precision, this implementation fully reproduces the results from the sequential, linked-list version (SFM).

In the CUDA approach the threads are grouped into thread blocks and the kernel launches the execution of a grid of thread blocks, assigning each block to a different Streaming Multiprocessor (SM). The GPU used in this study is the Nvidia GeForce 560 GTX, with 336 cores. Since the multiprocessors execute in parallel groups of 32 threads, called warps, the best performance is obtained when the thread blocks size is a multiple of the warp size. We conclude after some tests that 256 threads per block is the optimal choice. Because the work space is 2D (*i.e.* a collection of vertical columns in 3D) the block size is chosen to be 16x16 threads and the grid of thread blocks launched by each kernel is 2D. The actual grid dimensions depend on the size of the simulated substrate according to:

$$Grid_x = \lceil (substrate_x + 15)/16 \rceil \tag{10}$$

$$Grid_y = \lceil (substrate_y + 15)/16 \rceil \tag{11}$$

where $substrate_x$ and $substrate_y$ are the number of voxels, which define the planar size of the substrate, and $\lceil x \rceil$ denotes the nearest integer towards infinity. This scheme guarantees the execution of one thread per column.

A very important aspect of any CUDA implementation is the need to optimize the memory bandwidth by performing memory coalescing, *i.e.* encouraging/promoting that contiguous threads access contiguous memory locations during read/write operations. This is easily obtained when updating the values of $\phi$, since each thread only needs to access its memory position. However, when each thread needs to access the neighboring $\phi$ values to calculate the spatial derivatives, memory coalescing is better achieved by directly using the texture memories of the GPU, since these are designed for exploiting memory access locality. With the chosen thread memory order, we find that this memory offers good performance for neighborhood access. In addition, the values of the etch rates $R(\theta, \Phi)$ are also stored in the texture memory to improve memory coalescing when calculating the local velocity of the front.

Based on the algorithm shown in Section 2.2, our gSFM implementation makes use of 5 kernels:

- **Kernel1** (Corresponds to step (7.1) of section 2.2)
  *For every column voxel between upper and lower boundaries labeled as $state = 0$, determine:*
  1. *The forward and backward derivatives using Eq. 5.*
  2. *The normal components $N_l = \phi_l / |\nabla \phi|$, where $l = x$, $y$ and $z$.*
  3. *The etch rate (convert the cartesian components to spherical coordinates and access the corresponding matrix element).*
  4. *The maximum value according to Eq. 7.*

- **Kernel2** (corresponds to step (7.3) of section 2.2)
  *For every column voxel between upper and lower boundaries labelled as $state = 0$:*
  1. *Update $\phi$ using Eq. 3.*
  2. *If $\phi^{n+1}_{i,j,k} < -0.5\Delta x$: label $(i, j, k)$ with $aux\_state = -1$.*
  3. *If $\phi^{n+1}_{i,j,k} > 0.5\Delta x$: label $(i, j, k)$ with $aux\_state = +1$.*

- **Kernel3** (corresponds to steps (7.4)-(7.5) of section 2.2)
  *For every column voxel between upper and lower boundaries:*
  - *If $(i, j, k)$ is labelled as $state = 1$*
    1. *Find the neighbor $(i_n, j_n, k_n)$ with $state = 0$ that has the smallest distance value $\phi^{n+1}_{i_n,j_n,k_n}$. Then update: $\phi^{n+1}_{i,j,k} = \phi^{n+1}_{i_n,j_n,k_n} + \Delta x$.*
    2. *If $|\phi^{n+1}_{i,j,k}| \leq 0.5\Delta x$: label $(i, j, k)$ with $aux\_state = 0$.*
    3. *If no $state = 0$ neighbor is found, label $(i, j, k)$ with $state = 2$ and $aux\_state = 2$.*
  - *If $(i, j, k)$ is labelled as $state = -1$*
    1. *Find the neighbor $(i_n, j_n, k_n)$ with $state = 0$ that has the maximum distance value $\phi^{n+1}_{i_n,j_n,k_n}$. Then update: $\phi^{n+1}_{i,j,k} = \phi^{n+1}_{i_n,j_n,k_n} - \Delta x$.*
    2. *If $|\phi^{n+1}_{i,j,k}| \leq 0.5\Delta x$: label $(i, j, k)$ with $aux\_state = 0$.*
    3. *If no $state = 0$ neighbor is found, label $(i, j, k)$ with $state = -2$ and $aux\_state = -2$.*

- **Kernel4** (corresponds to step (7.6)-(7.8) of section 2.2)
  *For every column voxel between one position above upper boundary and one position below lower boundary:*
  1. *Update $state = aux\_state$.*
  2. *If any voxel labelled with $state = 0$ is above (below) the upper (lower) boundary, increment (decrement) it in one voxel.*

- **Kernel5** (corresponds to step (7.9) of section 2.2)
  *For every column voxel between upper and lower boundaries:*
  - *Search for neighbor points $(i_{np}, j_{np}, k_{np})$ with $state = 2$ and:*
    1. *Label $(i_{np}, j_{np}, k_{np})$ with $state = 1$ and $aux\_state = 1$.*
    2. *Update: $\phi^{n+1}_{i_{np},j_{np},k_{np}} = \phi^{n+1}_{i,j,k} + \Delta x$.*
  - *Search for neighbor points $(i_{np}, j_{np}, k_{np})$ with $state = -2$ and:*
    1. *Label $(i_{np}, j_{np}, k_{np})$ with $state = -1$ and $aux\_state = -1$.*
    2. *Update: $\phi^{n+1}_{i_{np},j_{np},k_{np}} = \phi^{n+1}_{i,j,k} - \Delta x$.*

Fig. 8 shows the workflow of the implemented gSFM algorithm. All the steps performed by the CPU are identical to those in the sequential algorithm of Section 2.2, except that the addition/removal of voxels to/from the SFM lists occurs by modifying the corresponding labels of *state* and *aux_state* of each voxel. In order to search the maximum value of Eq. 7 over every voxel with $state = 0$, each thread searches the maximum within a grid column and this value is passed to the CPU.

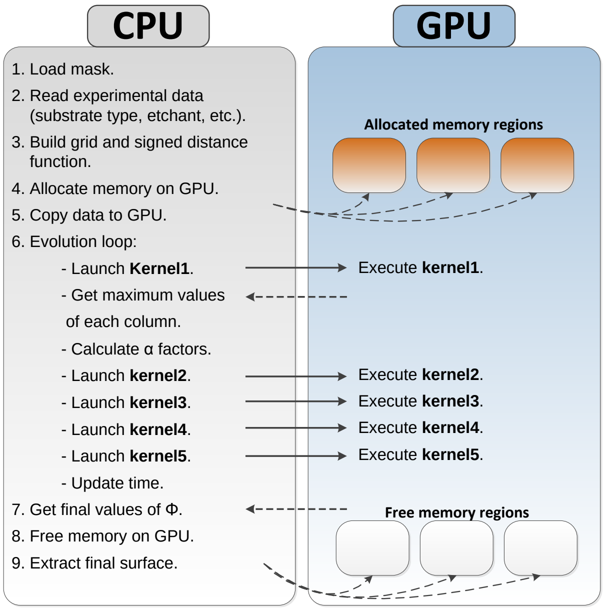


Figure 8: Workflow of the gSFM implementation: distribution of algorithm steps between the CPU and GPU.

Thus, in each iteration the CPU has to find the maximum value among the maxima found by each thread. Once obtained, the three maxima $\alpha_x$, $\alpha_y$ and $\alpha_z$ are returned to the GPU, which then continues with the next step. Although there is room for further optimization, the time required by the CPU to find the maximum value over a two dimensional grid is negligible while the time spent doing the transfer from the GPU to the CPU is only about 2.5% of the total GPU time. To minimize data transfer from the CPU to the GPU, all the model information is stored in the GPU, in such a way that data transfers take place only at the beginning and end of the simulation, as shown by steps 5 and 7 in Fig. 8. At the beginning, the CPU sends the following data to the GPU: (i) the current values of the signed distance function $\phi^0$, (ii) the etch rates $R(\theta, \Phi)$, (iii) the current *state* of the voxels and (iv) the mask pattern. At the end, the GPU copies the final values of $\phi$ to the CPU.

### 4.2. CPU implementation: cSFM

Since the benchmarks of this study are performed on a quad-core 2.8 GHz Intel Core i7, we consider the possibility to optimize the performance by taking advantage of specific properties of this CPU. This includes the use of (a) Intel's Streaming SIMD Extensions (SSE) for improved data transfer within the microprocessor and (b) multi-threading options due to the multicore nature of the microprocessor.

The SSE instruction set enables applying the same instruction to multiple data objects in one clock cycle. Based on the use of Microsoft Visual Studio 2010 in this study, the code is automatically optimized by generating SSE2 instructions using the */arch:SSE2* compilation option. On the other hand, there are different choices to take advantage of the multicore architecture, such as openMP directives or the Open Source POSIX Threads (Pthreads) library recently adapted to Windows O.S. Based on the use of the Pthreads library in this study, we avoid data conflict by enforcing that (i) each thread creates a sub-list of SFM lists and (ii) access to each list is exclusive for the creator. As an example, if two threads ($i = 0$ and $i = 1$) are created, the following lists are generated: $L0\_sub[i]$, $Lp1\_sub[i]$, $Ln1\_sub[i]$, $S0\_sub[i]$, $Sp1\_sub[i]$ and $Sn1\_sub[i]$. Nevertheless, barriers are required after steps 7.1), 7.2) and 7.4) through 7.9) of the main loop of the algorithm shown in Section 2.2 to ensure that all the values of the global variables, such as the distance function, are updated correctly. Taking advantage of both the SSE optimization and multi-threading, the cSFM implementation with 8 threads is about 3.9 – 4.9 times faster than the unoptimized, sequential code for the simulation of anisotropic etching.

### 4.3. Results

This section compares the computational performance of the cSFM and gSFM implementations. The cSFM program is developed in C while the gSFM code is implemented in CUDA C. Both implementations are executed on a 2.8 GHz Intel Core i7 with 4 GB of RAM and an Nvidia GeForce 560 GTX using single precision (32 bits). In the compilation of the cSFM code the following options have been selected: /arch:SSE2, /fp:fast, /Ox and /Ot. In addition, the best performance is achieved when creating 8 execution threads.

The comparison is made by considering the systems shown in Figs. 6-7. Since both versions implement the SFM the results are identical to those shown in both figures. In addition, the memory use by cSFM is also the same than that presented in Table 1 and Fig. 7. On the other hand, gSFM does not use any linked lists but requires storing the auxiliary state variable, the maximum alpha values of each column as well as the upper and lower boundaries of each thread, implying an small increase of the main memory usage. In terms of computational performance, Table 2 collects the corresponding data for the cSFM/gSFM results of Figs. 6-7. The table demonstrates that the GPU algorithm runs up to 7.4 times faster than the CPU version. Although the results confirm that current CPUs stand as an acceptable platform to perform this type of wet etching simulations, the massively parallel architecture of current GPUs provides better computational efficiency with only a slightly larger use of memory. In fact, gSFM is expected to become increasingly more efficient when simulating larger substrates, since the massively parallel architecture of a GPU is progressively more suitable when more and more threads are created, in contrast to the multi-core CPU. Nevertheless, both implementations of the SFM algorithm produce simulated results within a few seconds and combine both the computational efficiency and the algorithmic accuracy required for the realistic simulation of wet etching fronts for improved MEMS design.

## 5. Conclusions

Based on the Level Set (LS) method, a simulator of anisotropic wet chemical etching is developed, aiming at versatile design of Micro-electro-mechanical systems (MEMS). The

|  | Simulation time (s) |  | Memory usage (MB) |  |
|---|---|---|---|---|
|  | cSFM | gSFM | cSFM | gSFM |
| Fig. 6 a) | 2.7 | 0.6 | 79.4 | 89.8 |
| Fig. 6 b) | 0.37 | 0.15 | 28.9 | 37.9 |
| Fig. 6 c) | 0.84 | 0.24 | 42.0 | 51.4 |
| Fig. 6 d) | 4.2 | 0.77 | 105.6 | 116.79 |
| Fig. 6 e) | 2.8 | 0.43 | 66.3 | 76.43 |
| Fig. 6 f) | 0.77 | 0.22 | 39.8 | 49.14 |
| Fig. 7 a) | 2.15 | 1.34 | 238.8 | 253.78 |
| Fig. 7 b) | 1.63 | 0.22 | 95.1 | 105.99 |

Table 2: Performance comparison between the cSFM and gSFM implementations for the systems shown in Figs. 6-7. The etched shapes of both implementations are identical to those presented in Figs. 6-7.

simulator is based on the Sparse Field Method, which focuses on updating only the active points around the evolving front while efficiently calculating the signed distance at every time step. Accordingly, three LS implementations are considered: (i) **SFM**, a purely-sequential CPU-based Java implementation, which is used to compare the computational performance and simulation accuracy with respect to the current benchmark simulation method for wet etching, namely, the Continuous Cellular Automaton (CCA); (ii) **cSFM**, a parallel-oriented CPU-based version benefitting from (a) the auto-vectorization of the code offered by some compilers and (b) the implementation of multi-threading based on the multicore nature of modern CPUs; and (iii) **gSFM**, a truly-parallel GPU-based implementation, where the most time-consuming tasks are efficiently computed by using the affordable, massively-parallel architecture of modern GPUs. Since the three versions produce exactly the same results, their computational performance can be meaningfully compared.

We conclude that SFM achieves similar accuracy as CCA while producing less fluctuations in the etch front and requiring roughly 4 times less memory, even if SFM needs about double the resolution than CCA. Although for highly anisotropic etchants SFM tends to soften the corners and edges, reducing slightly its accuracy, the differences between the simulated features by SFM and CCA are of the order of a few microns for substrates measuring even thousands of microns. In terms of computational performance, CCA is up to 2 times faster than SFM for anisotropic etchants while SFM becomes up to 10 times faster than CCA for isotropic etchants, for which SFM provides a smooth alternative to the noisy CCA results. In comparison, we find that cSFM and gSFM provide the same accuracy as SFM while gSFM results up to 7.4 times faster than cSFM. We conclude that the massively parallel platforms are more suitable for performing SFM simulations of wet etching and, thus, have a large potential to successfully simulate the process efficiently and accurately in the near future.

Compared to CCA, the largest strengths of the Level Set implementations for the simulation of wet etching are: (i) the absence of a time-consuming calibration procedure prior to performing the simulations, which is strictly necessary in the CCA approach when the etchant is modified, (ii) the direct application of the simulation tool to any type of substrate, which typically requires a dedicated effort to analyze and classify the different atomistic neighborhoods in the CCA approach, (iii) the smaller use of memory in comparison to CCA, and (iv) the faster simulation of isotropic etchants. For these reasons, the proposed gSFM implementation provides accurate and fast simulations. Future benchmarks with parallel implementations of the CCA method are required in order to determine which method is more efficient in parallel environments.

## Acknowledgements

We thank the anonymous reviewers for their valuable comments and suggestions. This work has been supported by the Spanish FPI-MICINN BES-2011-045940 grant and the Ramón y Cajal Fellowship Program by the Spanish Ministry of Science and Innovation. Also, we acknowledge support by the JAE-Doc grant form the Junta para la Ampliación de Estudios program co-funded by FSE and the Professor Partnership Program by NVIDIA Corporation.

[1] D. F. Weirauch, Correlation of the anisotropic etching of single-crystal silicon spheres and wafers, Journal of Applied Physics 46 (4) (1975) 1478–1483.

[2] H. Seidel, L. Csepregi, A. Heuberger, H. Baumgartel, Anisotropic etching of crystalline silicon in alkaline solutions: I . orientation dependence and behavior of passivation layers, Journal of The Electrochemical Society 137 (11) (1990) 3612–3626.

[3] D. Z. D, J. Frühauf, Determination of rates for orientation-dependent etching, Sensors and Actuators A: Physical 48 (2) (1995) 151–156.

[4] R. A. Wind, M. A. Hines, Macroscopic etch anisotropies and microscopic reaction mechanisms: a micromachined structure for the rapid assay of etchant anisotropy, Surface Science 460 (13) (2000) 21–38.

[5] M. A. Gosálvez, K. Sato, A. S. Foster, R. M. Nieminen, H. Tanaka, An atomistic introduction to anisotropic etching, Journal of Micromechanics and Microengineering 17 (4) (2007) S1–S26.

[6] M. A. Gosálvez, P. Pal, N. Ferrando, H. Hida, K. Sato, Experimental procurement of the complete 3d etch rate distribution of si in anisotropic etchants based on vertically micromachined wagon wheel samples, Journal of Micromechanics and Microengineering 21 (12) (2011) 125007.

[7] K. Sato, M. Shikida, Y. Matsushima, T. Yamashiro, K. Asaumi, Y. Iriye, M. Yamamoto, Characterization of orientation-dependent etching properties of single-crystal silicon: effects of koh concentration, Sensors and Actuators A: Physical 64 (1) (1998) 87–93.

[8] I. Zubel, M. Kramkowska, The effect of alcohol additives on etching characteristics in koh solutions, Sensors and Actuators A: Physical 101 (3) (2002) 255–261.

[9] A. Charbonnieras, C. Tellier, Characterization of the anisotropic chemical attack of {hk0} silicon plates in a t.m.a.h. solution: Determination of a database, Sensors and Actuators A: Physical 77 (2) (1999) 81–97.

[10] M. Shikida, K. Sato, K. Tokoro, D. Uchikawa, Differences in anisotropic etching properties of koh and tmah solutions, Sensors and Actuators A: Physical 80 (2) (2000) 179–188.

[11] M. A. Gosálvez, I. Zubel, E. Viinikka, Wet etching of silicon, Chap. Ed. H. Seidel in Silicon Based MEMS Materials & Technologies, Part 4, Chapter 24, Book Ed. V. Lindroos, M. Tilli A. Lehto and T. Motooka, Micro & Nano Technologies, William Andrew/Elsevier, 2010.
URL `http://books.google.es/books?id=BLMiELHNj9sC`

[12] P. Pal, K. Sato, Various shapes of silicon freestanding microfluidic channels and microstructures in one-step lithography, Journal of Micromechanics and Microengineering 19 (5) (2009) 055003.

[13] P. Pal, M. A. Gosálvez, K. Sato, Silicon micromachining based on surfactant-added tetramethyl ammonium hydroxide: Etching mechanism and advanced applications, Japanese Journal of Applied Physics 49 (5) (2010) 056702.

[14] I. Zubel, M. Kramkowska, Etch rates and morphology of silicon (h k l) surfaces etched in koh and koh saturated with isopropanol solutions, Sensors and Actuators A: Physical 115 (23) (2004) 549–556.

[15] J. Fruhauf, K. Trautmann, J. Wittig, D. Zielke, A simulation tool for orientation dependent etching, Journal of Micromechanics and Microengineering 3 (3) (1993) 113–115.

[16] K. Asaumi, Y. Iriye, K. Sato, Anisotropic-etching process simulation system microcad analyzing complete 3d etching profiles of single crystal silicon, in: Workshop on Micro Electro Mechanical Systems. MEMS '97, Proceedings, IEEE., Tenth Annual International, 1997, pp. 412–417.

[17] O. Than, S. Büttgenbach, Simulation of anisotropic chemical etching of crystalline silicon using a cellular automata model, Sensors and Actuators A: Physical 45 (1) (1994) 85–89.

[18] H. Camon, A. M. Gue, J. S. Daniel, M. Djafari-Rouhani, Modelling of anisotropic etching in silicon-based sensor application, Sensors and Actuators A 33 (1) (1992) 103–105.

[19] T. Hubbard, E. K. Antonsson, Cellular automata modeling in mems design, Sensors and Materials 9 (1997) .

[20] M. A. Gosálvez, R. Nieminen, P. Kilpinen, E. Haimi, V. Lindroos, Anisotropic wet chemical etching of crystalline silicon: atomistic montecarlo simulations and experiments, Applied Surface Science 178 (14) (2001) 7–26.

[21] Z. Zhu, C. Liu, Micromachining process simulation using a continuous cellular automata method, Journal of Microelectromechanical Systems 9 (2) (2000) 252–261.

[22] M. A. Gosálvez, Y. Xing, K. Sato, Analytical solution of the continuous cellular automaton for anisotropic etching, Journal of Micromechanical Systems 17 (2) (2008) 410–431.

[23] N. Ferrando, M. A. Gosálvez, J. Cerdá, R. Gadea, K. Sato, Faster and exact implementation of the continuous cellular automaton for anisotropic etching simulations, Journal of Micromechanics and Microengineering 21 (2) (2011) 025021.

[24] N. Ferrando, M. A. Gosálvez, J. Cerdá, R. Gadea, K. Sato, Octree-based, gpu implementation of a continuous cellular automaton for the simulation of complex, evolving surfaces, Computer Physics Communications 182 (3) (2011) 628–640.

[25] Z. Moktadir, H. Camon, Monte carlo simulation of anisotropic etching of silicon: investigation of $\langle 111 \rangle$ surface properties, Modelling and Simulation in Materials Science and Engineering 5 (5) (1997) 481–488.

[26] J. Flidr, Y.-C. Huang, M. A. Hines, An atomistic mechanism for the production of two- and three-dimensional etch hillocks on si(111) surfaces, The Journal of Chemical Physics 111 (15) (1999) 6970–6981.

[27] M. A. Gosálvez, R. M. Nieminen, Surface morphology during anisotropic wet chemical etching of crystalline silicon, New Journal of Physics 5 (1) (2003) 100.1–100.28.

[28] Y. Xing, M. A. Gosálvez, K. Sato, M. Tian, H. Yi, Evolutionary determination of kinetic monte carlo rates for the simulation of evolving surfaces in anisotropic etching of silicon, Journal of Micromechanics and Microengineering 22 (8) (2012) 085020.

[29] M. A. Gosálvez, R. Nieminen, P. Kilpinen, E. Haimi, V. Lindroos, Anisotropic wet chemical etching of crystalline silicon: atomistic montecarlo simulations and experiments, Applied Surface Science 178 (14) (2001) 7–26.

[30] Y. Xing, M. A. Gosálvez, K. Sato, H. Yi, Orientation-dependent surface morphology of crystalline silicon during anisotropic etching using a continuous cellular automaton, Journal of Micromechanics and Microengineering 20 (1) (2010) 015019.

[31] Z. F. Zhou, Q. A. Huang, W. H. Li, W. Deng, A cellular automaton-based simulator for silicon anisotropic etching processes considering high index planes, Journal of Micromechanics and Microengineering 17 (4) (2007) S38.

[32] IntelliSense-Corp., Intellietch webpage, `http://www.intellisense.com/modules/IntelliEtch.html` (aug 2011).

[33] M. A. Gosálvez, Y. Xing, K. Sato, R. Nieminen, Discrete and continuous cellular automata for the simulation of propagating surfaces, Sensors and Actuators A: Physical 155 (1) (2009) 98–112.

[34] N. Ferrando, M. A. Gosálvez, R. J. Colom, Evolutionary continuous cellular automaton for the simulation of wet etching of quartz, Journal of Micromechanics and Microengineering 22 (2012) 025021.

[35] S. Osher, J. A. Sethian, Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations, Journal of Computational Physics 79 (1) (1988) 12–49.

[36] H.-K. Zhao, S. Osher, R. Fedkiw, Fast surface reconstruction using the level set method, in: Workshop on Variational and Level Set Methods in Computer Vision. Proceedings. IEEE, 2001, pp. 194–201.

[37] M. Roberts, J. Packer, M. C. Sousa, J. R. Mitchell, A work-efficient gpu algorithm for level set segmentation, in: Proceedings of the Conference on High Performance Graphics, HPG '10, Eurographics Association, 2010, pp. 123–132.

[38] Z. Xu, H. Huang, X. Li, P. Meakin, Phase field and level set methods for modeling solute precipitation and/or dissolution, Computer Physics Communications 183 (1) (2012) 15–19.

[39] H. Ki, Level set method for two-phase incompressible flows under magnetic fields, Computer Physics Communications 181 (6) (2010) 999–1007.

[40] D. Adalsteinsson, J. Sethian, A level set approach to a unified model for etching, deposition, and lithography i: Algorithms and two-dimensional simulations, Journal of Computational Physics 120 (1) (1995) 128–144.

[41] D. Adalsteinsson, J. Sethian, A level set approach to a unified model for etching, deposition, and lithography ii: Three-dimensional simulations, Journal of Computational Physics 122 (2) (1995) 348–366.

[42] D. Adalsteinsson, J. A. Sethian, A level set approach to a unified model for etching, deposition, and lithography iii: Redeposition, reemission, surface diffusion, and complex simulations, Journal of Computational Physics 138 (1) (1997) 193–223.

[43] O. Ertl, S. Selberherr, A fast level set framework for large three-dimensional topography simulations, Computer Physics Communications 180 (8) (2009) 1242–1250.

[44] O. Ertl, S. Selberherr, Three-dimensional level set based bosch process simulations using ray tracing for flux calculation, Microelectronic Engineering 87 (1) (2010) 20–29.

[45] B. Radjenovic, J. K. Lee, M. Radmilovic-Radjenovic, Sparse field level set method for non-convex hamiltonians in 3d plasma etching profile simulations, Computer Physics Communications 174 (2) (2006) 127–132.

[46] B. Radjenovic, M. Radmilovic-Radjenovic, M. Mitric, Nonconvex hamiltonians in three dimensional level set simulations of the wet etching of silicon, Applied Physics Letters 89 (21) (2006) 213102 –213102–2.

[47] B. Radjenovic, M. Radmilovic-Radjenovic, Level set simulations of the anisotropic wet etching process for device fabrication in nanotechnologies, Hemijska industrija 64 (2) (2010) 93–97.

[48] B. Radjenovic, M. Radmilovic-Radjenovic, M. Mitric, Level set approach to anisotropic wet etching of silicon, Sensors 10 (5) (2010) 4950–4967.

[49] B. Radjenovic, M. Radmilovic-Radjenovic, Three-dimensional simulations of the anisotropic etching profile evolution for producing nanoscale devices, Acta Physica Polonica A 119 (3) (2011) 447–450.

[50] R. T. Whitaker, A level-set approach to 3d reconstruction from range data, International Journal of Computer Vision 29 (1998) 203–231.

[51] M. G. Crandall, P. L. Lions, Two approximations of solutions of hamilton-jacobi equations, Mathematics of Computation 43 (167) (1984) 1–19.

[52] C.-W. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, ii, Journal of Computational Physics 83 (1) (1989) 32–78.

[53] S. K. Godunov, A finite difference method for the computation of discontinuous solutions of the equations of fluid dynamics, Matematicheskii Sbornik 47 (1959) 357–393.

[54] D. Adalsteinsson, J. A. Sethian, A fast level set method for propagating interfaces, Journal of Computational Physics 118 (2) (1995) 269–277.

[55] J. Gomes, O. Faugeras, Reconciling distance functions and level sets, Journal of Visual Communication and Image Representation 11 (1999) 209–223.

[56] R. Courant, K. Friedrichs, H. Lewy, Über die partiellen differenzengleichungen der mathematischen physik, Mathematische Annalen 100 (1928) 32–74.

[57] W. E. Lorensen, H. E. Cline, Marching cubes: A high resolution 3d surface construction algorithm, SIGGRAPH Comput. Graph. 21 (4) (1987) 163–169.

[58] Simode, collection of examples (c), gesselschaft fur mikroelektronikanwendung chemnitz mbh.

[59] L. R. Arana, N. de Mas, R. Schmidt, A. J. Franz, M. A. Schmidt, K. F. Jensen, Isotropic etching of silicon in fluorine gas for mems micromachining, Journal of Micromechanics and Microengineering 17 (2) (2007)

384–392.

[60] A. E. Lefohn, J. E. Cates, R. T. Whitaker, Interactive, gpu-based level sets for 3d segmentation, in: Medical Image Computing and Computer-Assisted Intervention-MICCAI, Vol. 2878 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2003, pp. 564–572.

[61] J. J. López, D. Carnicero, N. Ferrando, J. Escolano, Parallelization of the finite-difference time-domain method for room acoustics modelling based on cuda, Mathematical and Computer Modelling (2011) .

[1] D. F. Weirauch, Correlation of the anisotropic etching of single-crystal silicon spheres and wafers, Journal of Applied Physics 46 (4) (1975) 1478–1483.

[2] H. Seidel, L. Csepregi, A. Heuberger, H. Baumgartel, Anisotropic etching of crystalline silicon in alkaline solutions: I . orientation dependence and behavior of passivation layers, Journal of The Electrochemical Society 137 (11) (1990) 3612–3626.

[3] D. Z. D, J. Frühauf, Determination of rates for orientation-dependent etching, Sensors and Actuators A: Physical 48 (2) (1995) 151–156.

[4] R. A. Wind, M. A. Hines, Macroscopic etch anisotropies and microscopic reaction mechanisms: a micromachined structure for the rapid assay of etchant anisotropy, Surface Science 460 (13) (2000) 21–38.

[5] M. A. Gosálvez, K. Sato, A. S. Foster, R. M. Nieminen, H. Tanaka, An atomistic introduction to anisotropic etching, Journal of Micromechanics and Microengineering 17 (4) (2007) S1–S26.

[6] M. A. Gosálvez, P. Pal, N. Ferrando, H. Hida, K. Sato, Experimental procurement of the complete 3d etch rate distribution of si in anisotropic etchants based on vertically micromachined wagon wheel samples, Journal of Micromechanics and Microengineering 21 (12) (2011) 125007.

[7] K. Sato, M. Shikida, Y. Matsushima, T. Yamashiro, K. Asaumi, Y. Iriye, M. Yamamoto, Characterization of orientation-dependent etching properties of single-crystal silicon: effects of koh concentration, Sensors and Actuators A: Physical 64 (1) (1998) 87–93.

[8] I. Zubel, M. Kramkowska, The effect of alcohol additives on etching characteristics in koh solutions, Sensors and Actuators A: Physical 101 (3) (2002) 255–261.

[9] A. Charbonnieras, C. Tellier, Characterization of the anisotropic chemical attack of {hk0} silicon plates in a t.m.a.h. solution: Determination of a database, Sensors and Actuators A: Physical 77 (2) (1999) 81–97.

[10] M. Shikida, K. Sato, K. Tokoro, D. Uchikawa, Differences in anisotropic etching properties of koh and tmah solutions, Sensors and Actuators A: Physical 80 (2) (2000) 179–188.

[11] M. A. Gosálvez, I. Zubel, E. Viinikka, Wet etching of silicon, Chap. Ed. H. Seidel in Silicon Based MEMS Materials & Technologies, Part 4, Chapter 24, Book Ed. V. Lindroos, M. Tilli A. Lehto and T. Motooka, Micro & Nano Technologies, William Andrew/Elsevier, 2010.
URL `http://books.google.es/books?id=BLMiELHNj9sC`

[12] P. Pal, K. Sato, Various shapes of silicon freestanding microfluidic channels and microstructures in one-step lithography, Journal of Micromechanics and Microengineering 19 (5) (2009) 055003.

[13] P. Pal, M. A. Gosálvez, K. Sato, Silicon micromachining based on surfactant-added tetramethyl ammonium hydroxide: Etching mechanism and advanced applications, Japanese Journal of Applied Physics 49 (5) (2010) 056702.

[14] I. Zubel, M. Kramkowska, Etch rates and morphology of silicon (h k l) surfaces etched in koh and koh saturated with isopropanol solutions, Sensors and Actuators A: Physical 115 (23) (2004) 549–556.

[15] J. Fruhauf, K. Trautmann, J. Wittig, D. Zielke, A simulation tool for orientation dependent etching, Journal of Micromechanics and Microengineering 3 (3) (1993) 113–115.

[16] K. Asaumi, Y. Iriye, K. Sato, Anisotropic-etching process simulation system microcad analyzing complete 3d etching profiles of single crystal silicon, in: Workshop on Micro Electro Mechanical Systems. MEMS '97, Proceedings, IEEE., Tenth Annual International, 1997, pp. 412–417.

[17] O. Than, S. Büttgenbach, Simulation of anisotropic chemical etching of crystalline silicon using a cellular automata model, Sensors and Actuators A: Physical 45 (1) (1994) 85–89.

[18] H. Camon, A. M. Gue, J. S. Daniel, M. Djafari-Rouhani, Modelling of anisotropic etching in silicon-based sensor application, Sensors and Actuators A 33 (1) (1992) 103–105.

[19] T. Hubbard, E. K. Antonsson, Cellular automata modeling in mems design, Sensors and Materials 9 (1997) .

[20] M. A. Gosálvez, R. Nieminen, P. Kilpinen, E. Haimi, V. Lindroos, Anisotropic wet chemical etching of crystalline silicon: atomistic monte-carlo simulations and experiments, Applied Surface Science 178 (14) (2001) 7–26.

[21] Z. Zhu, C. Liu, Micromachining process simulation using a continuous cellular automata method, Journal of Microelectromechanical Systems 9 (2) (2000) 252–261.

[22] M. A. Gosálvez, Y. Xing, K. Sato, Analytical solution of the continuous cellular automaton for anisotropic etching, Journal of Micromechanical Systems 17 (2) (2008) 410–431.

[23] N. Ferrando, M. A. Gosálvez, J. Cerdá, R. Gadea, K. Sato, Faster and exact implementation of the continuous cellular automaton for anisotropic etching simulations, Journal of Micromechanics and Microengineering 21 (2) (2011) 025021.

[24] N. Ferrando, M. A. Gosálvez, J. Cerdá, R. Gadea, K. Sato, Octree-based, gpu implementation of a continuous cellular automaton for the simulation of complex, evolving surfaces, Computer Physics Communications 182 (3) (2011) 628–640.

[25] Z. Moktadir, H. Camon, Monte carlo simulation of anisotropic etching of silicon: investigation of $\langle 111 \rangle$ surface properties, Modelling and Simulation in Materials Science and Engineering 5 (5) (1997) 481–488.

[26] J. Flidr, Y.-C. Huang, M. A. Hines, An atomistic mechanism for the production of two- and three-dimensional etch hillocks on si(111) surfaces, The Journal of Chemical Physics 111 (15) (1999) 6970–6981.

[27] M. A. Gosálvez, R. M. Nieminen, Surface morphology during anisotropic wet chemical etching of crystalline silicon, New Journal of Physics 5 (1) (2003) 100.1–100.28.

[28] Y. Xing, M. A. Gosálvez, K. Sato, M. Tian, H. Yi, Evolutionary determination of kinetic monte carlo rates for the simulation of evolving surfaces in anisotropic etching of silicon, Journal of Micromechanics and Microengineering 22 (8) (2012) 085020.

[29] M. A. Gosálvez, R. Nieminen, P. Kilpinen, E. Haimi, V. Lindroos, Anisotropic wet chemical etching of crystalline silicon: atomistic monte-carlo simulations and experiments, Applied Surface Science 178 (14) (2001) 7–26.

[30] Y. Xing, M. A. Gosálvez, K. Sato, H. Yi, Orientation-dependent surface morphology of crystalline silicon during anisotropic etching using a continuous cellular automaton, Journal of Micromechanics and Microengineering 20 (1) (2010) 015019.

[31] Z. F. Zhou, Q. A. Huang, W. H. Li, W. Deng, A cellular automaton-based simulator for silicon anisotropic etching processes considering high index planes, Journal of Micromechanics and Microengineering 17 (4) (2007) S38.

[32] IntelliSense-Corp., Intellietch webpage, `http://www.intellisense.com/modules/IntelliEtch.html` (aug 2011).

[33] M. A. Gosálvez, Y. Xing, K. Sato, R. Nieminen, Discrete and continuous cellular automata for the simulation of propagating surfaces, Sensors and Actuators A: Physical 155 (1) (2009) 98–112.

[34] N. Ferrando, M. A. Gosálvez, R. J. Colom, Evolutionary continuous cellular automaton for the simulation of wet etching of quartz, Journal of Micromechanics and Microengineering 22 (2012) 025021.

[35] S. Osher, J. A. Sethian, Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations, Journal of Computational Physics 79 (1) (1988) 12–49.

[36] H.-K. Zhao, S. Osher, R. Fedkiw, Fast surface reconstruction using the level set method, in: Workshop on Variational and Level Set Methods in Computer Vision. Proceedings. IEEE, 2001, pp. 194–201.

[37] M. Roberts, J. Packer, M. C. Sousa, J. R. Mitchell, A work-efficient gpu algorithm for level set segmentation, in: Proceedings of the Conference on High Performance Graphics, HPG '10, Eurographics Association, 2010, pp. 123–132.

[38] Z. Xu, H. Huang, X. Li, P. Meakin, Phase field and level set methods for modeling solute precipitation and/or dissolution, Computer Physics Communications 183 (1) (2012) 15–19.

[39] H. Ki, Level set method for two-phase incompressible flows under magnetic fields, Computer Physics Communications 181 (6) (2010) 999–1007.

[40] D. Adalsteinsson, J. Sethian, A level set approach to a unified model for etching, deposition, and lithography i: Algorithms and two-dimensional simulations, Journal of Computational Physics 120 (1) (1995) 128–144.

[41] D. Adalsteinsson, J. Sethian, A level set approach to a unified model for etching, deposition, and lithography ii: Three-dimensional simulations, Journal of Computational Physics 122 (2) (1995) 348–366.

[42] D. Adalsteinsson, J. A. Sethian, A level set approach to a unified model for etching, deposition, and lithography iii: Redeposition, reemission, surface diffusion, and complex simulations, Journal of Computational Physics 138 (1) (1997) 193–223.

[43] O. Ertl, S. Selberherr, A fast level set framework for large three-dimensional topography simulations, Computer Physics Communications 180 (8) (2009) 1242–1250.

[44] O. Ertl, S. Selberherr, Three-dimensional level set based bosch process simulations using ray tracing for flux calculation, Microelectronic Engineering 87 (1) (2010) 20–29.

[45] B. Radjenovic, J. K. Lee, M. Radmilovic-Radjenovic, Sparse field level set method for non-convex hamiltonians in 3d plasma etching profile simulations, Computer Physics Communications 174 (2) (2006) 127–132.

[46] B. Radjenovic, M. Radmilovic-Radjenovic, M. Mitric, Nonconvex hamiltonians in three dimensional level set simulations of the wet etching of silicon, Applied Physics Letters 89 (21) (2006) 213102 –213102–2.

[47] B. Radjenovic, M. Radmilovic-Radjenovic, Level set simulations of the anisotropic wet etching process for device fabrication in nanotechnologies, Hemijska industrija 64 (2) (2010) 93–97.

[48] B. Radjenovic, M. Radmilovic-Radjenovic, M. Mitric, Level set approach to anisotropic wet etching of silicon, Sensors 10 (5) (2010) 4950–4967.

[49] B. Radjenovic, M. Radmilovic-Radjenovic, Three-dimensional simulations of the anisotropic etching profile evolution for producing nanoscale devices, Acta Physica Polonica A 119 (3) (2011) 447–450.

[50] R. T. Whitaker, A level-set approach to 3d reconstruction from range data, International Journal of Computer Vision 29 (1998) 203–231.

[51] M. G. Crandall, P. L. Lions, Two approximations of solutions of hamilton-jacobi equations, Mathematics of Computation 43 (167) (1984) 1–19.

[52] C.-W. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, ii, Journal of Computational Physics 83 (1) (1989) 32–78.

[53] S. K. Godunov, A finite difference method for the computation of discontinuous solutions of the equations of fluid dynamics, Matematicheskii Sbornik 47 (1959) 357–393.

[54] D. Adalsteinsson, J. A. Sethian, A fast level set method for propagating interfaces, Journal of Computational Physics 118 (2) (1995) 269–277.

[55] J. Gomes, O. Faugeras, Reconciling distance functions and level sets, Journal of Visual Communication and Image Representation 11 (1999) 209–223.

[56] R. Courant, K. Friedrichs, H. Lewy, Über die partiellen differenzengleichungen der mathematischen physik, Mathematische Annalen 100 (1928) 32–74.

[57] W. E. Lorensen, H. E. Cline, Marching cubes: A high resolution 3d surface construction algorithm, SIGGRAPH Comput. Graph. 21 (4) (1987) 163–169.

[58] Simode, collection of examples (c), gesselschaft fur mikroelektronikanwendung chemnitz mbh.

[59] L. R. Arana, N. de Mas, R. Schmidt, A. J. Franz, M. A. Schmidt, K. F. Jensen, Isotropic etching of silicon in fluorine gas for mems micromachining, Journal of Micromechanics and Microengineering 17 (2) (2007) 384–392.

[60] A. E. Lefohn, J. E. Cates, R. T. Whitaker, Interactive, gpu-based level sets for 3d segmentation, in: Medical Image Computing and Computer-Assisted Intervention-MICCAI, Vol. 2878 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2003, pp. 564–572.

[61] J. J. López, D. Carnicero, N. Ferrando, J. Escolano, Parallelization of the finite-difference time-domain method for room acoustics modelling based on cuda, Mathematical and Computer Modelling (2011) .